Computational Social Science: An Introductory workshop

Louise Capener

Research Associates at Cathie Marsh Institute and UK Data Service

Table of Contents for this workshop

Computational social science is…

The use of computational and empirical methods to address social science questions.

This requires:

- Human-thinking to identify important research questions,
- Computer-thinking to turn questions into computational/empirical methods,
- Human-thinking to effectively communicate the results.

Computational Social Science is NOT just…

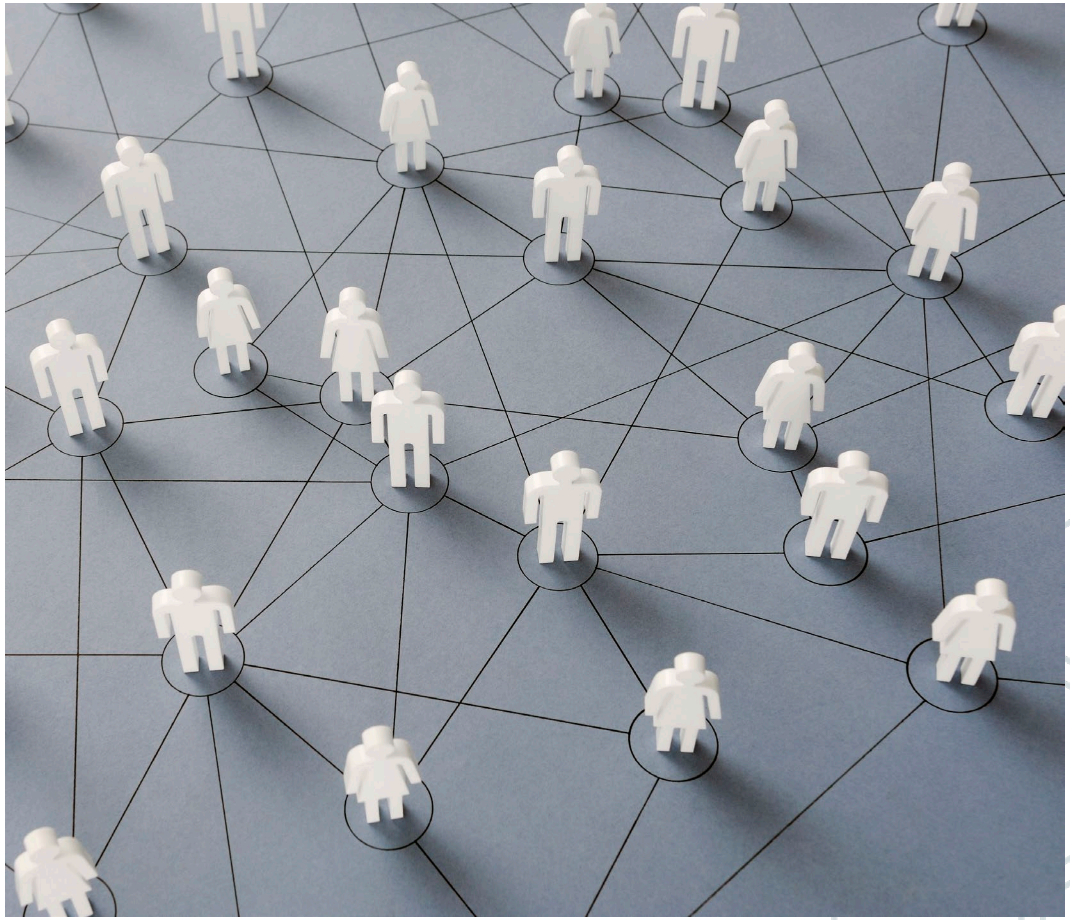
- using computers within a social science research project,
- using digital versions of purely traditional social science methods, or
- using digital but purely non-empirical methods.

Let's have some examples of CSS projects:

- Collect, process, and analyse millions of online news articles to show changing political attitudes
- Use real-time weather and traffic data to show how travellers react
- Combine data from novel wearables/apps to establish correlation between social media activity and heart rate
- Import, process and format centuries of parish records to map family names over time

Key factors in CSS:

- Data volume, complexity, speed, difficulty or novelty is more important than exact data source/type.
- Data must pertain to people, actions, behaviours, choices, statements, etc.
- Exact research question is not important BUT must be a social science question.



In essence, CSS is:

“an opportunity to do socially valuable research that would not be possible without computational methods and tools”

(Halford & Savage 2017)

Interaction Time

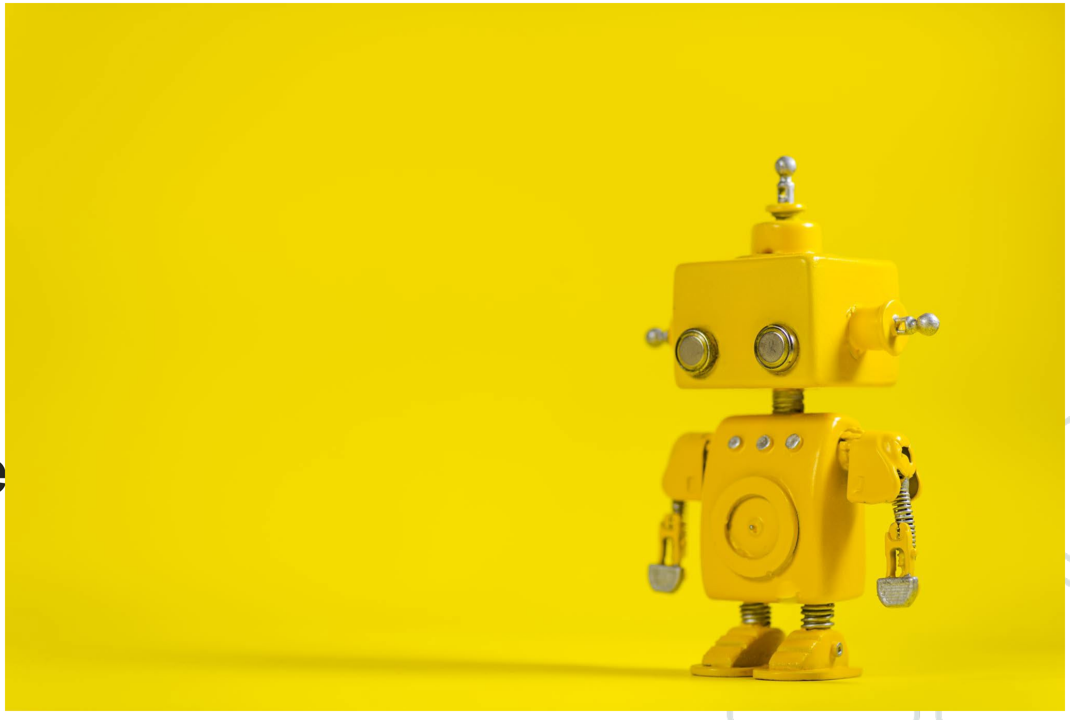
The following slides give you a chance to vote on whether you think the described project is or is not an example of CSS!

Social Scientists …

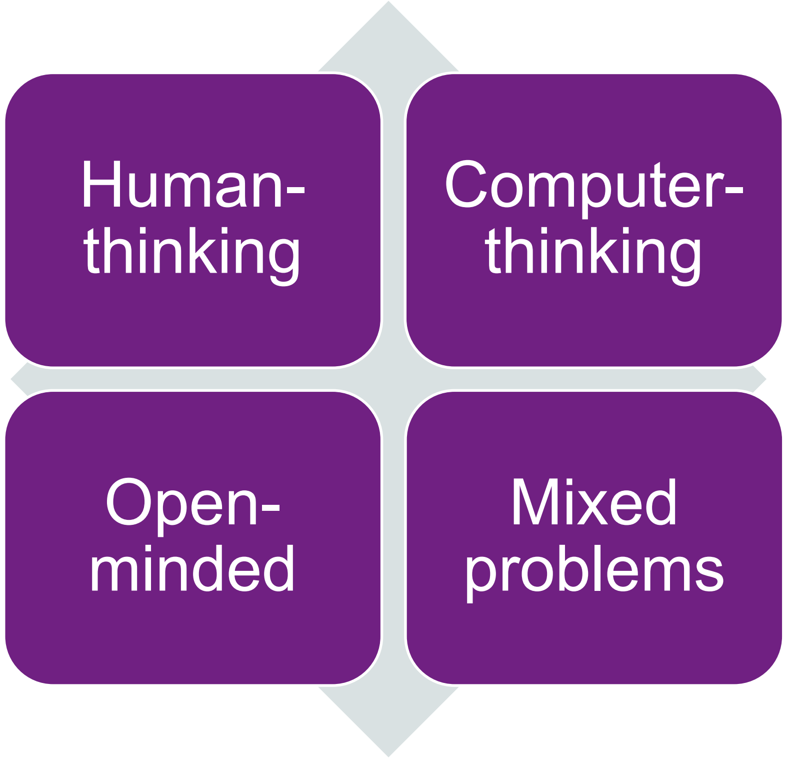
- Social scientists "think like people"
- Study people, interactions, behaviours, etc.
- Thinking skills = abstraction, inference, fuzzy categories, background knowledge, etc.
- Data skills = response categorisation/coding, quality evaluation, pattern detection, etc.
- Use computers, but do not usually write computer code

… and Computer Scientists

- Computer scientists “think like computers”
- Solve information/processing problems
- Thinking skills = concrete definitions, absolutes, strict hierarchies/categories, clearly defined and scoped variables/rules, etc.
- Data skills = Collect/analyse/manipulate data through code/tech/computational methods
- Not usually taught to identify/motivate research projects with societal impact/value



How to do CSS?


Human-thinking
Computer-thinking
Open-minded
Mixed problems

Human-thinking

Skills like:
- Identifying important problems or knowledge gaps,
- Considering possible solutions,
- Connecting problems to relevant theories or perspectives, and
- Collecting relevant information and research to frame approach.

Easy(ish) for social scientists trained in abstraction, communication, subtle context, and shared societal knowledge.

Harder for computer/data scientists not trained in ill-defined, overlapping, context-dependent concepts or using assumptions/background knowledge for interpretation

Computer-thinking

Skills like:
- accessing, organising, processing and vast and/or complex data,
- writing (collaborative) code, and
- documenting workflows.

Easy(ish) for computer/data scientists trained in computational methods, strict rules, exclusive definitions, and extremely formal and structured processes (Jewett and Kling 1991)

Harder for social scientists, but they can build on training to code responses, format surveys, and draw statistical analyses from complex data (among others)

Open-minded (and eager to learn)

No one starts out with all of the skills they need.

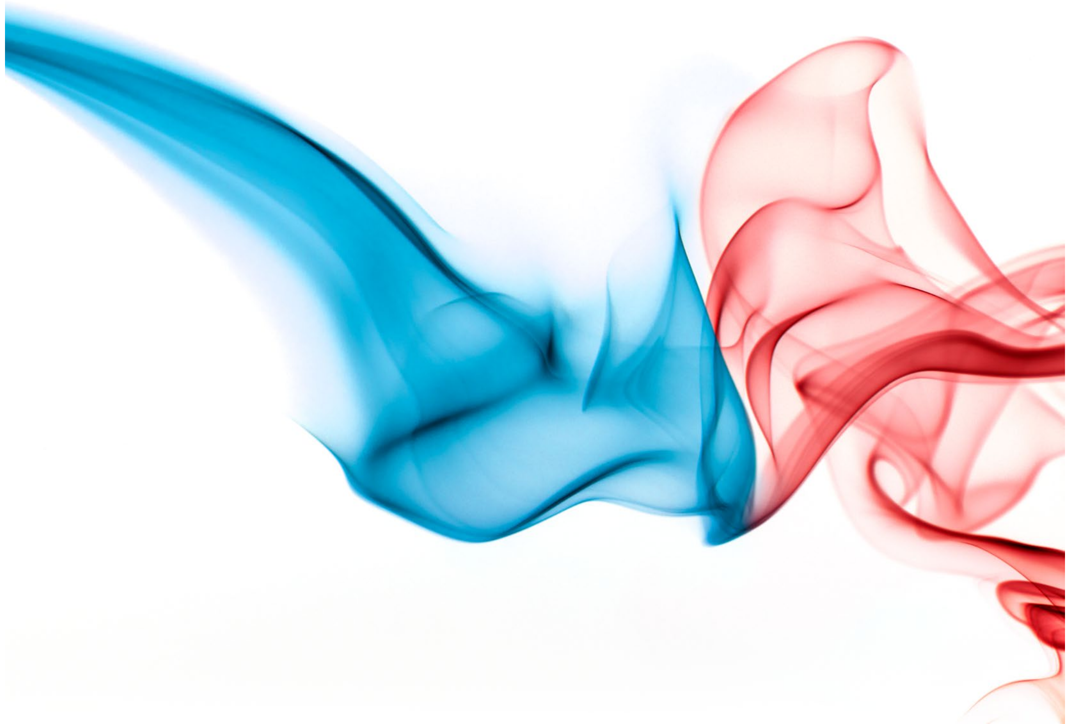
No one knows all the skills they might need to acquire

Approach with an open mind, curiosity, and a willingness to learn.

Some skills will be easier to pick up or use than others.

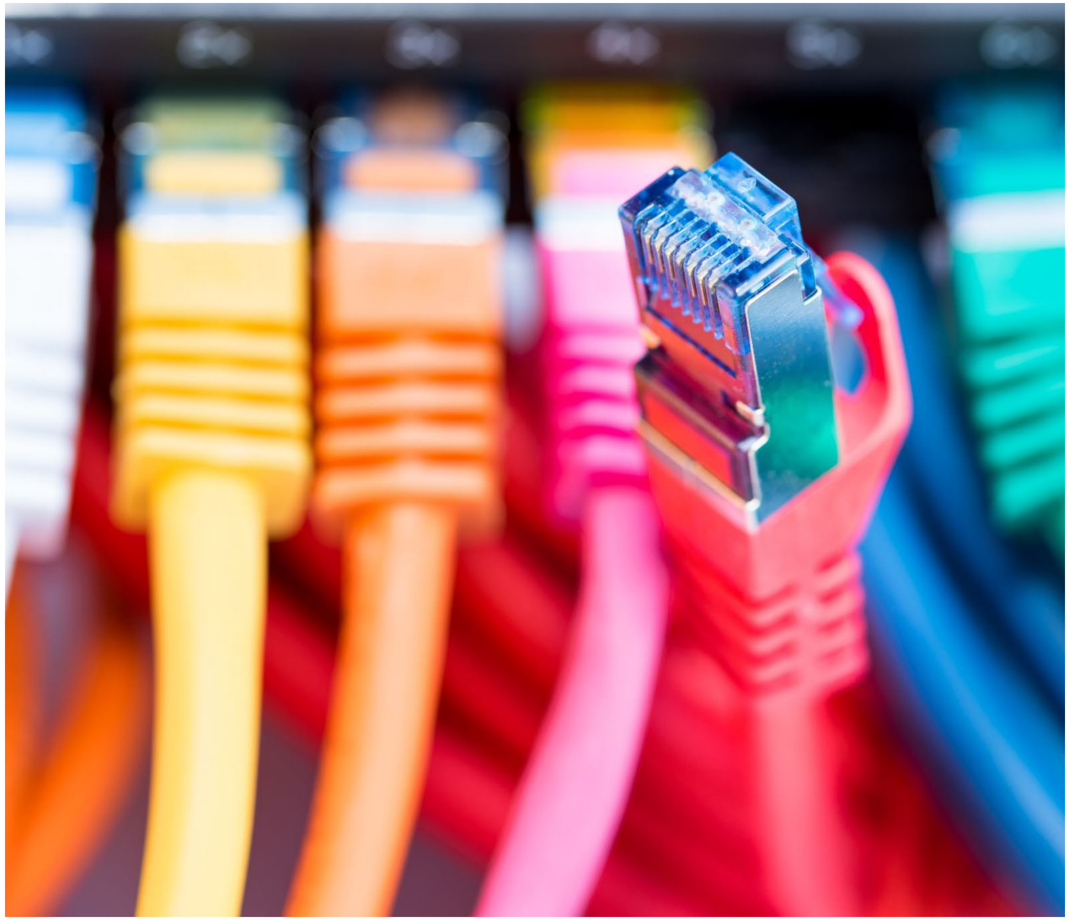
You can’t do it all yourself - be prepared to collaborate.

Mixed problems



- Mixed problems = need human-thinking AND computer-thinking

Mixed problems (and the need to pick 'em)

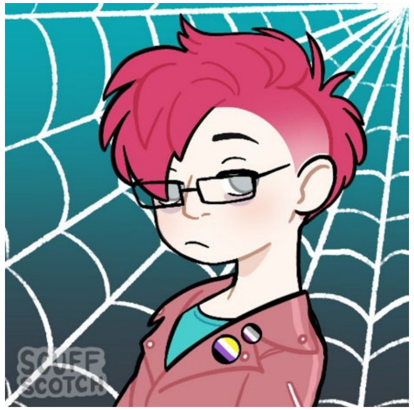


- Mixed problems = need human-thinking AND computer-thinking

- Will become more important as:
- Resources are digitised
- Interactions, objects and processes become 'smart' or networked
- Large volumes of data are made available/are updated faster
- Other changes in the future

Pathways into Computational Social Science

Dr. J. Kasmire



Linguistics (BA)

Evolution of Language and Cognition (MSc)

Transition Management in Complex Adaptive Systems (PhD)

Skills gained:

- text-mining and statistical analysis of human speech and writing.
- Advanced stats and agent-based modelling, data analysis.

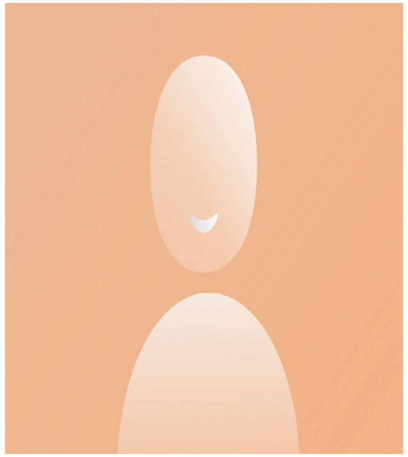
Nadia Kennar

Criminology (BSc)

Criminology and Social Statistics (MRes)

Skills gained:

- Excel/STATA/SPSS and R
- EDA and regression analysis using census and crime data
- Advanced stats and geospatial stats
- Mapping



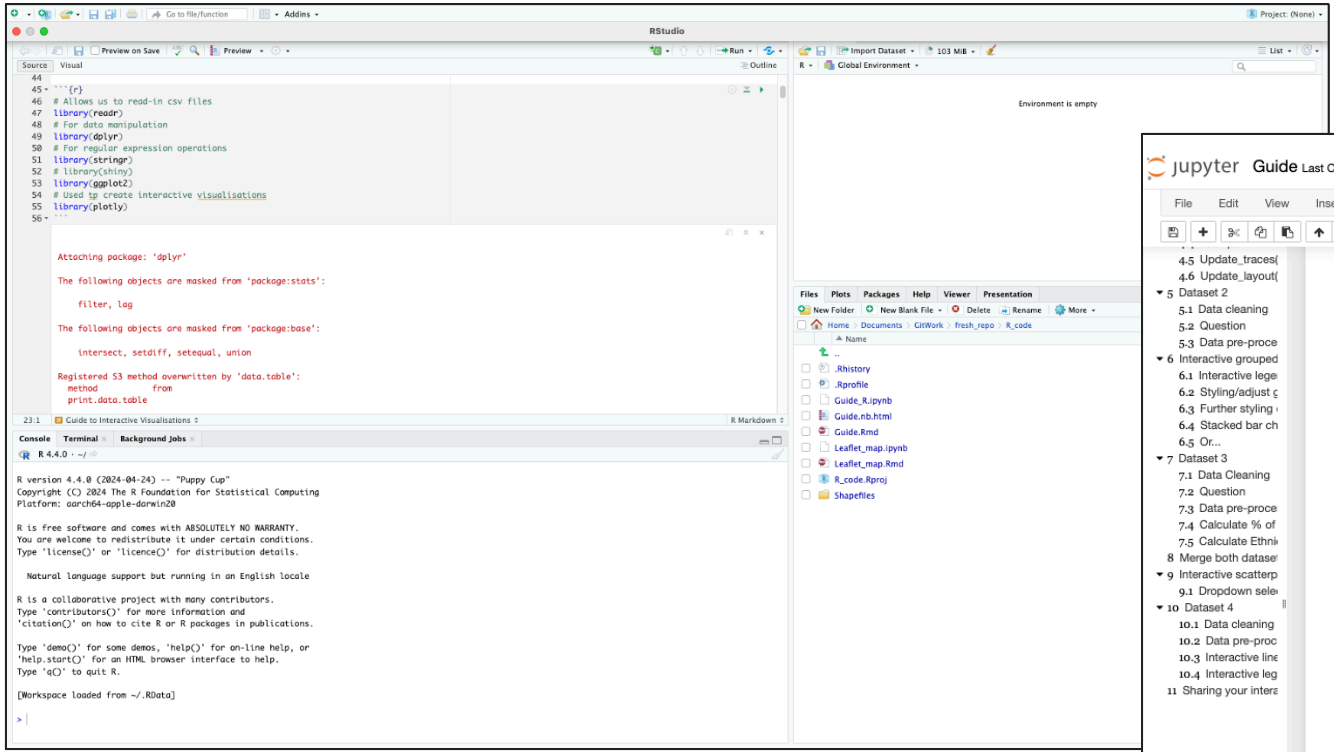
Politics (BA)

Data Science and Artificial Intelligence (MSc)

Skills gained:

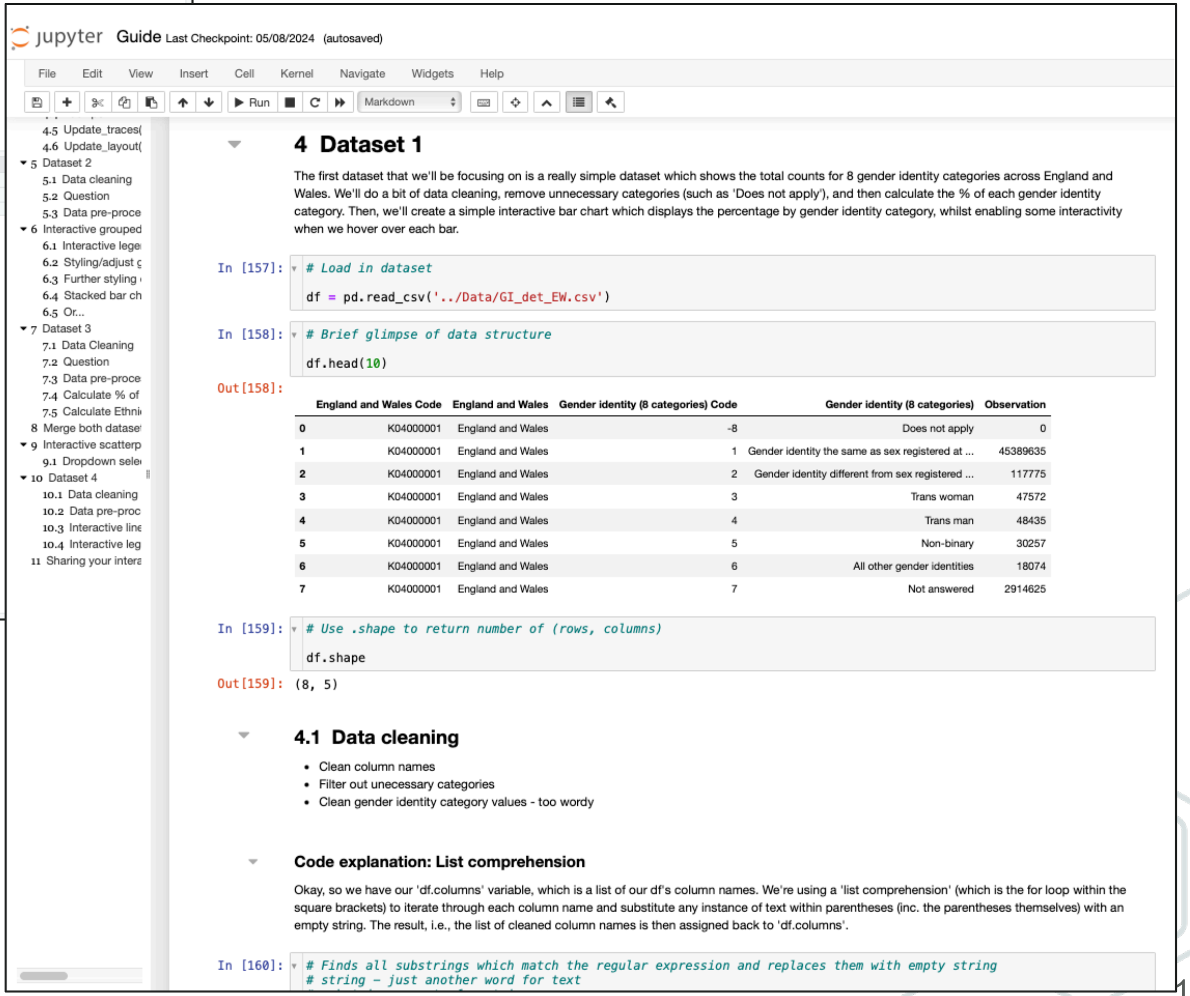
- STATA/SPSS and Python
- Data mining techniques
- Key topics in AI, including ML and NLP

What about coding?



R via RStudio

Python via Jupyter Notebook



Practically, you can follow my 8-step process!

- Identify the problem
- Explore the problem
- Formalise the concepts
- Collect data, implement software, verify
- Experiment and analyse data
- Discussions and conclusions
- Communicate, publish, present
- Share, document and validate

1. Identify the problem

- Be as clear and specific as possible about the pattern, problem, lack of insight.

- Also identify who is involved, where it is, etc.

2. Explore the problem

- Gather information and perspectives in multiple ways (surveys, observations, secondary data analysis, app creation, web-scraping, API's, expert interviews, etc.).

- Spell out sub-problems, processes, relationships, simplifications, assumptions, related issues, existing specialties, etc.

3. Formalise the concepts

- Make all the concepts and processes explicit, formal and both computer and human understandable.

- Often known as ‘pseudo-code’.

- Example:
- “trust” is defined as a variable between 0 and 100.
- “trust” between two parties increases following mutually beneficial interactions.
- Existing levels of “trust” decrease to zero if an interaction is judged to be deceitful.
- Etc.

4. Collect data, implement software, verify

- Select and implement one or more methods.

- The choice of method will be highly dependent on the research topic.

- Thoroughly check that the selected method has been implemented correctly – essentially answering the question “Did we do the thing right?”

5. Experiment and analyse data

- Run the experiments! Build the models! Analyse the data! Or otherwise use the methods selected in previous step!

- Identify and explain the results within the context of the experiments/model/method.

6. Discussions and conclusions

- Going beyond the experiment/model/method, draw some conclusions about what the results mean.

- Do you support policy recommendations?

- Who or what do these results affect? Why does it matter?

- What should change? Who benefits from that proposed change?

7. Communicate, publish, present

- All of the previous steps must be communicated to multiple audiences in multiple ways.

- Short term and long term engagement.

- Public, academic, political, students, etc.

- Consider conferences, journals, blogs, white papers, academic societies, workshops or university classes, etc.

8. Share, document, validate

- Help make sure the ‘right thing was done’ by allowing your work to be studied, reproduced and/or modified as needed through openly available:
- Workflows (methodologies/steps taken)
- Code
- Data

- As transparent, well documented and openly as possible (not always entirely possible)

Important to Note

- These steps are NOT LINEAR!
- Most (or all) will require many ITERATIONS.
- Documentation (step 8) actually applies THROUGHOUT all the other steps – don't wait until the end to start!

References

Arnold, B., L. Bowler, S. Gibson, P. Herterich, R. Higman, A. Krystalli, A. Morley, M. O'Reilly, K. Whitaker and others (2019). The Turing Way: a handbook for reproducible data science (Version v0.0.4).

Brooker, P. D. (2019). Programming with Python for Social Scientists, SAGE Publications Limited.

Halford, S. and M. Savage (2017). "Speaking Sociologically with Big Data: Symphonic. Social Science and the Future for Big Data Research." Sociology **51**(6): 1132--1148.

Heiberger, R. H. and J. R. Riebling (2016). "Installing computational social science: Facing the challenges of new information and communication technologies in social science." Methodological Innovations **9**.

Jewett, T. and R. Kling (1991). "The dynamics of computerization in a social science research team: A case study of infrastructure, strategies, and skills." Social Science Computer Review. **9**(2): 246—275.

Kasmire, J. and D. McDonnell (2020). Thinking like a computational social Scientist: Organising thoughts and organising data. Big Surv '20 Big Data meets Survey Science.

UK Data Service

Thank you.

Louise Capener

Email to louise.capener@manchester.ac.uk

@CapenerLouise on Twitter

UKDS

@UKDataService on Twitter