Big Data and Automated Content Analysis
Part I and II (12 ECTS)

Course Manual

dr. Felicia Loecherbach

Graduate School of Communication
University of Amsterdam

f.loecherbach@uva.nl
www.felicialoecherbach.com

Office: REC-C, 8th floor

Academic Year 2023/24
Semester 2, block 1 and 2

Chapter 1

About this course

This course manual contains general information, guidelines, rules and schedules for the Research Master course Big Data & Automated Content Analysis Part I and II (12 ECTS). Please make sure you read it carefully, as it contains information regarding assignments, deadlines and grading.

1.1 Course description

"Big data" refers to data that are more voluminous, but often also more unstructured and dynamic, than traditionally the case. In Communication Science and the Social Sciences more broadly, this in particular concerns research that draws on Internet-based data sources such as social media, large digital archives, and public comments to news and products. This emerging field of studies is also called *Computational Social Science* (Lazer et al., 2009) or, narrowed down to the analysis of communication, *Computational Communication Science* (Shah, Cappella, & Neuman, 2015).

The course will provide insights in the concepts, challenges and opportunities associated with data so large that traditional research methods (like manual coding) cannot be applied any more and traditional inferential statistics start to loose their meaning. Participants are introduced to strategies and techniques for capturing and analyzing digital data in communication contexts. We will focus on (a) data harvesting, storage, and preprocessing and (b) computer-aided content analysis, including natural language processing (NLP) and computational social science approaches. In particular, we will use advanced machine learning approaches and models like word embeddings.

To participate in this course, students are expected to be interested in learning how to write own programs in Python. Some basic understanding of programming languages is helpful, but not necessary to enter the course. Students without such knowledge are encouraged to follow one of the many (free) online introductions to Python to prepare.

1.2 Goals

Upon completion of this course, the following goals are reached:

A Students can explain the research designs and methods employed in existing research articles on Big Data and automated content analysis.

B Students can on their own and in own words critically discuss the pros and cons of research designs and methods employed in existing research articles on Big Data and automated content analysis; they can, based on this, give a critical evaluation of the methods and, where relevant, give advice to improve the study in question.

C Students can identify research methods from computer science and computational linguistics which can be used for research in the domain of communication science; they can explain the principles of these methods and describe the value of these methods for communication science research.

D Students can on their own formulate a research question and hypotheses for own empirical research in the domain of Big Data.

E Students can on their own chose, execute and report on advanced research methods in the domain of Big Data and automatic content analysis.

F Students know how to collect data with scrapers, crawlers and APIs; they know how to analyze these data and to this end, they have basic knowledge of the programming language Python and know how to use Python-modules for communication science research.

G Students can critically discuss strong and weak points of their own research and suggest improvements.

H Students participate actively: reading the literature carefully and on time, completing assignments carefully and on time, active participation in discussions, and giving feedback on the work of fellow students give evidence of this.

1.3 Help with practical matters

While making your first steps with programming in Python, you will probably have a lot of questions. Nevertheless, `htttps://google.com` and `https://stackoverflow.com` should be your first points of contact. After all, that's how we solve our problems as well...

It is also very much encouraged to try to solve problems together with your classmates. A succesful technique can be *pair programming*, where two colleagues sit behind one screen (for an explanation, see `https://en.wikipedia.org/wiki/Pair_programming`).

If you really cannot find a solution on your own, you can pose such questions during the Friday lab sessions. However, keep the remark in Section 2.8 in mind!

Chapter 2

Rules, assignments, and grading

The final grade of this course will be composed of the grade of two mid-term take home exam ($2 \times 20\%$) and one individual project (60%).

2.1 Mid-term take-home exams ($2 \times 20\%$)

In two mid-term take-home exam, students will show their understanding of the literature and prove they have gained new insights during the lectures and lab sessions. They will be asked to critically assess various approaches to Big Data analysis and make own suggestions for research.

2.2 Final individual project (60%)

The final individual project typically consists of the following elements:

- a carefully collected and relevant dataset of non-trivial size;
- a set of scripts for collecting, preprocessing, and analyzing the data. The scripts should be well-documented and tailored to the specific needs of the own project;
- output files;
- a report with
  - introduction including references to relevant (course) literature, an overarching research question plus subquestions and/or hypotheses (1–2 pages);
  - an overview of the analytic strategy, referring to relevant methods learned in this course;
  - an answer to the RQ;
  - a well-substantiated conclusion and directions for future research.

The project should cover the majority of techniques from the course, unless the approaches contradict each other (e.g., it makes sense to focus on either supervised or unsupervised approaches; data may be either text or image; one may use either webscraping or an API). The project has to

- include techniques from *both* Part I and Part II of the course;
- aim to use the most appropriate technique in case of alternatives. In particular, this means that introductory examples that are discussed for educational purposes only may not be sufficient, if better alternatives ave been discussed in the course as well.

If in doubt, discuss with me in advance!

2.3 Grading and 2nd try

Students have to get a pass (5.5 or higher) for both mid-term take-home exams and the individual project. If the grade of one of the mid-term exams is lower, a re-sit will be organized, typically ten days after the grade is communicated to the student. If the grade of the final project is lower, an improved version can be handed in within ten days after the grade is communicated to the student. If the improved version still is graded lower than

5.5, the course cannot be completed. Improved versions of the final individual project cannot be graded higher than 6.0.

In any case of a second try, the grade of the second try replaces the grade of the first try, regardless of whether it is higher or lower than the first try.

2.4 Presence and participation

Attendance is compulsory. Missing more than three meetings – for whatever reason – means the course cannot be completed.

Next to attending the meetings, students are also required to prepare the assigned literature and to continue working on the programming tasks after the lab sessions. To successfully finish the course, attending the lab sessions is not enough, but has to go hand-in-hand with continuous self-study.

2.5 Staying informed

It is your responsibility to check the means of communications used for this course (i.e., your email account, but – if applicable – also e-learning platforms or any other tool that the lecturer decides to use) on a regular basis, which in most cases means daily.

2.6 Plagiarism & fraud

Plagiarism is a serious academic violation. Cases in which students use material such as online sources or any other sources in their written work and present this material as their own original work without citation/referencing, and thus conduct plagiarism, will be reported to the Examencommissie of the Department of Communication without any further negotiation. If the committee comes to the conclusion that a student has indeed committed plagiarism the course cannot be completed. General UvA regulations about fraud and plagiarism apply, see also `https://student.uva.nl/en/content/az/plagiarism-and-fraud/plagiarism-and-fraud.html`.

2.6.1 Attributing code

Plagiarism rules apply to code in the same way as they apply to text. Just like how your writing often builds on other scholars (who you cite), your code often builds on what others have done or are inspired by solutions of others, for instance in communities like Stackoverflow. Analoguous to plagiarism regulations in normal writing, also in your code, it needs to be transparent *what is from you and what is not*. We don't really have something like APA references for code, but you can simply use `#comments` for that purpose, such as: `# The following function is copied from https://stackoverflow.com/XXXXX/XXXXX` for (almost) literal copy-pasting, or `# The code in this cell is inspired by https://...; I modified Y and Z`.

Make sure that there is no doubt about what is yours and what is from someone else!

2.6.2 Use of AI tools

It speaks for itself that – analogous to the rules outlined above – the use of AI tools such as ChatGPT or Copilot to generate code is not acceptable. Handing in such code will be considered fraud.

2.6.3 Working together

The take-home exams as well as the final project are *individual exams*. Collaboration with fellow students is thus considered academic fraud and will be reported and handled accordingly.

In some instances, some overlap between final projects may be unavoidable, for instance when relying on very similar data sources. If in doubt, always discuss with me in advance to avoid problems after you handed in! It is almost always possible to find a solution, if discussed in advance.

In contrast to these graded assignments, for all ***ungraded*** *assignments and exercises, collaboration is explicitly encouraged. Experience with this course over the last decade has shown that those students who met regularly with classmates to go over the week's code examples and exercises together enjoyed the course most, and also performed best.*

2.7 Deadlines and handing in

Per assignment, the lecturer will specify whether it has to be handed in via Canvas or via Filesender. If no specific instructions are given, or if there is any issue with submitting an assignment through Canvas, use Filesender. Please send all assignments and papers as a PDF file (and do not use formats like .docx) to ensure that it can be read and is displayed the same way on any device. Multiple files should be compressed and handed in as one .zip file or .tar.gz file. Do not email assignments directly but send them via `https://filesender.surf.nl/` to my mail-address. This way, you can also transfer huge files.

Final papers and take-home exams that are not handed in on time, will be not be graded and receive the grade 1. This rule also applies for any other assignment that might be given. The deadline is only met when the all files are submitted.

2.8 Your computer and operating system: your responsibility

Python is a language, not a program, and there are many ways of running Python code.

It is really hard to help with stuff related to your computer, your operating system, your keyboard etc. Things may work slightly different on Windows compared to MacOS or Linux; it's your responsibility to take care of these things. For instance, a downloaded file may be put under `/home/felicia/Downloads` on Linux, under `/Users/felicia/Downloads` on MacOS, and under `c:\\Users \\Felicia\\Downloads` on Windows – or maybe something very different. You are expected to have a sufficiently deep understanding of your own computer to deal with these idiosyncrasies yourself.

Therefore you need to make sure that the following things apply. Otherwise, you need to fix it – we won't have room for that during the course. So, do this **before the course starts**.

- You need to have a recent version of a Python interpreter installed, and you need to be able to run Jupyter Notebooks. Chapter 1 of van Atteveldt, Trilling, and Arcila Calderón (2022) gives instructions (you can skip the part about R, though). You can either use Python natively or use Anaconda (see Chapter 1).
- You need to be able to install third-party packages (see Chapter 1). Test this by checking whether you can install the package `shifterator` and then load it with `import shifterator` without getting an error.
- You need to make sure that your keyboard produces straight quotes `"` rather than typographical ones when typing. Test that by making sure that you can run `print("hello world")` in a Jupyter notebook.
- Create a folder for this course at an easy to remember place, such as `/home/felicia/bdaca` (Linux), `/Users/felicia/bdaca` (MacOS), or `c://Users//felicia//bdaca` (Windows). You could potentially also use `c://Users//felicia//Desktop//bdaca` (Win) or `/Users/felicia/Desktop/bdaca` (Mac) or similar. Make sure that you can locate files in that folder in Jupyter Notebook. And remember/write down it's name! Preferably, don't use spaces (can lead to confusion), and Windows users, replace \ by either a double \\ or by forward slashes (/).

If – even after trying – you do not succeed in any of these steps, ask your classmates. If that also does not help, contact me as soon as possible.

Chapter 3

Schedule and Literature

The following schedule gives an overview of the topics covered each week, the obligatory literature that has to be studied each week, and other tasks the students have to complete in preparation of the class. In particular, the schedule shows which chapter of van Atteveldt et al. (2022) will be dealt with. Note that some basic chapters that explain how to install the software we are going to use have to be read before the course starts.

Next to the obligatory literature, the following books provide the interested student with more and deeper information. They are intended for the advanced reader and might be useful for final individual projects, but are by no means required literature. Bear in mind, though, that you may encounter slightly outdated examples.

- VanderPlas, 2016: A book on numpy, pandas, scikit-learn and more. It can also be read online for free on `https://jakevdp.github.io/PythonDataScienceHandbook/`, and the contents are avaibale as Jupyter Notebooks as well `https://github.com/jakevdp/PythonDataScienceHandbook`.
- The pandas cookbook by Julia Evans, a collection of notebooks on github: `https://github.com/jvns/pandas-cookbook`.
- Hovy, 2020: A thin book on bottom-up text analysis in Python with both a bit more math background and ready-to-use Python code implementations.
- Salganik, 2017: Not a book on Python, but on research methods in the digital age. Very readable, and a lots of inspiration and background about techniques covered in our course.

Before the course starts: Prepare your computer.

✔ Chapter 1: Introduction
Make sure that you have a working Python environment installed on your computer. You cannot start the course if you have not done so.

PART I: Basics of Python and ACA

Week 1: Programming for Computational (Communication|Social) Scientists

Wednesday, 7–2. Lecture with exercises.

We discuss what Big Data and Computational (Social|Communication) Science are. We talk about challenges and opportunities as well as the implications for the social sciences in general and communication science in particular. We also pay attention to the tools used in CSS, in particular to the use of Python.

Mandatory readings (in advance): Boyd and Crawford (2012), Kitchin (2014), Hilbert et al. (2019).

Additionally, the journal *Commmunication Methods and Measures* had a special issue (volume 12, issue 2–3) about Computational Communication Science. Read at least the editorial (van Atteveldt & Peng, 2018), but preferably, also some of the articles (you can also do that later in the

course).

Towards the end of the lecture, we will make first contact with writing code.

Friday, 9–2. Lecture with exercises.

✔ Chapter 3: Programming concepts for data analysis
✔ Chapter 4: How to write code

You will get a very gentle introduction to computer programming. During the lecture, you are encouraged to follow the examples on your own laptop.

We will do our first real steps in Python and do some exercises to get the feeling with writing code.

Week 2: From files and APIs to lists, dictionaries, or dataframes

✔ Chapter 5: From file to dataframe and back

We talk about file formats such as `csv` and `json`; about encodings; about reading these formats into basic Python structures such as dictionaries and lists as opposed to reading them into dataframes; and about retrieving such data from local files, as parts of packages, and via an API.

Wednesday, 14–2. Lecture.

A conceptual overview of different file formats and data sources, and some practical guidance on how to handle such data in basic Python and in Pandas.

Friday, 16–2. Lab session.

We will exercise with the data structures we learned in week 1, as well as with different file formats.

Week 3: Data wrangling and exploratory data analysis

Of course, you don't need Python to do statistics. Whether it's R, Stata, or SPSS – you probably already have a tool that you are comfortable with. But you also do not want to switch to a different environment just for getting a correlation. And you definitly don't want to do advanced data wrangling in SPSS... This week, we will discuss different ways of organizing your data (e.g., long vs wide formats) as well as how to do conventional statistical tests and simple plots in Python.

Wedneday, 21–2. Lecture.

✔ Chapter 6: Data wrangling
✔ Chapter 7.1. Simple exploratory data analysis
✔ Chapter 7.2. Visualizing data

We will learn how to get your data in the right shape and how to get a first understanding of your data, using exploratory analysis and visualization techniques. We will cover data wrangling with pandas: converting between wide and long formats (melting and pivoting), aggregating data, joining datasets, and so on.

Friday, 23–2. Lab session.

We will apply the techniques discussed during the lectures to multiple datasets.

Week 4: Machine learning basics

In this week, we will make the transition from classic statistical modeling as you know it from your previous courses to machine learning. We will discuss how both approaches are related (or even identical) and where the differences are.

Wednesday, 28–2. Lecture

✔ Chapter 7.3. Clustering and Dimensionality Reduction
✔ Chapter 8: Statistical Modeling and Supervised Machine Learning
✖ (you can skip 8.4 Deep Learning for now)

We will discuss what unsupervised and supervised machine learning are, what they can be used for, and how they can be evaluated.

Friday, 1–3. Lab session.

Departuring from a brief encounter with statsmodels (Seabold & Perktold, 2010), a library for statistical modelling, you will learn how to work with scikit-learn (Pedregosa et al., 2011), one of the most well-known machine learning libraries.

Week 5: Processing textual data

In this week, we will dive into how to deal with textual data. How is text represented, how can we process it, and how can we extract useful information from it? Unfortunately, text as written by humans usually is pretty messy. We will therefore dive into ways to represent text in a clean(er) way. We will introduce the Bag-of-Words (BOW) representation and show multiple ways of transforming text into matrices.

Wedneday, 6–3. Lecture.

✔ Chapter 9: Processing text
✔ Chapter 10: Text as data
✔ Chapter 11, Sections 11.1–11.3: Automatic analysis of text

This lecture will introduce you to techniques and concepts like lemmatization, stopword removal, n-grams, word counts and word co-occurrances, and regular expressions. We then proceed to introducing BOW representations of text.

Additional recommended background reading on stopwords: Nothman, Qin, and Yurchak (2018).

Friday, 8–3. Lab session.

You will combine the techiques discussed on Wednesday and write a first automated content analysis script.

Take-home exam

In week 5, the first midterm take-home exam is distributed after the Friday meeting. The answer sheets and all files have to be handed in no later than the day before the next meeting, i.e. Tuesday evening (14–3, 23.59).

Week 6: Supervised Approaches to Text Analysis

✔ Chapter 11, Section 11.4: Automatic analysis of text

Wednesday, 13–3. Lecture.

We discuss why and when to choose supervised machine learning approaches as opposed to dictionary- or rule-based approaches, and explore how BOW representations can be used as an input for supervised machine learning.

Mandatory reading: Boumans and Trilling (2016).

Friday, 15–3. Lab session.

Exercises with scikit-learn.

Week 7: Supervised Approaches to Text Analysis II

Wednesday, 20–3. Lecture.

We will continue with the topic in week 8, with special attention on how to find the best model using techniques such as crossvalidation and gridsearch.

Friday, 22–3. Lab session.

Exercises with scikit-learn.

Break between block 1 and 2

PART II: Advanced analyses

Week 8: Beyond Bag-of-Words

Wednesday, 3–4. Lecture with exercises.

✔ Chapter 10.3.3. Word Embeddings
✔ Chapter 8.3.5. Neural Networks

✔ Chapter 8.4. Deep Learning

In this week, we will talk about a problem of standard forms of ACA: they treat words as independent from each other, and as either present or absent. For instance, if "teacher" is a feature in a specific model, and a text mentions "instructor", then this is not captured – even though it probably should matter, at least to some extend. Word embeddings are a technique to overcome this problem. But also, they can reveal hidden biases in the texts they are trained on. You will also be provided with examples for how to apply a word2vec model and get a short introduction to keras.

Mandatory readings (in advance): Kusner, Sun, Kolkin, and Weinberger (2015) and Garg, Schiebinger, Jurafsky, and Zou (2018).

Friday, 5–4. No meeting (Good Friday).

Week 9: Transformers

Wednesday, 10–4.

In this lecture, we will introduce Transformer models such as BERT. These models have revolutionized the field in many ways. On the one hand, they have lead to large perfomance increases for many tasks, and they make impressive applications like ChatGPT possible. On the other hand, they form a black box and require extraordinary resources to create. We will discuss the idea behind transformers, introduce the concept of *finetuning* such a pretrained model, and briefly mention few-shot and zero-shot learning.

Mandatory reading (in advance): Lin, Welbers, Vermeer, and Trilling (2023) and Bender, Gebru, McMillan-Major, and Shmitchell (2021).

Friday, 12–4. Lab session.

We will exercise with finetuning a transformer model using the Huggingface library.

Week 10: Intermezzo: How to gather online data

By now, you know a lot about the analysis of existing data sets – but the big elephant in the room is, of course: how can you get the data to answer your specific research questions?

Reserve some time for exercising in this week. Web scraping can be really hard, because there are so many specifics of specific websites to consider. After all, every website is different, and we need to customize scrapers for every site! At the same time, it is one of the most useful techniques to know, and the majority of students in previous cohorts used web scraping as a part of their final project.

Wednesday, 17–4. Lecture.

✔ Chapter 12: APIs and web scraping

We first discuss the principles behind so-called application programming interfaces (APIs) and learn how to use them to retrieve JSON data. However, not all data that we may be interested in are available in such a format. Therefore, we move on to explore techniques to download data from web pages and to extract meaningful information like the text (or a photo, or a headline, or the author) from an article on `http://nu.nl`, a review (or a price, or a link) from `http://kieskeurig.nl`, or similar.

Friday, 19-4. Lab session.

Exercises with APIs and/or web scraping.

Take-home exam

In week 10, the second midterm take-home exam is distributed after the Friday meeting. The answer sheets and all files have to be handed in no later than the day before the next meeting, i.e. Tuesday evening (25–4, 23.59).

Week 11: Unsupervised Machine Learning for Text

✔ Chapter 11.5. Unsupervised text analysis: Topic modeling and beyond

Wednesday, 24–4. Lecture.

In Part I of this course, we introduced the fundamental distinction between supervised and unsupervised machine learning. Also, when talking about embeddings and transformers, the idea of the unsupervised training on large corpora of text came up again. What we did not discuss so far is the use of unsupervised models for the explorative analysis of text.

A first approach that has historically been employed to do this is to simply apply unsupervised methods such as PCA and k-means clustering on a BOW representation of text – something that you could actually have done already with your knowledge from Part I. Starting from there, we proceed to discuss a second approach, Latent Dirichlet Allication (LDA), also referred to as (a form of) topic modeling.

Both approaches have been influential for the field, but are less of a silver bullet then many students and researchers seem to think. We will therefore introduce a much more state-of-the-art approach that is build on top of a pre-trained Transformer instead of relying on a BOW representation.

Mandatory readings (in advance): Maier et al. (2018) and Grootendorst (2022)

Friday, 26–4. No meeting - day after Koningsdag.

Week 12: No Teaching (UvA Teaching Free Week)

Week 13: Multimedia data

Wednesday, 8–5. Lecture

✔ Chapter 14 Multimedia data
We will look beyond text and discuss approaches to the computational analysis of multimedia data.

Friday, 10–5. No meeting - day after Ascension Day

Week 14: Wrapping up

Wednesday, 15–5. Open Lab.

Opportunity to exercise with APIs and libraries presented during the lecture and/or previous week.

Friday, 17–5. Open Lab.

Open meeting with the possibility to ask last (!) questions regarding the final project.

Final project

Deadline for handing in: Wednesday, 31–5, 23.59.

Literature

Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). *On the dangers of stochastic parrots: Can language models be too big?* (Vol. 1) (No. 1). Association for Computing Machinery. doi: 10.1145/3442188.3445922

Boumans, J. W., & Trilling, D. (2016). Taking stock of the toolkit: An overview of relevant autmated content analysis approaches and techniques for digital journalism scholars. *Digital Journalism*, *4*(1), 8–23. doi: 10.1080/21670811.2015.1096598

Boyd, D., & Crawford, K. (2012). Critical questions for Big Data. *Information, Communication & Society*, *15*(5), 662-679. doi: 10.1080/1369118X.2012.678878

Garg, N., Schiebinger, L., Jurafsky, D., & Zou, J. (2018). Word Embeddings as a Lens to Quantify 100 Years of Gender and Ethnic Stereotypes. *Proceedings of the National Academy of Sciences*, *115*(16), E3635–E3644. doi: 10.1073/pnas.1720347115

Grootendorst, M. (2022). Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*.

Hilbert, M., Barnett, G., Blumenstock, J., Contractor, N., Diesner, J., Frey, S., ... Zhu, J. J. H. (2019). Computational Communication Science : A Methodological Catalyzer for a Maturing Discipline. *International Journal of Communication*, *13*, 3912–3934.

Hovy, D. (2020). *Text analysis in Python for social scientists: Discovery and exploration.* Cambridge, UK: Cambridge University Press. doi: 10.1017/9781108873352

Kitchin, R. (2014). Big Data, new epistemologies and paradigm shifts. *Big Data & Society*, *1*(1), 1–12. doi: 10.1177/2053951714528481

Kusner, M. J., Sun, Y., Kolkin, N. I., & Weinberger, K. Q. (2015). From Word Embeddings To Document Distances. *Proceedings of The 32nd International Conference on Machine Learning*, *37*, 957–966.

Lazer, D., Pentland, A., Adamic, L., Aral, S., Barabási, A.-L., Brewer, D., ... van Alstyne, M. (2009). Computational social science. *Science*, *323*, 721–723. doi: 10.1126/science.1167742

Lin, Z., Welbers, K., Vermeer, S., & Trilling, D. (2023). Beyond discrete genres: Mapping news items onto a multidimensional framework of genre cues. In *International Conference on the Web and Social Media (ICWSM).* (`https://arxiv.org/abs/2212.04185`)

Maier, D., Waldherr, A., Miltner, P., Wiedemann, G., Niekler, A., Keinert, A., ... Adam, S. (2018). Applying LDA topic modeling in communication research: Toward a valid and reliable methodology. *Communication Methods and Measures*, *12*(2-3), 93–118. doi: 10.1080/19312458.2018.1430754

Nothman, J., Qin, H., & Yurchak, R. (2018). Stop Word Lists in Free Open-source Software Packages. In *Proceedings of workshop for nlp open source software (nlp-oss)* (pp. 7–12). Stroudsburg, PA, USA: Association for Computational Linguistics. doi: 10.18653/v1/W18-2502

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Salganik, M. J. (2017). *Bit by bit: Social research in the digital age.* Princeton, NJ: Princeton University Press.

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with Python. In *9th Python in science conference.*

Shah, D. V., Cappella, J. N., & Neuman, W. R. (2015). Big Data, digital media, and computational social science: Possibilities and per-

ils. *The ANNALS of the American Academy of Political and Social Science*, *659*(1), 6–13. doi: 10.1177/0002716215572084

van Atteveldt, W., Trilling, D., & Arcila Calderón, C. (2022). *Computational analysis of communication: A practical introduction to the analysis of texts, networks, and images with code examples in python and r*. Hoboken, NJ: Wiley.

van Atteveldt, W., & Peng, T. Q. (2018). When Communication Meets Computation: Opportunities, Challenges, and Pitfalls in Computational Communication Science. *Communication Methods and Measures*, *12*(2-3), 81–92. doi: 10.1080/19312458.2018.1458084

VanderPlas, J. (2016). *Python data science handbook: Essential tools for working with data*. Sebastopol, CA: O'Reilly.