

Big Data and Automated Content Analysis
(6 ECTS)

Course Manual

dr. Anne Kroon

Graduate School of Communication
University of Amsterdam

a.c.kroon@uva.nl
@annekroon

Office: REC-C, 7th floor

March 1, 2022

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 (6 ECTS). Please make sure you read it carefully, as it contains information regarding assignments, deadlines and grading.

1.1 Course description

“Big data” is a relatively new phenomenon, and 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 even *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 lose 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 ex-

pected to be interested in learning how to write own programs where off-the-shelf software is not available. Some basic understanding of programming languages is helpful, but not necessary to enter the course. Students without such knowledge are encouraged to follow a (free) online course (such as the one at <https://www.codecademy.com/learn/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 choose, execute and report on research methods in the domain of Big Data and automatic content analysis.
- F Students know how to collect data with 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, <http://google.com> and <http://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 successful 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 Thursday 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 one mid-term take home exam (30%) and one individual project (70%).

2.1 Mid-term take-home exam (30%)

In a 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 (70%)

The final individual project typically consists of the following elements:

- 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;
- 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 well-substantiated conclusion with an answer to the RQ and directions for future research.

2.3 Grading and 2nd try

Students have to get a pass (5.5 or higher) for both the mid-term take-home exam and the individual project. If the grade of one of these is lower, an improved version can be handed in within one week 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.

2.4 Presence and participation

Attendance is compulsory. Missing more than two of the 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.

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. Analogous 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 Working together

The take-home exam 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 re-

lying 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. In the past, there were quite some issues as some things work just a little bit different on Windows compared to Linux or MacOS (those two, belong to the UNIX family, are essentially identical for our purposes), which is why

we advised students to install a Virtual Machine with Linux, which allows everyone to have the same environment (see the old book at https://github.com/damian0604/bdaca/blob/master/book/bd-aca_book.pdf). We experimented last year with letting students more freedom in their choice of an environment, and we want to continue to do so.

But because it is really hard to help with stuff related to your computer, your operating system, your keyboard etc., 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 Attevelde, Trilling, and Arcila Calderón (2021) gives instructions (you can skip the part about R, though). You can either use Python natively or use Anaconda (see Chapter 1), or you can also still opt for the Virtual Machine way as outlined in Trilling (2019). **It is important that *you* know how to use it on your machine.**
- 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/damian/bdaca` (Linux), `/Users/damian/bdaca` (MacOS), or `c://Users//damian//bdaca` (Windows). You could potentially also use

`c://Users//damian//Desktop//bdaca` (Win) or `/Users/damian/Desktop/bdaca` (Mac) or similar. Make sure that you can locate files in that folder in Jupyter Notebook. And remember/write down its 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 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. (2021) 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 (e.g., Python 2, now-defunct APIs etc.).

- McKinney, 2012: A lot of examples for data analysis in Python. A PDF of the book can be downloaded for free on <http://it-ebooks.info/book/1041/>.
- VanderPlas, 2016: A more recent 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 available 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.

Each week:

- Read the book chapter and/or literature *before* the Monday session
- Submit questions for the lab sessions (generally the Thursday meetings) no later than Wednesday morning
- Work on writing code during the lab sessions; ask questions if you encounter issues.

Schedule

Week 1: What is Computational Social Science, and why Python?

Monday, 4–4. Lecture.

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 *Communication 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).

Thursday, 7–4. Lab session.

✓ CHAPTER 2: FUN WITH DATA

During the lab session, we will run our first code. We will showcase some possibilities, and leave the technical background for next week.

Week 2: Getting started with Python

Monday, 11–4: Lecture

✓ CHAPTER 3: PROGRAMMING CONCEPTS FOR DATA ANALYSIS

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

Thursday, 8–4. Lab session

✓ CHAPTER 4: HOW TO WRITE CODE

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

Week 3: Data formats

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.

Monday, 18–4. Eastern Monday – no lecture

Thursday, 21–4. Lecture plus lab session

✓ CHAPTER 5: FROM FILE TO DATAFRAME AND BACK

✓ CHAPTER 12.1: USING WEB APIs: FROM OPEN RESOURCES TO TWITTER

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. We will practice with writing a script to collect and handle some JSON data.

Week 4: Data wrangling, simple statistics and visualizations

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 definitely 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.

Monday, 25–4. Short lecture plus lab session.

✓ CHAPTER 6: DATA WRANGLING

We will learn how to do data wrangling with pandas: converting between wide and long formats (melting and pivoting), aggregating data, joining datasets, and so on.

Thursday, 28–4. Short lecture plus lab session.

✓ CHAPTER 7.1. SIMPLE EXPLORATORY DATA ANALYSIS

✓ CHAPTER 7.2. VISUALIZING DATA

Take-home exam

In week 4, the midterm take-home exam is distributed after the Thursday meeting. The answer sheets and all files have to be handed in no later than Monday evening (2–5, 23.59).

Week 5: No Teaching

Monday, 2–5: Teaching-free week UvA

Thursday, 2–5: Teaching-free week UvA

Week 6: Working with text

In this week, we will dive into how to deal with textual data. How is text represented, how can we clean it, and how can we extract useful information from it?

Monday, 9–5: Lecture plus lab session

✓ CHAPTER 9: PROCESSING TEXT

We discuss basic string operations and regular expressions. You will write a script to conduct a top-down automated content analysis, in which you check for the occurrence of predefined patterns or strings, and extract data from text based on regular expressions.

Thursday, 29–5. Lecture

✓ CHAPTER 10: TEXT AS DATA

In this lecture, we will dive a bit deeper 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. This lecture will introduce you to techniques and concepts like stemming, stopword removal, n-grams, word counts and word co-occurrences, and regular expressions. We will do some exercises during the lecture. You are encouraged to practice at home with combining the techniques discussed this week and write a full automated content analysis script using a top-down dictionary or regular-expression approach. Preparation: Mandatory reading: Boumans and Trilling (2016).

Week 7: Machine learning

During the final week, we will discuss the basics of machine learning. You will be introduced to scikit-learn (Pedregosa et al., 2011), one of the most well-known machine learning libraries. We do not have the time to discuss machine learning techniques in depth. Rather, a practical and hands-on introduction is provided.

Monday, 16–5. Lecture

✓ CHAPTER 8: STATISTICAL MODELING AND SUPERVISED MACHINE LEARNING

✗ (YOU CAN SKIP 8.4 DEEP LEARNING)

We will discuss the basics of supervised machine learning, and how its performance can be evaluated.

Thursday, 19–5. Lab session.

We exercise with supervised machine learning as a technique for automated content analysis. Possibility to ask last (!) questions regarding the final project.

Final project

Deadline for handing in: Friday, 26–5, 23.59.

Literature

- Boumans, J. W., & Trilling, D. (2016). Taking stock of the toolkit: An overview of relevant automated 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
- Hilbert, M., Barnett, G., Blumenstock, J., Contractor, N., Diesner, J., Frey, S., ... Zhu, J. J. H. (2019). Computational Communication Science : A Methodological Catalyst 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
- 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
- McKinney, W. (2012). *Python for data analysis*. Sebastopol, CA: O'Reilly.
- 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.
- Shah, D. V., Cappella, J. N., & Neuman, W. R. (2015). Big Data, digital media, and computational social science: Possibilities and perils. *The ANNALS of the American Academy of Political and Social Science*, 659(1), 6–13. doi: 10.1177/0002716215572084
- Trilling, D. (2019). Doing computational social science with Python: An introduction. Version 1.3. *SSRN*. Retrieved from <http://papers.ssrn.com/abstract=2737682>
- van Atteveldt, W., Trilling, D., & Arcila Calderón, C. (2021).
- 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.