# Computational Communication Science 2
# Course Manual

dr. Anne Kroon (a.c.kroon@uva.nl)
dr. Marthe Möller (a.m.moller@uva.nl)

College of Communication
University of Amsterdam

Spring Semester 2022 (block 2)

# Contents

# Chapter 1

# About this course

This course manual contains general information, guidelines, rules and schedules for the course Computational Communication Science 2 (6 ECTs), part of the Communication in the Digital Society Minor offered by the College of Communication at the University of Amsterdam. Please make sure you read it carefully, as it contains information regarding assignments, deadlines and grading.

## 1.1 Course description

A sales website that recommends new products to you personally, a company that uses a chatbot to answer your questions, or an algorithm that automatically identifies and warns you about fake news content: In our digital society, we use computational methods to communicate with each other every day. In this course, we will zoom in on the computational methods that lay behind these new ways of communication. We will explore the basic principles of their design, acquire an understanding of their implications, and learn how these methods can be used for science. We will work with some of these methods ourselves in the weekly tutorials to get hands-on experience with these techniques and experience their advantages and limitations first hand. During weekly lectures, we will critically discuss the role that these methods play in our daily lives and what responsibility we have when working with them. At the end of the course, you will have a basic understanding of the methods that underlie different ways of communication in the digital society, you can formulate an informed opinion about the implications of these new techniques, and you will have some first-hand experience in working with them.

In this 7-week course, each week consists of one lecture that zooms in on a specific computational method and the possible applications of this method and of one tutorial in which we work with this method. Through this mixture of introductions to computational methods in the lectures and a hands-on approach during the tutorials, you will acquire knowledge on computational communication science that continues on the knowledge that you gained in CCS-2. In total, there are 28 contact hours in this course (7x 2-hour lecture and 7x 2-hour tutorial).

## 1.2 Goals

Upon completion of this course, students should:

a Have a general understanding of state-of-the-art computational techniques useful to study communication phenomena in the digital society.

b Have a basic understanding of how to apply rule-based, unsupervised and supervised techniques to answer research questions in the field of communication science.

c Be able to identify key benefits and drawbacks associated with different rule-based and machine learning techniques.

d Have basic knowledge of what communication scientific questions can be answered using computational methods.

e Be able to apply a subset of these techniques independently in order to answer some basic research questions in the field of communication in the digital society.

f Have experience with independently solving problems in Python scripts by gathering information from online platforms.

g Be able to clearly communicate through written texts what steps were taken in a research project using computational methods.

## 1.3  Study materials

During this course, Python is the programming language that we will be working with. Hence, students should bring a laptop to each class, with a working Python environment installed.

In addition, a list of assigned readings is made available on the course Canvas page. All readings are available for download online using the UvA Digital Library or Google Scholar. If a reading is not available online, the material will be made available on the course Canvas page.

# Chapter 2

# Rules, assignments, and grading

Assessment for this course is based on a mixture of individual and group assignments.

## 2.1 Overview of assessments

The overall course grade is based on the following assignments:

- Regular multiple-choice questions (20%)
- Group assignment: Written report (20%)
- Group assignment: Presentation (10%)
- Take-home exam (50%)

## 2.2 Regular multiple-choice questions (20%)

At the start of six of the tutorial meeting, students will be asked to answer four questions about that week's literature and/or the preceeding lecture as well as about the content discussed in that week's lecture. In total, students need to answer 24 questions. To receive full marks for this assignment, 18 questions need to be answered correctly. Hence, if one of the classes during which the MC-questions need to be answered is missed, it is still possible to receive full marks on the assignment.

## 2.3 Group assignment: report (20%)

In groups of 4 students, you will work on a group assignment. See the description here in Section 3

## 2.4 Group assignment: presentation (10%)

In week 7, you will present your group assignment in class.

## 2.5 Take-home exam (50%)

During the course's last lab session, students will recieve a take-home exam. This exam will test students understanding of concepts as discussed in the course. In addition, students are presented with a computational coding challenge on one of the techniques discussed in the course. The take-home exam is an individual assignment.

## 2.6 Grading

Students have to get a pass (5.5 or higher) for (1) the average of the group report and presentation, and (2) for the take-home exam. If the average grade of the group assignments or of the take-home exam is lower than 5.5, an improved version of the written work can be handed in within one week after the grade is communicated to the student(s). If the improved version still is graded lower than 5.5, the course cannot be completed. Improved versions of the group assignment and the take-home exam cannot be graded higher than 6.0.

## 2.7 Deadlines and submitting assignments

Please send all assignments and papers as a PDF file to ensure that it can be read and is displayed the same way on any device. Hard-copies are not required. Multiple files should be compressed and handed in as one .zip file or .tar.gz file. Anything exceeding a reasonable file size (approximately 5 MB) has to be send via `https://filesender.surf.nl/` instead of direct email. Ensure that for all the files and/or folders you submit, your name(s) are included in the file-/ foldername.

Assignments that are not completed on time, will be not be graded and receive the grade 1.

- For the group report and the take-home exam, this means that all required files need to be submitted before the deadline.

- The group presentation needs to held during the assigned class, meaning that slides or any other material used for the presentation needs to be present in class as well as at least one group member to give the presentation.

Note that the deadline of an assignment is only met when the all files are submitted *before* the deadline.

## 2.8 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 Science 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.

## 2.9 Presence and participation

Attending the lectures as well as the tutorial meetings is compulsory. Virtual attendance is only possible *if* this is communicated to and agreed upon by the lecturers beforehand. Maximum two meetings can be missed *if* both instructors are informed of this beforehand. Missing more than two meetings means the course cannot be completed.

As you will have noticed in Computational Communication Science 1 and/or similar courses on coding, developing your skills to work with computational methods requires you to practice regularly and to be proactive when it comes to solving problems in your code. Hence, students of Computational Communication Science 2 are expected to practice with and revise the materials discussed in class at home. By practicing at home regularly in between classes, you will get the most out of this course and you will acquire the skills that you need to continue developing your programming knowledge after you finished this course.

## 2.10 Staying informed

It is your responsibility to check the means of communications used for this course (i.e., your email account, the course Canvas page, and the course Github page) on a regular basis, which in most cases means daily.

# Chapter 3

# Group Assignment

This section describes the group assignment.

## Teams

Form groups of around 4 students (5 is the maximum). Your tutorial teacher will help you with this.

## Datasets

Together with your group, you can select from one of two datasets

1. Dataset on podcasts: `https://www.kaggle.com/listennotes/all-podcast-episodes-published-in-december-2017`. This link will lead you to two .csv files. You and your group may decide yourself whether you want to work with one of the files, or combine both of them.

2. Dataset on books. More specifically `google_books_1299.csv`: `https://www.kaggle.com/bilalyussef/google-books-dataset?select=google_books_1299.csv`.

For both datasets, it is important to note that you do not need to consider all the columns. Make a selection of relevant variables yourself, and argue in your assignment why you have decided to include or exclude specific information.

This dataset will form the basis of the assignment. Your task is to explore this dataset, describe it in a meaningful, data-scientific way, and ultimately, to build a recommender system.

The group assignment consists of three tasks: Writing a research report, exploring a dataset,

and building a simple knowledge-based or content-based recommender system. Specifically, the following tasks are part of this assignment:

## 3.1 Write a research report (30% of final grade)

The research report consists of...

**Method section**

- A description of the steps you took, which type of variables were selected and how they were transformed.

- Explain your analytical strategy;

- What techniques (and why) will you be using to describe the dataset?

- What type of recommender system are you building? Why?

**Result section**

- A description of the dataset (how many observations, what type of variables)

- Results of the inductive analysis (e.g., description of the topics you've found).

- Demonstration of the recommender system; explanation of how it works, and some examples from the type of recommendations you get for different types of input.

## 3.2 Explorative data analysis (30% of final grade)

Explore, pre-process, and clean the dataset, and provide some descriptive analyses.

- Explore the dataset, and inspect what type of relevant variables are present, what data can be used. Select which variables might be of interest and can be used later on.

- Feature engineering is an important step here (keeping in mind the type of descriptive analysis you want to conduct in step 2). The literature and code examples from week 1 and week 2 should help you here.

- Describe the dataset using an inductive analysis.

- Provide a clear description of data you will be working with. E.g., describe the most interesting variables in terms of data 'type', number of unique observations, mean, distribution, etc.

- Plotting the data, to visualise some of the relations in the dataset, is appreciated.

- Describe the dataset using some of the techniques as discussed in week 3 and week 4. For example, apply LDA to describe the number of topics present in the dataset.

## 3.3 Recommender system (20% of final grade)

Build a simple knowledge-based or content-based recommender system.

- Build a recommender system, based on the insights from week 6. It's up to you to decide whether you build a knowledge-based or content-based recommender system.

- Think about relevant features that you want to use in your algorithm design. Based on which features do you want to recommend content?

## 3.4 Quality of the code and documentation (20% of final grade)

Make sure your code is well documented and understandable for people that see your code for the first time.

## 3.5 Handing in

One member of your group can hand in the group assignment until Friday, 20 May, 17:00 via `https://filesender.surf.nl`. Send this to your tutorial teacher: Include n.m.i.a.fatahelrahmanabdulqadir@uva.nl as recipient. **Please compress all files into a single .zip or .tar.gz file and use GroupAssignment as subject line.**

Please include the following files:

## File formats

- A set of scripts used to preprocess and analyse the data, and to build the recommender system

- *You can hand in the answer to task 2 either as one Jupyter Notebook-file, integrating code, output, and explanations or as one .py file containing the Python code and one PDF file with output and explanations.*

- The research report in .pdf format

*Note: You may, but do **not** have to create a shared (public) github repository where you store all code and documentation. In that case, please include the link to the github repository together with the written assignment to your tutorial teacher.*

**Compress all files into one single .zip or .tar.gz file with your group name!** If you want to compress your files on Linux, you can do so as follows. Imagine you have a folder called '/home/anne/groupassignment' in which you have everything you want to hand in, you can do

```
cd /home/anne
tar -czf /home/anne/Desktop/groupassignment-team1.
    tar.gz groupassignment
```

to create a file `groupassignment-team1tar.gz` on your Desktop.

Then go to `https://filesender.surf.nl` and upload the file. You get a mail that confirms that you have handed it.

***Good luck!!!***

# Chapter 4

# Course Schedule

This course is set-up in the following manner. In a regular week, on the Mondays, we have lectures. Here, key concepts are explained from a conceptual and theoretical perspective, and examples of code implementations is python are provided. On the Tuesdays, we have lab sessions. During the lab sessions, we generally start with knowledge questions about that week's literature as well as the preceeding lecture. Afterwards, students will work on assignments that are provided. When students have finished the assignments, they can work on the group assignment. The tutorial lecture (Noon will walk around, and will help students with issues they encounter). If multiple students are encountering the same issues, these problems will be discussed plenarily. Active participation in the tutorial meetings is needed to ensure that students understand the materials, and can work on the (group) assignments.

## Week 1: Course introduction & Working with textual data

## Week 1: Course introduction & Text-as-data

### Monday, 4–4: Lecture

✔ Read this manual and inspect the course Canvas page.
✔ Make sure your computer is ready to use for the course (see "study materials" on Canvas).
✔ Read in advance: **?**.

During the first meeting, we will introduce the course: we explain the course goals and policies. We also take a first look into using computational methods for communication science by discussing what we can learn from analyzing texts.

### Tuesday, 5–4. Lab session

During our first tutorial lecture, we will take what we discussed in the lecture and use this to analyze text.

## Week 2: Text as data

### Monday, 11–4: Lecture

✔ Read in advance: chapter 10: Text as data van Atteveldt, Trilling, and Arc (2021).
✔ Read in advance: Boumans and Trilling (2016a).

During this meeting, we will continue with were we left off in the first week. Using some of the techniques of week 1, we will explore some simple, deductive approaches and reflect on their advantages and disadvantes. As an alternative, we will reflect on some of the benefits of inductive, bottom-up approaches. To this aim, we will discuss Bag-of-Words (BOW) representations of textual data, and discuss multiple ways of transforming text into matrices.

### Tuesdag, 12–4: Lab session

In this week's lab session, we will practice with some simple top-down and bottom-up algorithm for

text analysis. In order to prepare ourselves for more advanced bottom-up techniques in week 4, we will start practicing with transforming textuel to matrices, using `sklearn`'s vectorizers.

## Week 3: Start of the group assignment

### Monday, 18–4: No lecture–UvA teaching free day

### Tuesday 19–4: Lab session

During this tutorial meeting, we will form small groups of 3-4 students and start working on the group assignment. Specifically, we will make a start with exploring the dataset as provided, inspect relevant variables, and start working on cleaning the dataset and division of tasks.

## Week 4: Bottom up approaches to text analysis

### Monday, 25–4: Lecture

✔Read in advance chapter 11: Automatic analysis of text van Atteveldt et al. (2021).
✔Read in advance: Brinberg and Ram (2021).

In this meeting, we will discuss some prominent techniques to analyses text using bottom up techniques. Specifically, we will discuss latent dirichlet allocation, a populair approach to detect topics in textual data. In addition, we will focus on different ways to measure similarity between texts, using cosine and soft cosine similarity.

### Tuesdag, 26–4: Lab session

During this tutorial meeting, we will practice with the concepts and code as introduced in Monday's lecture. We will try to measure the similarity between sets of documents using different approaches. In addition, we will practice with a simple LDA model.
In case you understand all concepts, and finished

the in-class assignments, you can continue working on the group assignment.

## Week 5: Taking a break

### Monday, 2–5: No Lecture – UvA teaching free week

### Tuesday, 3–5: No Lab session – UvA teaching free week

✔ Take a break

*Note that this week is an "education-free week", meaning that there will not be a lecture or a tutorial this week. Take a well-deserved brake and we continue the course in week 6!*

## Week 6: Recommender systems

### Monday, 9–5: Lecture

✔ Read in advance: Möller, Trilling, Helberger, and van Es (2018).
✔ Read in advance: Loecherbach and Trilling (2020).

In today's lecture, we will discuss different types of recommender systems. In order to understand how recommender systems work, an understanding of the concepts as discussed in previous weeks is crucial. Specifically, in order to build a recommender system, one needs to understand how to preprocess data, transfrom textual data to data matrices, and calculate similarity between texts.

### Tuesday, 10–5: Lab session

During this week's lab session, we will practice with designing a recommender system yourself. You will experiment with different settings, and inspect the effects thereof on the recommendation content.

# Week 7: Text classification, part 1

## Monday, 16–5: Lecture

✔ WATCH: HTTPS://WWW.YOUTUBE.COM/WATCH?V=81VTQTZ2PBM.
✔ READ IN ADVANCE VAN ZOONEN AND VAN DER MEER (2016).

In this lecture, we will take a look at text classification. We briefly review strictly rule-based methods and then we will explosure supervised machine learning.

## Tuesday, 17–5: Lab session

This week, you will present your group assignment.

## Deadline group project: Friday 20–5

The deadline for handing in the group assignment is end of this week, **Friday 20–5 at 17:00**.

# Week 8: Text classification, part 2

## Monday, 23–5: Lecture

✔ READ IN ADVANCE JORDAN AND MITCHELL (N.D.).
✔ READ IN ADVANCE MEPPELINK ET AL. (N.D.).

In this lecture, we will take a deeper dive into supervised machine learning. We will discuss some commonly used machine learning models as well as how to validate classifiers.

## Tuesday, 24–5: Lecture

At the end of this last tutorial meeting, you will recieve the take-home exam. You will have time to read it in class, and ask questions if needed.

## Deadline take home exam: Sunday 29–7

Deadline of the take-home exam is **Friday 27–5 at 17:00**.

# literature

Boumans, J. W., & Trilling, D. (2016a). 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

Boumans, J. W., & Trilling, D. (2016b). Taking stock of the toolkit: An overview of relevant automated content analysis approaches and techniques for digital journalism scholars. , *4*(1), 8–23. Retrieved 2022-02-28, from `http://www.tandfonline.com/doi/full/10.1080/21670811.2015.1096598` doi: 10.1080/21670811.2015.1096598

Brinberg, M., & Ram, N. (2021). Do New Romantic Couples Use More Similar Language Over Time? Evidence from Intensive Longitudinal Text Messages. *Journal of Communication*, *71*(3), 454–477. doi: 10.1093/joc/jqab012

Jordan, M. I., & Mitchell, T. M. (n.d.). Machine learning: Trends, perspectives, and prospects. , *349*(6245), 255–260. Retrieved 2021-12-23, from `https://www.science.org/doi/10.1126/science.aaa8415` doi: 10.1126/science.aaa8415

Loecherbach, F., & Trilling, D. (2020). 3bij3 – Developing a framework for researching recommender systems and their effects. *Computational Communication Research*, *2*(1), 53–79. doi: 10.5117/ccr2020.1.003.loec

Meppelink, C. S., Hendriks, H., Trilling, D., van Weert, J. C., Shao, A., & Smit, E. S. (n.d.). Reliable or not? an automated classification of webpages about early childhood vaccination using supervised machine learning. , *104*(6), 1460–1466. Retrieved 2021-12-23, from `https://linkinghub.elsevier.com/retrieve/pii/S0738399120306376` doi: 10.1016/j.pec.2020.11.013

Möller, J., Trilling, D., Helberger, N., & van Es, B. (2018). Do not blame it on the algorithm: an empirical assessment of multiple recommender systems and their impact on content diversity. *Information Communication and Society*, *21*(7), 959–977. Retrieved from `https://doi.org/10.1080/1369118X.2018.1444076` doi: 10.1080/1369118X.2018.1444076

van Atteveldt, W., Trilling, D., & Arc´, C. (2021). *Computational Analysis of Communication: A Practical Introduction to the Analysis of Texts, Networks, and Images with Code Examples in Python and R.* John Wiley & Sons. Retrieved from `https://books.google.nl/books?id=dDk9zgEACAAJ`

Van Zoonen, W., & Van der Meer, T. G. (2016). Social media research: The application of supervised machine learning in organizational communication research. , *63*, 132–141. Retrieved 2021-12-23, from `https://linkinghub.elsevier.com/retrieve/pii/S0747563216303557` doi: 10.1016/j.chb.2016.05.028