Computational Communication Science 2
Course Manual

Course coordination and lectures:
dr. Anne Kroon (a.c.kroon@uva.nl)
dr. Saurabh Khanna (s.khanna@uva.nl)

Lab sessions:
Roeland Dubel (r.dubel@uva.nl)

College of Communication
University of Amsterdam

Spring Semester 2025 (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 lab sessions 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 lab sessions, you will acquire knowledge on computational communication science that continues on the knowledge that you gained in CCS-1. 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 in this syllabus. 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

Assignments, grading, and rules

Assessment for this course is based on a combination 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%)

- Exam (50%)

2.2 Regular multiple-choice questions (20%)

At the start of the lab sessions, students will be asked to answer four questions about that week's literature and/or the preceeding literature as well as about the content discussed in that week's lecture. There will be MC questions in the lab sessions of week 1, 2, 3, 7 and 8. This means that there are five weeks with MC-questions. In total, students can answer 20 questions (5 weeks * 4 questions). To receive full marks for this assignment, 16 questions need to be answered correctly. Hence, even if one of the lab sessions 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, you will work on a group assignment. See the description here in Section 3

2.4 Group assignment: presentation (10%)

In week 6, you will present your group assignment in class.

2.5 Exam (50%)

During the last tutorial meeting of the course, students will make an exam. This assignment 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 final assignment is an individual assignment.

2.6 Grading

Students have to get a pass (5.5 or higher) for (1) the group report, and (2) for the final assignment. If the grade of the group assignments 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 grade of final assignment is lower than 5.5, students have to complete a resit for the assignment. If the improved version of the group assignment or the resit of the final assignment is still graded lower than 5.5, the course cannot be completed. Improved versions of the group assignment and the resit of the final assignment 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. Hardcopies are not required. Multiple files should be compressed and handed in as one .zip file or .tar.gz file. If instructions ask you to submit your assignment by sending it to the instructor, send your assignment 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 final assignment, this means that all required files need to be submitted before the deadline.

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

2.8 Plagiarism & fraud

The provisions of the regulations governing fraud and plagiarism for UvA students apply in full to students of this course.

Students are expected to be aware what plagiarism and fraud is. Consult the Regulations Governing Fraud and Plagiarism for UvA Students before you write a paper or essay. These regulations explain exactly what the university understand plagiarism and fraud to mean, and the rules you must adhere to. In line with these regulations, know that all documents handed in for this course must reflect your own individual knowledge and skills. You cannot make use of or reproduce work from others and present it as your own work. This applies to scientific work, content produced by AIs, and also includes work by fellow students. It means that:

- incorrect citations (not correctly following APA 7 rules for citations),

- working together on an individual assignment,

- using text or other content generated by AI,

- correcting each other's assignments,

- sharing your assignment with others as an example, or

- copying (parts of) an assignment from others (including example papers) or your own work (self-plagiarism – including assignments handed in previous editions of this or other courses)

are not allowed unless explicitly mentioned in the instructions. When in doubt, ask your lecturer.

The program Turnitin will be used for the detection of plagiarism in written assignments. In submitting a text, the student consents to the text being entered in the database of the detection program concerned. Any suspicion of fraud and plagiarism (including self plagiarism) will be reported to the Examinations Board. To avoid doing prohibited things 'by accident' or 'due to ignorance', we urgently advise you to consult the following site about **fraud and plagiarism**: `https://student.uva.nl/en/content/az/plagiarism-and-fraud/plagiarism-and-fraud.html`

2.8.1   Attributing code

Plagiarism rules extend to code as well. It is understandable that your code is inspired by the code on the slides of the lectures, and by solutions found on sources such as `stackoverflow`. That is how coding works; you build on the insights and solutions of others. While it is perfectly fine to rely on solutions provided by others, it is *highly important* that you are transparent with respect to *where you got your code from.* More specifically, it should be clear what is your own code, and which parts are derived from other sources. That is, if you copy-paste code and use it in your group assignment or final assignment, please include a reference to it's source (by simply including a comment like: `# 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.`

Ensure absolute clarity regarding the distinction between your own work and that of others.

2.8.2 Use of AI tools

It is self-evident that, in alignment with the aforementioned rules, the use of AI tools such as ChatGPT or Copilot to generate code is not acceptable. Handing in such code will be considered fraud.

2.9 Presence and participation

Students must attend all meetings as attendance is mandatory. Failing to attend more than one tutorial meeting, regardless of the reason, will result in exclusion from the course. In addition to attending the meetings, students must read all the assigned literature and continue working on programming tasks after the lab sessions. Solely attending the lab sessions is insufficient to successfully finish the course; it must be accompanied by continuous self-study.

2.9.1 Class lateness policy

Students are expected to arrive at class *on time*. Being late twice will be considered as one nonattendance.

2.10 Language

All meetings will be held in English. The assignments must be written and presented in English.

2.11 Study advisors

The Study Advisers from Communication Science (https://student.uva.nl/communication-science/contact/study-advisers/study-advisers-cs.htm) are the go-to persons for all planning related questions concerning the Minor's program and should be the ones contacted when indicated by the lecturers or by the course policies. They may refer you to the Study Advisers from your own course (if you do not follow the Communication Science Bachelor's programme) when needed.

2.12 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.

2.13 Teaching team and contacting the lecturers

In this edition, the course will be taught by Anne Kroon, Saurabh Khanna, and Roeland Dubel. Anne and Saurabh will teach the lectures and Roeland will teach the tutorials.

Please contact Anne *and* Saurabh for any questions related to the logistics of the lectures, assignments, or the course in general. Please contact Roeland for any questions regarding the logistics of the tutorials.

Note that the team will not answer e-mails about specific coding issues (e.g., questions about error messages or general Python issues). Please use the tutorial meetings to ask question related to your specific code.

Chapter 3

Group assignment: Research report with code

This section describes the requirements of the group assignment.

Teams

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

Dataset selection

Together with your group, you can select from one of the following datasets. The datasets are available on Canvas.

1. *News dataset*: This dataset contains internet news and user engagement metrics.

2. *Books dataset*: This dataset is a combination of two data sources, the Goodreads dataset and the 7k Books dataset.

3. *Recipe dataset*: This dataset contains data on numerous recipes.

4. *TV Series dataset*: This dataset contains details on TV series TV series

When working with one of these datasets, it's crucial to note that not all columns need to be taken into consideration. You should choose the relevant variables yourself and provide justification in your assignment for including or excluding specific information.

The assignment will be centered around the choosen dataset. Your objective is to analyze the dataset in a data-scientific manner, and ultimately develop a recommender system.

The group assignment will be evaluated based on the following key components:

1. A written research report.

2. Demonstrated capability to write and apply code for preprocessing the data, conducting explorative data analyses, and building a simple knowledge-based or content-based recommender system.

3. Quality and documentation of the provided code.

Please find detailed information about these components next.

3.1 Write a research report (50% of final grade)

Ensure that the research report is written in a clear and understandable manner. It is encouraged to include citations from relevant academic articles to support your analysis. Additionally, you may cite the book (Van Atteveldt, Trilling, & Calderon, 2022) to justify specific techniques or decisions made. The research report is structured into four sections:

3.1.1 Introduction

Approximatly 0.5 - 1 page

- Briefly explain why you chose the specific dataset.

- Highlight its relevance, potential insights, and why it is the most interesting for your group's objectives.

- State the overall aim of the Recommender System that you will develop in this assignment.

  - *For example, the Recommender System may focus on generating personalized book recommendations tailored to users' preferences, genres, and other users' behavior. Depending on the data you work with, this could be aimed at promoting reading habits or facilitating the sharing of relevant news within the community.*

3.1.2 Method section

Approximatly 2.5 - 4.5 pages

Data preprocessing

- Provide a detailed description of the final selection of variables, including their data types and any transformations applied.

- Explain the reasoning behind the approach taken for data preprocessing. Explain to your reader *why* these preprocessing steps are needed. You can think of the following reasons to apply preprocessing (see also the next section):

  - Visualizing the data.
  - Computing summary statistics.

- Explain clearly how the preprocessing steps are needed for the successful exectution of these techniques.

Exploratory Data Analysis (EDA)

- Describe the analytical strategy for EDA, including techniques such as visualization, summary statistics, or inductive methods like word clouds.

- Explain how these techniques will be used to gain insights into the dataset.

- Justify the selection of these techniques and their relevance to the dataset. For example:

  - Visualizing the data: This helps to identify outliers, trends, or patterns within the dataset.
  - Summary statistics: This helps in understanding the distribution, central tendencies, and variability of variables.
  - Inductive techniques like word clouds: Useful for textual data, so to uncover underlying themes and topics.

Recommender System development

- Specify the type of recommender system being developed and provide rationale for this choice.

- There are multiple decisions to consider at this stage, that you should elaborate on.

  - Specifically, explain whether you will build a knowledge-based or a content-based recommender system, and justify the choice of techniques used to create the recommendations. For example, you may choose to use cosine scores or soft-cosine scores.
  - Additionally, clarify which variables are included in the final recommendations and why they were selected.
  - Explain how the selected techniques will be used to build the recommender system.

- Justify the suitability of the chosen recommender system type for addressing the objectives of the assignment as described in the Introduction section.

3.1.3 Result section

Approximatly 2.5 - 4.5 pages

- Provide a comprehensive description of the dataset, including the number of observations and the types of variables present.

- Present the results of the Explorative Data Analysis (EDA) conducted on the dataset.

  - This may include insights gained from visualization, summary statistics, and/or inductive methods like word clouds. Describe patterns or trends that you have discovered during this analysis.

- Demonstrate the functionality of the developed recommender system.

- Explain how the system works and provide examples of recommendations generated for various types of input.

- This demonstration should showcase how the recommender system works in providing relevant recommendations based on user preferences or content characteristics.

3.1.4 Short conclusion

Approximatly 0.5 - 1 page

- Briefly summarize the key findings from the Exploratory Data Analysis (EDA).

- Reflect on the extent to which the original aim of the study specified in the Introduction was achieved: Was the recommendation system successful in meeting its objectives?

  - Please note that the aim is not necessarily to develop a perfect recommender system. You can provide critical insights here and suggest future strategies for enhancing its performance.

3.1.5 Quality of the writing, adherence to APA guidelines and references

Ensure the report is clearly written and adheres to APA guidelines. Including references to related academic work, if relevant, is appreciated.

3.1.6 Appendix: Division of labor

Approximatly 0.5 page

We believe it's essential for all team members to contribute equally to the group assignment. Therefore, in the Appendix, we request each team member to provide a brief description of their individual contributions to the group assignment. These contributions should be honestly reflected and cover the elements each team member worked on.

3.2 All code written in the project (35% of final grade)

3.2.1 Code for data preprocessing

- Code used to read in the data, explore the dataset, and inspect the relevant variables present.

- Your objective here is to select variables that might be of interest and usable later on.

- The code for cleaning and preprocessing your data is crucial here. Feature engineering plays a significant role in this step, considering the type of descriptive analysis you plan to conduct in the subsequent stages.

3.2.2 Code for Exploratory Data Analysis (EDA)

- Provide code to conduct exploratory data analysis (EDA) on the dataset.

- Ensure that the code yields a clear description of the data you will be analyzing, highlighting key variables such as their data types, number of unique observations, mean values, and distribution characteristics.

- Utilize data visualization techniques, such as plotting, to visually represent relationships and patterns within the dataset.

- Apply various analytical techniques discussed in the first weeks of the course, such as word clouds, to uncover insights and themes present in the dataset.

3.2.3 Code to create a recommender system

- Provide the code to build a simple knowledge-based or content-based recommender system.

- Specifically, you have to provide code that builds a recommender system, based on the insights from week 3. 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?

3.3 Quality of the code, and documentation (15% of final grade)

Make sure your code is well documented and understandable for people that see your code for the first time. This involves providing clear and concise comments within the code to explain its purpose and functionality. This requires providing clear and concise comments within the code to explain its purpose and functionality. Additionally, minimize code repetition and provide that is scalable if possible.

3.4 Handing in

One member of your group can hand in the group assignment until Monday, 6 May, 11:00 AM via Canvas. **Please compress all files into a single .zip or .tar.gz file**
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 <u>one</u> 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 written research report should be .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.*

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 (Isa) 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

Monday, 31–3: Lecture

During this first lecture, we will kickstart the course. We also take a first look into using computational methods for communication science by discussing what we can learn from analyzing texts.

Tuesday, 1–4. Lab session

✔ Read this manual and inspect the course Canvas page.
✔ Make sure your computer is ready to use for the course
✔ Read in advance: van Atteveldt and Peng (2018).
✔Read in advance: Boumans and Trilling (2016).

During the first tutorial meeting, we will introduce the course: We will explain the course goals and policies. We will take what we discussed in the lecture and use this to analyze text. We will start slow, and there will be time to go over some of the basic concepts as discussed in CCS-1. We will practice with some simple top-down and bottom-up algorithm for text analysis.

Week 2: Bottom-up approaches to text analysis

Monday, 7–4: Lecture

✔Read in advance chapter 10: Text as data Van Atteveldt et al. (2022).
✔Read in advance: Kroon, Welbers, Trilling, and van Atteveldt (2024). ✔Read from section "From BoW to Large Language Models: Revolutions in text representations" until "The rise of the transformer". (pages 147–149)

This week, we will discuss different ways to represent text data, starting with traditional methods like Bag of Words and TF-IDF before moving to word embeddings.

Tuesdag, 8–4: Lab session

During this lab session, you will complete the first set of MC-questions.

During this tutorial meeting, we will practice with the concepts and code as introduced in Monday's lecture. We will start practicing with transforming textual data to matrices, using `sklearn`'s vectorizers. 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 addition, you will form small groups of around 4 students for the group assignment. In case you understand all concepts, and finished the in-class assignments, you can start working on the group assignment. Specifically, we can start with exploring the dataset as provided, inspect relevant variables, and start working on cleaning the dataset and division of tasks.

Week 3: Recommender systems

Monday, 14–4: 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.

Tuesdag, 15–4: 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 4: Group presentation recommender system

Monday, 21–4: No Lecture (Easter Monday)

Tuesday, 22–4: Lab session

During this week's lab session, you will present your group assignment in class.

Week 5: Taking a break

Monday, 28–4: No Lecture – UvA teaching free week

Tuesday, 29–4: 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: Introduction to supervised machine learning)

Deadline group project: Tuesday 6–5

The deadline for handing in the group assignment is at the start of this week, **Tuesday 6–5 at 11:00AM**.

Monday, 5–5: No Lecture (Liberation Day)

Tuesday, 6–5: Lecture

✔ Read in advance: Vermeer, Araujo, Bernritter, and van Noort (2019)
✔ Read in advance Meppelink et al. (2021).

In this week, we will not have a lab session – but rather a lecture on Tuesday. We will discuss the introduction to supervised machine learning. We will discuss the basic principles of supervised machine learning using practical examples as well as example code. We will also discuss the role and applications of supervised machine learning in research and society.

Week 7: Validation in supervised machine learning

Monday, 12–5: Lecture

✔ Read in advance Birkenmaier, Lechner, and Wagner (2023).

In this lecture, we will take a deeper dive into supervised machine learning whereby we also discuss how to validate classifiers.

Tuesday, 13–5: lab session

Week 8: Coding in an academic context

Monday, 19–5: Lecture

✔ Read in advance Hube, Fetahu, and Gadiraju (2019).
✔ Read in advance Bender, Gebru, McMillan-Major, and Shmitchell (2021).

In the last lecture, we reflect on what we learned but mostly on how we can use this responsibly as scholars and researchers.

Tuesday, 20–5: Lab session

In the last lab session of the course, we will discuss scholars' use of code for research and the opportunities and responsibilities that this brings. It will also include a hands-on session on how to create responsible code.

Week 9: Final week

Monday, 26–5: No lecture

In this week, there will be no lecture so you can prepare for the exam.

Tuesday, 27–5: Exam

In the last lab session of the course, the exam will take place.

Thursday 5–6: Inspection exam

Tuesday 10–6: Resit

Thursday 19–6: Inspection resit exam

literature

Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? `[U+FFFD][U+FFFD]`. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency* (pp. 610–623). Association for Computing Machinery. Retrieved 2024-02-29, from `https://dl.acm.org/doi/10.1145/3442188.3445922` doi: 10.1145/3442188.3445922

Birkenmaier, L., Lechner, C. M., & Wagner, C. (2023). The search for solid ground in text as data: A systematic review of validation practices and practical recommendations for validation. , *0*(0), 1–29. Retrieved 2024-02-29, from `https://doi.org/10.1080/19312458.2023.2285765` (Publisher: Routledge _eprint: https://doi.org/10.1080/19312458.2023.2285765) doi: 10.1080/19312458.2023.2285765

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

Hube, C., Fetahu, B., & Gadiraju, U. (2019). Understanding and mitigating worker biases in the crowdsourced collection of subjective judgments. In *Proceedings of the 2019 CHI conference on human factors in computing systems* (pp. 1–12). ACM. Retrieved 2024-01-09, from `https://dl.acm.org/doi/10.1145/3290605.3300637` doi: 10.1145/3290605.3300637

Kroon, A., Welbers, K., Trilling, D., & van Atteveldt, W. (2024). Advancing automated content analysis for a new era of media effects research: The key role of transfer learning. *Communication Methods and Measures*, *18*(2), 142–162.

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. (2021). Reliable or not? an automated classification of webpages about early childhood vaccination using supervised machine learning. , *104*(6), 1460–1466. Retrieved 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., & 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. Retrieved from `https://doi.org/10.1080/19312458.2018.1458084` doi: 10.1080/19312458.2018.1458084

Van Atteveldt, W., Trilling, D., & Calderon, C. A. (2022). *Computational analysis of communication*. Wiley-Blackwell.

Vermeer, S. A. M., Araujo, T., Bernritter, S. F., & van Noort, G. (2019). Seeing the wood for the trees: How machine learning can help firms in identifying relevant electronic word-of-mouth in social media. , *36*(3), 492–508. Retrieved 2024-02-29, from `https://www.sciencedirect.com/science/article/pii/S0167811619300102` doi: 10.1016/j.ijresmar.2019.01.010