An Immersive Computational Text Analysis Course for Non-Computer Science Students at Barnard College

Adam Poliak **Jalisha Jenifer**
Barnard College, Columbia University
{apoliak,jjenifer}@barnard.edu

Abstract

We provide an overview of a new Computational Text Analysis course that will be taught at Barnard College over a six week period in May and June 2021. The course is targeted to non Computer Science at a U.S. Liberal Arts college that wish to incorporate fundamental Natural Language Processing tools in their research and studies. During the course, students will complete daily programming tutorials, read and review contemporary research papers, and propose and develop independent research projects.

1 Introduction

As computing has become foundational to 21st century literacy (Guzdial, 2019), students across many disciplines are increasingly interested in gaining computing literacy and skills. Barnard College is meeting this increased demand through its *Thinking Quantitatively and Empirically* and *Thinking Technologically and Digitally* requirements that respectively aim for students "to develop basic competence in the use of one or more mathematical, statistical, or deductive methods" and "foster students' abilities to use advanced technologies for creative productions, scholarly projects, scientific analysis or experimentation."[1]

This new 3-credit Computational Text Analysis course will enable roughly 25 students to leverage advanced NLP technologies in their work.[2] The goal of the course is to introduce students to the tools and quantitative methods to discover, measure, and infer phenomena from large amounts of text. Unlike a traditional Natural Language Processing class, students will not implement key algorithms from scratch. Rather, students will learn how to use existing python libraries like nltk (Loper and Bird, 2002), gensim (Rehurek and Sojka, 2011), spacy (Honnibal et al., 2020), and sklearn (Pedregosa et al., 2011) to analyze large amounts of textual data. Students will also gain famillarity with webscraping tools and APIs often used to collect textual data.

Week	Topic
1	Bash, Python, & Math bootcamp
2	Words, Words, Words
3	Topic Modeling
4	Data Collection
5	Machine Learning
6	Advanced Topics & Final Projects

Table 1: Theme for each of the six weeks.

2 Course Design

Due to COVID-19 schedule changes, the course will meet remotely for 95 minutes four days a week over 6 weeks. Each week will revolve around a class of methods used in computational text analysis, shown in Table 1. Course meetings will begin with a 30 minute lecture motivating and explaining a specific concept or method. Corresponding readings introducing these methods will be provided before class. In the subsequent 30 minute interval, we will collaboratively work through a Jupyter-Notebook demonstrating the day's concept using real-world textual data.[3] Both the lecture and collaborative demo will be recorded and available to students after class. Students can use the remaining 35 minutes to begin and complete daily homework tutorials where they will further experiment with applying methods to provided real-world text.

[1] http://catalog.barnard.edu/barnard-college/curriculum/requirements-liberal-arts-degree/foundations/

[2] https://coms2710.barnard.edu/

[3] We will continue this practice from a previous online Introductory Data Science class (https://coms1016.barnard.edu/) where students found live coding demos to be helpful.

Assignment	Grade Percent
Daily Tutorials	20
Weekly Homework	30
Paper Reviews	15
Final Project	35

Table 2: Assignments and corresponding grade weights.

2.1 Assignments

Students will complete four types of assignments during the six week course to gain familiarity, comfort, and confidence applying computational textual analysis methods to large corpora (Table 2). With an emphasis on learning by doing, completing **daily tutorials** independently will help students solidify their understanding of the day's material while working through examples where these methods have been successfully deployed. Daily tutorials will primarily be graded using an autograder. Many of the daily tutorials are adaptions of notebooks from similar courses, e.g. Jonathan Reeve's *Computational Literary Analysis* course[4] at Columbia University, Matthew Wilkens's and David Mimno's *Text Mining for History and Literature* course[5] at Cornell, and Christopher Hench's and Claudia von Vacano's *Rediscovering Text as Data* course[6] at Berkeley.

Each week, students will be given a corpus and a research question and will be tasked with using the previous week's methods to try answering the specific research question. These **weekly homeworks** will give students freedom to experiment with different methods and see how different design choices can have significant impacts. Table 3 outlines the theme and corpora for each weekly homework. Students will be allowed to work on these assignments in pairs. Completed notebooks containing students' code, figures, and a brief write-up will be graded manually. To help prepare students for final projects, feedback will include questions and comments related to both the technical methods used and presentation.[7]

[4]https://icla2020b.jonreeve.com/

[5]https://github.com/wilkens-teaching/info3350-f20

[6]https://github.com/henchc/Rediscovering-Text-as-Data

[7]Daily tutorials and weekly homeworks will be publicly available at https://github.com/BC-COMS-2710/summer21-material. Solutions and autograders will be available to other instructors wishing to adopt this course.

Theme	Corpus
Readability	U.S Presidential Inaugural Addresses
Document Similarity	NYTimes Obituaries
Web Scraping	Columbia Course Reviews
Machine Learning	Political Tweets

Table 3: Themes and corresponding corpus for the four week-long homeworks different assignments.

Students will complete five brief **paper reviews** during the course. Each week, students will choose from a set of research papers and write a brief review of the paper. Each review will include a short summary and a few sentences reflecting on their opinion on the paper's methods and findings. This will help students appreciate the wide-ranging applicability of these methods and see more examples where researchers in other fields have successfully leveraged computational text analysis methods.

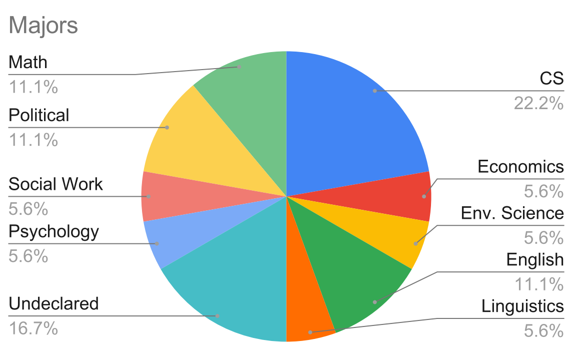
For their **final projects**, students will develop their own research question based on a corpus that they will collect. Students will apply at least two methods covered in the course to answer their research question. By the end of the fourth week, students will propose their research question and will begin to collect a corpus to study. Students will present a brief progress update to the rest of the class during the last scheduled course meeting (Monday June 14^{th}) and provide feedback to each other. Students will use the official finals period to incorporate feedback and submit a final write up describing their research.

2.2 Computational Infrastructure

All assignments will be completed using JupyterLabs hosted on a JupyterHub server maintained by Columbia University IT at no cost to students. This enables equal access for students who might not have resources to store and process large amounts of text. Additionally, this will ensure students use the same library and tools versions.

2.3 Prerequisites

We strongly recommend students to have taken at least one prior programming course. While we make no assumptions about students prior programming experience and we cover bash and python basics, we believe having one prior programming course will help students hit the ground running. Furthermore, this allows us to cover more advanced topics in depth. In future versions of the course taking place during a traditional 13-week semester, we



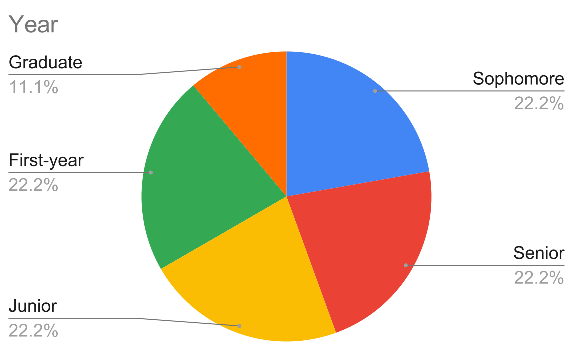


Figure 1: Statistics from 18 registered students who completed a pre-course interest form. Breakdown of students by major (left) and year (right).

will consider removing this recommendation.

2.4 Textbooks

We will primarily use the recent *Text Analysis in Python for Social Scientists: Discovery and Exploration* textbook (Hovy, 2021) because of its assertion that "using off-the-shelf libraries . . . strips away some of the complexity of the task, which makes it a lot more approachable" thus enabling "programming novices to use efficient code to try out a research idea." We will use select readings from the NLTK book (Bird et al., 2009), the most recent edition of *Speech and Language Processing* (Jurafsky and Martin, 2000),[8], An Introduction to Text Mining: Research Design, Data Collection, and Analysis (Ignatow and Mihalcea, 2017), and Melanie Walsh's online *Cultural Analytics & Python* textbook.[9]

3 Prospective Students

With summer course registration underway, 25 students are currently registered for the course. Since the course is under active development and students come from a range of academic backgrounds, we have asked students to complete a brief survey to help us prepare and tailor the class as much as possible. Figure 1 shows results from the 12 students who have completed the survey so far. While we see interest from Computer Science students, most of the students are programming novices – half of the survey participants indicated they have no prior programming experience outside of one or two programming courses.[10] Many students indicated they have corpora in mind that they are interested in studying or that they plan on using these methods to enhance their undergraduate theses in other fields.

4 Measuring STEM-Related Attitudes & Beliefs

The beliefs and attitudes students possess about learning play a significant role in their academic performance (Elliot and Dweck, 2013). Given the fact that our course is targeting non-computer science majors, we are interested in seeing how the academic attitudes held by participating students relate to and change throughout their participation in the course. We are particularly interested in measuring students' beliefs about the malleability of their own intelligence, as these beliefs have been shown to predict student motivation, engagement, and performance in academic courses (De Castella and Byrne, 2015). To measure the effect such beliefs on students in this course, we will administer the self-theory implicit theories of intelligence scale (De Castella and Byrne, 2015). We will also measure students' motivation for computer science using an adapted version of the Science Motivation Questionnaire (Glynn et al., 2007). We will have students complete these surveys at the beginning and end of the course, which will enable us to a) measure whether students' initial attitudes significantly relate to their performance in the course, and b) measure whether participation in the course leads to changes in student attitudes over time.

5 Conclusion

We described a new Computational Text Analysis course that will be taught at Barnard College in

[8]Specifically Chapters 4 and 5 for Machine Learning.

[9]`https://melaniewalsh.github.io/Intro-Cultural-Analytics/welcome.html`

[10]The larger number of interested computer science students might be due to the course being offered by the Computer Science department.

May-June 2021. The broad goal of the course is for students to gain technical skills allowing them to successfully incorporate Natural Language Processing methods into their respective fields and research. We will rely on proven methods to measure the effect the course has on students' perception of their abilities and skills. All material developed for the course (slides, assignments, autograders) will be available for instructors to adopt at their institutions. An updated version of this paper will be available on the first author's webpage with reflections and lessons learned once the course is completed.

Acknowledgements

We thank Maya Bickel for providing feedback and debugging assignments. Additionally, we thank Michael Weisner and his team at Columbia University Information Technology for helping maintain the computational infrastructure for the course.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Krista De Castella and Donald Byrne. 2015. My intelligence may be more malleable than yours: The revised implicit theories of intelligence (self-theory) scale is a better predictor of achievement, motivation, and student disengagement. *European Journal of Psychology of Education*, 30(3):245–267.

Andrew J Elliot and Carol S Dweck. 2013. *Handbook of competence and motivation*. Guilford Publications.

Shawn M Glynn, Gita Taasoobshirazi, and Peggy Brickman. 2007. Nonscience majors learning science: A theoretical model of motivation. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching*, 44(8):1088–1107.

Mark Guzdial. 2019. Computing education as a foundation for 21st century literacy. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 502–503.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Dirk Hovy. 2021. *Text Analysis in Python for Social Scientists: Discovery and Exploration*. Elements in Quantitative and Computational Methods for the Social Sciences. Cambridge University Press.

Gabe Ignatow and Rada Mihalcea. 2017. *An introduction to text mining: Research design, data collection, and analysis*. Sage Publications.

Daniel Jurafsky and James H Martin. 2000. Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).