Course Description

Social actors interact using language. As a result, testing social science theories usually requires analyzing, in one way or another, written language. Thankfully, recent advances in computational linguistics have considerably increased the reach of scholars interested in working with textual data. Moreover, swathes of digitized documents have been made available to researchers in recent years. This includes parliamentary records, committee proceedings, bills, laws, international treaties, news reports, social media discussions, blogs, websites, and so forth. How to process and analyze such large quantities of textual data meaningfully is the central focus of this course.

The course introduces students to the state of the art in the field of computer-assisted textual analysis. It covers the most widely used methods for the empirical analysis of textual data, from the preprocessing stages to the interpretation of findings. The course also includes an introduction to machine learning. By the end of this course, students will have gained expertise with an important branch of computational social science. They will also have developed skills with the Python programming language.

Course Format

The course takes place in the Ramsay Wright computer lab. Classes will be a combination of advanced lectures and interactive exercises, every Tuesdays. Registered students will also be invited to present an independent research project during the last weeks of the course.
Software

The course takes place in a computer lab. Class exercises and demonstrations will be performed mostly using the Python programming language. Python is the most popular programming language in the world, and provides an extensive collection of tools that facilitate the analysis of textual data. The course includes an introduction to the language.

Class examples will rely upon Python 3. Computers from the lab should also have the language installed. In-class examples will be provided from the Jupyter notebook, a user-friendly environment for interactive computing.

Python is freely available on all operating systems. It should be installed by default on Mac computers. Students can easily reproduce exercises and replicate examples on their personal computers, but those who do not dispose of a personal computer may practice in one of several computer labs on the campus.

Requirements

This course may be of interest to graduate students using either qualitative or quantitative methods (or both). Although there are no formal requirements for the course, it will involve some advanced concepts in programming and statistics. Some background in statistical analysis and/or computing would be useful. However, the pedagogical approach is tailored to students who may not have had an extended training in mathematics or computing as undergraduate students (as is often the case in the social sciences).

Marking Scheme

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Percentage</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written Assignment #1</td>
<td>20%</td>
<td>October 16, 2018</td>
</tr>
<tr>
<td>Written Assignment #2</td>
<td>20%</td>
<td>November 13, 2018</td>
</tr>
<tr>
<td>Oral Presentation</td>
<td>15%</td>
<td>During the last two or three weeks (depending on enrollment)</td>
</tr>
<tr>
<td>Term Paper</td>
<td>35%</td>
<td>December 7, 2018</td>
</tr>
<tr>
<td>Participation</td>
<td>10%</td>
<td></td>
</tr>
</tbody>
</table>

Readings

No textbook is perfectly tailored to the needs of this course. Instead, we will focus on a collection of chapters from the following set of seminal texts in the field. Together they will cover most of the materials under study. The readings recommended for each class can be very helpful to supplement the lecture notes that will be made available to students. All of these books are accessible for free online, either from the authors’ websites or electronically through the UofT Library.

  - An accessible introduction to natural language processing in Python. The book is available online for free.

  - A key reference that covers most of the topics discussed in this course, and more. Online versions are available.
  ◦ Another useful reference for exploring some of the topics in more depth. Some *chapters* are available online for free.

  ◦ An older reference that nonetheless covers key basic concepts for this course. The book is available electronically through the UofT Library.

  ◦ A useful reference on the particular topic of machine learning. The book is available electronically through the UofT Library.

**Evaluations**

The course uses a variety of evaluation formats to help students develop different skills related to scientific research.

**Written Assignments**

The written assignments are problem sets designed to evaluate students’ ability to put the methods learned into practice. They may involve practicing various types of textual analysis using Python, reviewing a paper in the field, and answering short factual questions about the models and their interpretation.

There is no better way to improve one’s skills than practice. Therefore, those exercises are not only useful as evaluations, but also as a way for students to gain concrete expertise with the subject-matter. Assignments are done individually. They are handed in during class at the due date, or else submitted directly by email to the instructor.

**Oral Presentation**

During the last weeks of the course, registered students will take turns to present the research project they are working on for the term paper. The presentations are between 5 and 10 minutes, followed by reactions from the audience.

At the time of the presentation, the research project will likely not be completed. Students will not be evaluated based on the results that they have obtained. Instead, the goal of the presentation is to evaluate whether students are able to invoke the concepts and methods studied during the course clearly and efficiently.

After each presentation, the rest of the class will be invited to formulate constructive comments, which may help the presenter to complete the term paper.

**Term Paper**

The term work takes the form of a scientific report in which students propose an application using any of the models for textual analysis discussed during the course. This represents the empirical section of a research paper on a topic of the graduate student’s choosing. Students can use one of
the corpora examined in class or use their own data sources. Graduate students may opt to work on a draft of a dissertation chapter.

The term paper will include a brief introduction stating the research question, an outline of the theory and some testable propositions (hypotheses). The main part of the term paper (roughly 4,000 words), however, consists of presenting an analysis involving textual data. The paper is expected to introduce the empirical research design and proceed with the key stages of the empirical analysis. Students should make sure to provide the replication scripts along with their study.

Class Schedule: Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 11</td>
<td>Computers and Language</td>
<td></td>
</tr>
<tr>
<td>September 18</td>
<td>Introduction to Python</td>
<td></td>
</tr>
<tr>
<td>September 25</td>
<td>Statistics for Textual Data I</td>
<td></td>
</tr>
<tr>
<td>October 2</td>
<td>Statistics for Textual Data II</td>
<td></td>
</tr>
<tr>
<td>October 9</td>
<td>Natural Language Processing</td>
<td></td>
</tr>
<tr>
<td>October 16</td>
<td>Lexicons and Vector Space Models</td>
<td>Assignment Due</td>
</tr>
<tr>
<td>October 23</td>
<td>Introduction to Machine Learning</td>
<td></td>
</tr>
<tr>
<td>October 30</td>
<td>Supervised Learning I</td>
<td></td>
</tr>
<tr>
<td>November 13</td>
<td>Supervised Learning II</td>
<td>Assignment Due</td>
</tr>
<tr>
<td>November 20</td>
<td>Unsupervised Learning I</td>
<td>(Student Presentations)</td>
</tr>
<tr>
<td>November 27</td>
<td>Unsupervised Learning II</td>
<td>(Student Presentations)</td>
</tr>
<tr>
<td>December 4</td>
<td>Advanced Topics</td>
<td>(Student Presentations)</td>
</tr>
<tr>
<td>December 7</td>
<td>[End of Semester]</td>
<td>Term Paper Due</td>
</tr>
</tbody>
</table>

Note: Topics by date are for information only. The schedule above (and the detailed structure in the following pages) may be adjusted during the term due to unforeseen circumstances or to improve the pedagogical benefits to students.
Class Schedule: Detailed

Topic 1: Computers and Text

September 11: Computers and Language
1. Brief history of computational text analysis.
2. Examples of recent applications.
3. How computers encode text.
4. Working with foreign languages.
5. Some fundamental concepts in natural language processing.
6. Introduction to Python.

September 18: Introduction to Python
1. Introduction to Python (continued).
2. Data types, lists and dictionaries.
3. Input/Output.
4. Functions and conditional statements.
5. Encoding text.
6. Processing textual data in Python.
7. Exercise: Parsing html and xml data.

Readings
• Bird, Klein, and Loper (2009), Ch. 2–4.
• Manning and Schütze (1999), Ch. 1.

Other Useful References
• Aggarwal and Zhai (2012b).
• McKinney (2013), Ch. 1.
• Downey, Elkner, and Meyers (2002), Ch. 1–2.
• D’Orazio et al. (2014).
• Jockers (2014).
• Weiss, Indurkhya, and Zhang (2015).
• Krippendorff (2013), Ch. 4.
• Watch a 45-minute introductory video on Python.
Topic 2: Statistics for Textual Data

September 25: Statistics for Textual Data I

1. Document retrieval and indexing.
2. Tokenization, sentence splitting.
3. Word counts and word distributions.
4. Vectorization.
5. Visualization techniques.

October 2: Statistics for Textual Data II

1. Term-frequency/inverse document frequency (tf–idf) weighting.
2. Word co-occurrences.
3. Comparing texts.
4. Statistical properties of texts.
5. Examples of applications: Wordscores and Wordfish.

Readings

• Manning, Raghavan, and Schütze (2009), Ch. 1–2.
• Manning and Schütze (1999), Ch. 5–6.

Other Useful References

• Bird, Klein, and Loper (2009), Ch. 2–4.
• Jiang (2012).
• Nenkova and McKeown (2012).
• Zipf (1932).
• Porter (1980).
• Python Online Documentation.

Examples of Applications

• Laver and Garry (2000).
• Laver, Benoit, and Garry (2003).
• Alfesi and Chambers (2007).
• Lowe (2008).
• Slapin and Proksch (2008).
• Monroe, Colaresi, and Quinn (2008).
• Gentzkow and Shapiro (2010).
• Proksch and Slapin (2010).
• Black et al. (2011).
• Däubler et al. (2012).
• Acton and Potts (2014).
• Yu (2014).
• Spirling (2016).
• Blaxill and Beelen (2016).
5. October 9: Introduction to Natural Language Processing

1. Overview of linguistic theory.
2. Unigrams, bi-grams and n-grams.
3. Part-of-speech tagging.
4. Stemming and lemmatization.
5. Grammar parsing.

6. October 16: Lexicons and Vector Space Models

1. Creating and using word lexicons (dictionaries).
2. Summarizing text properties.
4. Word embeddings.
5. Word similarities and word relations.

Readings

- Bird, Klein, and Loper (2009), Ch. 5.
- Manning and Schütze (1999), Ch. 3, 10.
- Turney and Pantel (2010).

Other Useful References

- Manning, Raghavan, and Schütze (2009), Ch. 6.
- Jurafsky and Martin (2008), Ch. 9–10.
- Miller et al. (1990).
- Mikolov et al. (2013).
- Manning et al. (2014).
- Python Online Documentation.

Examples of Applications

- Tausczik and Pennebaker (2010).
- Bollen, Mao, and Zeng (2011).
- Golder and Macy (2011).
- Michel et al. (2011).
- Young and Soroka (2012).
- Jensen et al. (2012).
- Coviello et al. (2014).
- Gentzkow, Shapiro, and Taddy (2016).
- Rheault et al. (2016).
- Vosoughi et al. (2018).
Topic 4: Machine Learning

October 23: Introduction to Machine Learning
2. Annotating texts and intercoder reliability.
3. Development, training and testing.
4. An introductory example: sentiment analysis.

October 30: Supervised Learning I
1. Features and classes.
2. "Bag of words" approach.
3. Feature selection.
5. Nearest Neighbor classifiers.
6. Multi-class problems.

November 13: Supervised Learning II
1. Evaluating classifiers.
2. Accuracy measures.
3. Ridge regression.
5. Applications in Python.

November 20: Unsupervised Learning I
1. Unsupervised learning.
3. Clustering analysis.
4. Principal component analysis.

November 27: Unsupervised Learning II
1. Latent Dirichlet Allocation (LDA).
2. Correlated and dynamic topic models.
3. Non-Negative Matrix Factorization.
4. Student presentations.

Readings
• Hastie, Tibshirani, and Friedman (2009), Ch. 2, 6–7, 12.
• Bird, Klein, and Loper (2009), Ch. 6.
• Steyvers and Griffiths (2011).
Other Useful References

- Manning, Raghavan, and Schütze (2009), Ch. 15.
- Blei, Ng, and Jordan (2003).
- Blei and Lafferty (2006a).
- Blei and Lafferty (2006b).
- Blei (2012).
- He and Garcia (2009).
- Aggarwal and Zhai (2012a).
- Richert and Coelho (2013).
- Lantz (2013).
- James et al. (2013).
- scikit-learn for Python: Online Documentation.

Examples of Applications

- Mosteller and Wallace (1964).
- Yu, Kaufmann, and Diermeier (2008).
- Hopkins and King (2010).
- Grimmer (2010).
- Diermeier et al. (2012).
- Hirst et al. (2014).
- Roberts et al. (2014).
- D’Orazio et al. (2014).
- Harris (2015).
- Reich et al. (2015).
- Roberts, Stewart, and Airoldi (2016).
- Tingley (2017).
Topic 5: Advanced Topics and Wrap-Up

December 4: Overview of Advanced Topics (As Time Permits)

1. Regular expressions.
2. Web-scraping and online text data retrieval.
4. Student presentations (continued).

Readings

- Hastie, Tibshirani, and Friedman (2009), Ch. 11.
- Aggarwal (2012).

Other Useful References

- Munzert et al. (2015).
- Beautiful Soup for Python: Online Documentation.
References


He, Haibo, and Edwardo A. Garcia. 2009. “Learning from Imbalanced Data.” *IEEE Transactions on Knowledge and Data Engineering* 21(9): 1263–1284.


James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. An Introduction to Statistical Learning with Applications in R. New York: Springer.


