DATA.STAT.840 Statistical Methods for Text Data Analysis
Exercises for Lecture 11: Visualization and feature extraction

Exercise 11.1: Principal component analysis of four newsgroups. Using the same data as in exercise 6.1 (four newsgroups), use principal component analysis to reduce the TF-IDF vectors of the documents (as computed in exercise 6.1) to two dimensions.
Plot the resulting two-dimensional vectors; plot the documents of the different four newsgroups with different colors. Discuss the results: does the PCA plot tell something about the distribution of the document features?
You can use the Python implementation discussed on the lecture (lecture 11, slide 4), or another language/library of your choice. Report your code, the resulting plot, and your discussion.

Exercise 11.2: t-SNE plot of four newsgroups. Using the same data as in exercise 6.1 (four newsgroups), use the t-distributed stochastic neighbor embedding (t-SNE) method to reduce the TF-IDF vectors of the documents (as computed in exercise 6.1) to two dimensions.
Plot the resulting two-dimensional vectors; plot the documents of the different four newsgroups with different colors. Discuss the results: does the t-SNE plot tell something about the distribution of the document features?
You can use the Python implementation discussed on the lecture (lecture 11, slide 7), or another language/library of your choice.

- Hint 1: If the t-SNE computation takes too long on your computer, it is ok to use a subset of the documents, e.g. 1000 or 500 samples.
- Hint 2: If the plot looks like a central mass of points and isolated groups of points very far from the center, the far-away groups may be outliers: in that case, you can set the axis limits of your plot to focus on the central group. In Python, if you created a myaxis object using `matplotlib.pyplot.subplots()`, then you can set the axis limits by `myaxes.axis([-50, 50, -50, 50])` where the first two numbers are the horizontal minimum and maximum and the last two are the vertical minimum and maximum.

Report your code, the resulting plot, and your discussion.

Exercise 11.3: Word embedding. In this exercise we use the same data as in exercise 7.4 (provided in "hmm_sentences.txt" for that exercise, or in "hmm_sentences.zip" on the Moodle page as separate files for each sentence). Note: you do not need to prune the vocabulary of these texts since it is so small already.
Using the gensim library as in lecture 11 slide 24, or using another language/library of your choice, create 5-dimensional word2vec embeddings for the data. Use a window size of 3 words.
Report the resulting embeddings (vectors) for the words "where", "dog", and "explain".

DATA.STAT.840 Statistical Methods for Text Data Analysis

Exercises for Lecture 12: Paragraph embedding

Exercise 12.1: Paragraph embedding of four newsgroups.
Using the same data as in exercise 6.1 (four newsgroups), use paragraph embedding to create a vector for each document, using the Python implementation discussed on the lecture (lecture 12, slides 12-13), or using a different language/library of your choice.
Then find, by Euclidean distance, the closest other documents for the following documents: 101551 (part of rec.autos), 103118 (part of rec.motorcycles), 98657 (part of rec.sport.baseball) and 52550 (part of rec.sport.hocket). Are the closest documents from the same newsgroup? Does their content seem to match that of the original document?
Report your code, results, and discussion.

DATA.STAT.840 Statistical Methods for Text Data Analysis

Exercises for Lecture 13: LSTM models

Exercise 13.1: LSTM model of "Pride and Prejudice".

In this exercise we use the same Project Gutenberg ebook of Jane Austen's "Pride and Prejudice" as in exercise 3.1. Process the ebook text as in that exercise, except as described below.

Try to build a LSTM model to predict the next word: that is, the context window should be the 10 previous words, and the target word to predict should be the next word. Use the Python implementation discussed on lecture 13 (slide 25 onwards), or another language/library of your choice.

a) Use the lemmatized and vocabulary-pruned text to learn the LSTM model. Then use the model to generate 100 words of new text, starting from "enough to leave her room elizabeth was glad to be" (if any of these words are not in your pruned vocabulary, leave them out and add enough words in their place to fill a window of 10 words). Discuss the result.

b) Use the texts without lemmatization and vocabulary pruning to learn the LSTM model. Then use the model to generate 100 words of new text, starting from "enough to leave her room elizabeth was glad to be". Discuss the result.

Report your code, generated text, and discussion.

NEXT