Business Analytics

Lecture 5: Topic Models

Ulrich Wohak[1]

[1]Department of Economics
Vienna University of Economics and Business

Introduction

- So far we have considered words and (transformation of) document vectors encoding those words
- Unfortunately, this is often not enough: polysemy and synonymy could case problems for us
- What we would really like the computer to understand is the *meaning* of a partciular word
- How can we think about this issue? Firth (1975) famously said:

  You shall know a word by the company it keeps.

From words to meaning

- Why do we bother with this? Well, it allows us to
  1. Compare texts on the basis of *meaning* (not keywords)
  2. Search based on *meaning*
  3. Represent the subject of a statement/document or corpus
  4. Extract keywords belonging to a particular *meaning*
- To achieve this, we need to infer what w_i *means*
- In fact, we need to infer what all w_i *mean*
- We need a different kind of vector for this!
  a. **word-topic vector:** a single word w_i is represented by a (topic) vector
  b. **document-topic vector:** a single vector represents each document (based on a combination of the corresponding word vectors)
- **Good news:** We are already equipped with most of the things we need to do this.

LSA: Pipeline

- We know (and have at our dispoal) a corpus of text
- We know how to preprocess our corpus
- and we know how to tf-idf transform our numeric representations
- We will stack our document vectors together into a $D \times V$ *Document-Term Matrix* $\mathbf{A}$
- Each row of $\mathbf{A}$ corresponds to a *tf-idf* representation of document $d \in D$
- Recall that this is a *sparse* matrix (i.e. lots of zeros)
- Our focus will be a method called Latent Semantic Indexing/Latent Semantic Analysis (**LSI/LSA**)
- It takes as an input a tf-idf transformed document-term matrix and applies a dimensionality-reduction operation (PCA) on it

LSA: Overview

- We know (and have at our dispoal) a corpus of text
- We know how to preprocess our corpus
- and we know how to tf-idf transform our numeric representations
- We will stack our document vectors together into a $D \times V$ *Document-Term Matrix* $\mathbf{A}$
- Each row of $\mathbf{A}$ corresponds to a *tf-idf* representation of document $d \in D$
- Recall that this is a *sparse* matrix (i.e. lots of zeros)
- Our focus will be a method called Latent Semantic Indexing/Latent Semantic Analysis (**LSI/LSA**)
- It takes as an input a tf-idf transformed document-term matrix and applies a dimensionality-reduction operation (SVD) on it
- We are moving from a vector space to a topic space (and infer meaning as we do so)!

LSA: Intuition

- SVD finds co-occurring words by calculating the correlation between the terms of the term-document matrix
- SVD simultaneously finds the correlation of term use between documents and the correlation of documents with each other
- With these two pieces of information SVD computes the linear combinations of terms that have the greatest variation across the corpus

$\Rightarrow$ **These linear combinations of term frequencies will become topics**

LSA: Considerations

- The machine does not understand what the combinations of words mean —just that they belong together
- *dog*, *cat*, and *love* appear together a lot $\rightarrow$ same topic
- No idea this is (might be!) topic *pets*
- It can include (near-)synonyms, but also antonyms
- A human has to look at the words with a high associated weight to label the topic

LSA: (Quick) Math

- We use our $D \times V$(tf-idf transformed) document-term matrix A and use SVD to decompose it into three matrices

$$A_{D\times V} = U_{D\times p} S_{p\times p} H^T_{p\times V} \tag{1}$$

where

$U_{D\times p}$ is the **term-topic** matrix (correlation between topics and words)

$S_{p\times p}$ is the **sigma matrix** (diagonal matrix telling us how much information is captured by each topic)

$H^T_{p\times V}$ is the **document-document** matrix (measures how often documents use the same topics in the new model)

Let's code!