ISA 414 – Managing Big Data

Lecture 16 – Text Mining (Part III)

Topic Modeling

Dr. Arthur Carvalho

arthur.carvalho@miamioh.edu

MIAMI UNIVERSITY

Announcements

- Midterm project
  - To be released on Tuesday, October 19th
  - Deadline: 11:59 pm on October 22nd
  - Individual or in groups of two
    - Send me an email in case you already have a defined group
- No office hours
  - Virtual office hours on Friday from 10:30 to 11:30
  - Email me for a WebEx link

Lecture Objectives

- Quick review of Homework 7

- Learn how to automatically extract topics from textual data

Lecture Instructions

- Download the notebook "*Lecture 16.ipynb*" available on Canvas
  - Open the above file with VS Code

- Download the file "*abcnews-date-text.csv*" to the same folder our notebook is

Text Mining

- Text mining (analytics): deriving information from textual data
  - Tasks
    - Text categorization (Lecture 15)
    - Sentiment analysis (Lecture 15)
    - Text clustering (*e.g.*, plagiarism checker)
    - Automatic summarization (*e.g.*, “consensus” of several reviews)
    - Topic modeling (today)

Text Mining

- Running example: can we automatically find the topic of news headlines?
  - Data: over 1 million news headlines
    - Source: www.kaggle.com/therohk/million-headlines
    - Two columns: published_date and headline_text
    - We only focus on the latter
  - Traditional problems faced in ISA 414 projects:
    - Which topics are users discussing when mentioning company/product X on social media?

Topic Modeling

- The mathematics behind the techniques we learn today are beyond the scope of this course
  - Traditionally covered in PhD courses
- These techniques can be seen as **unsupervised learning** methods
  - We **do not** have to define a target variable
    - We “cluster” the documents into groups
  - If we had a target, we could train models to classify (find the topics) of future texts
    - See our previous class

Topic Modeling

- Traditional topic modeling techniques
  - Latent Semantic Analysis (LSA or LSI)
    - Technically, it is a dimensionality reduction model
      - Main idea: words occur in similar pieces of text if they have similar meaning
  - Latent Dirichlet Allocation (LDA)
    - Probabilistic model for identifying abstract topics from a corpus
      - Main idea: each document has a mix of topics, and every topic has a mix of words
    - Our focus in this course
    - For the brave and bold:
      - Search for the paper “Latent Dirichlet Allocation” published in the Journal of Machine Learning Research in 2003

LDA

- LDA is based upon two general assumptions:
  - Documents that have similar words usually have the same topic
  - Documents that have groups of words frequently occurring together usually have the same topic

- Mathematically, the above two assumptions can be represented as:
  - Documents are probability distributions over latent topics
  - Topics are probability distributions over words

LDA

- Think about LDA as an algorithm that learns two “layers” from a corpus of documents
  - The first layer is the distribution of categories/topics
    - For example, financial, weather, and political news
  - The second layer is distribution of words within each topic
    - For example, “sunny” and “cloud” are often in “weather” news while “money” and “stock” are in “financial” news

Finance	Weather	Arts	Sport
Money	Sunny	Music	World Cup
Stock	Cloud	Piano	Soccer
Trade	Humanity	Calligraphy	Tennis
Market	Temperature	Photography	Sailing

LDA

- Important points
  - Some words (*e.g.*, “a”, “with”, “can”) do not contribute to topic modeling
    - These words exist in almost all documents
    - Consequently, they will likely be in all categories with the same probability
    - Therefore, data preprocessing is crucial
      - For example, stop words removal
  - LDA works with counts of words
    - That is, with a DTM having absolute TF values
    - So far, we have only learned how to create DTMs that have TFIDF and normalized TF scores

Topic Modeling

- Let's return to our example
  - Can we automatically find the topic of news headlines?
- Let's start by loading the data
  - There are over 1.2 million data points
    - It would take a very long time to work with that much data
    - Let's work with a sample of 10,000 headlines only

```
import pandas
raw_data = pandas.read_csv("abcnews-date-text.csv")
corpus = raw_data["headline_text"]
corpus = corpus[0:10000]
```

Topic Modeling

- Next, let's create a DTM
  - Remember that LDA works with absolute frequencies
  - We use the CountVectorizer in sklearn instead of TfidfVectorizer we used in previous classes

```
from sklearn.feature_extraction.text import CountVectorizer
count_vect = CountVectorizer(stop_words='english')
dtm = count_vect.fit_transform(corpus)
```

  - There is a lot more we could do when creating the DTM
    - *E.g.*, put an upper (max_df) and lower (min_df) threshold on the proportion of documents having a term
      - See http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

Topic Modeling

- Finally, let's run LDA on the DTM
  - More important points
    - One **must** define (guess) the number of topics when running LDA
      - It is common practice to run LDA with different numbers
    - One may also define (guess) initial probabilities of topics to help the algorithm
      - See https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html
  - LDA with 5 topics

```
from sklearn.decomposition import LatentDirichletAllocation
LDA = LatentDirichletAllocation(n_components=5)
LDA.fit(dtm)
```

Topic Modeling

- LDA: understanding the results
  - We now have 5 topics, each one defined by words

```
first_topic  = LDA.components_[0]
second_topic = LDA.components_[1]
third_topic  = LDA.components_[2]
fourth_topic = LDA.components_[3]
fifth_topic  = LDA.components_[4]
```

  - How popular is each word inside a topic?
    - Look inside any of the above lists
    - You will see “scores” for each index column of our DTM

Topic Modeling

- LDA: understanding the results
  - Let's retrieve the top 10 words in each topic
    - 2-step process in Python:
      - Step 1: sort the indexes of the words in each topic by score and retrieve the top 10

```
top10_index_1st_topic = first_topic.argsort()[-10:]
top10_index_2nd_topic = second_topic.argsort()[-10:]
top10_index_3rd_topic = third_topic.argsort()[-10:]
top10_index_4th_topic = fourth_topic.argsort()[-10:]
top10_index_5th_topic = fifth_topic.argsort()[-10:]
```

Topic Modeling

- LDA: understanding the results
  - Let's retrieve the top 10 words in each topic
    - 2-step process in Python:
      - Step 2: retrieve the top 10 words based on their indexes

```
all_words = count_vect.get_feature_names()

top10_words_1st_topic = [all_words[i] for i in top10_index_1st_topic]
top10_words_2nd_topic = [all_words[i] for i in top10_index_2nd_topic]
top10_words_3rd_topic = [all_words[i] for i in top10_index_3rd_topic]
top10_words_4th_topic = [all_words[i] for i in top10_index_4th_topic]
top10_words_5th_topic = [all_words[i] for i in top10_index_5th_topic]
```

Topic Modeling

- LDA: understanding the results
  - Let's look at other metrics to understand the composition of topics
    - Install the pyLDAvis module

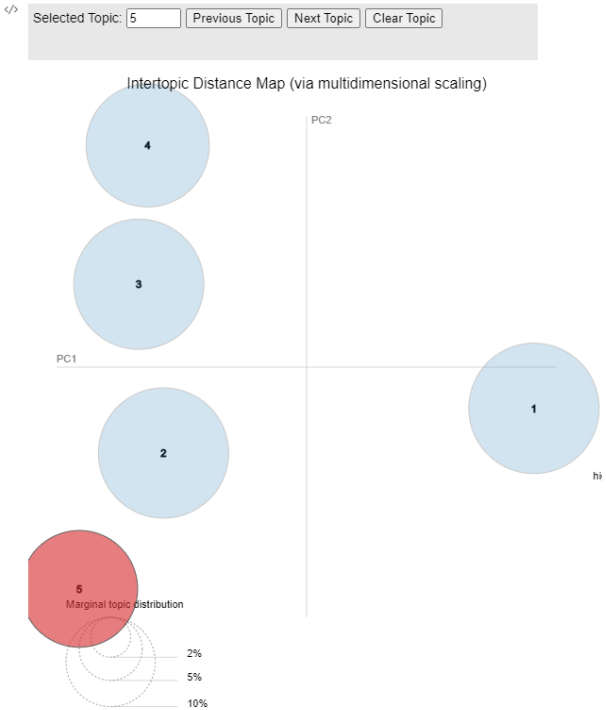
      ```
      pip install pyldavis
      ```

    - Run the LDA visualization module for the top 10 words

      ```
      import pyLDAvis
      import pyLDAvis.sklearn
      pyLDAvis.enable_notebook()
      pyLDAvis.sklearn.prepare(LDA, dtm, count_vect, R = 10)
      ```

Topic Modeling

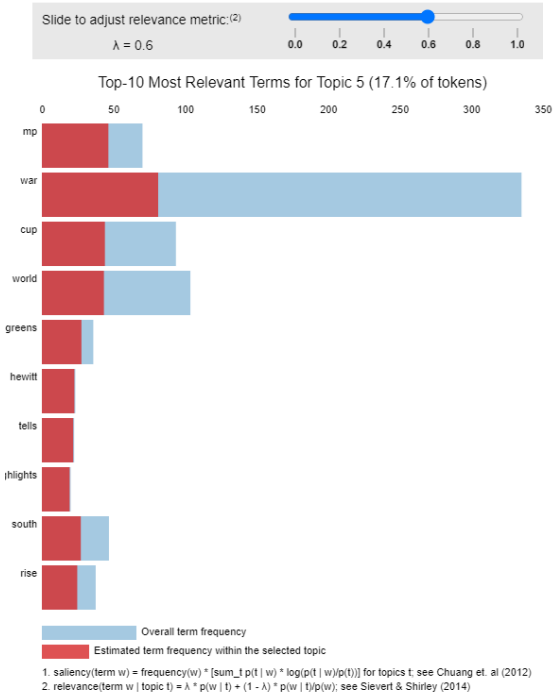
- LDA: understanding the results
  - The left panel shows the predominance of the topics and how related (overlapping) they are
    - Ideally, one wants to avoid overlapping as much as possible
    - There are different ways of calculating the distance between topics
      - Technically, topics are mapped through dimensionality reduction (PCA/t-sne) on distances between each topic's probability distributions into 2D space
      - For more information, search for the paper "*LDAvis: A method for visualizing and interpreting topics*"



MIAMI UNIVERSITY

Topic Modeling

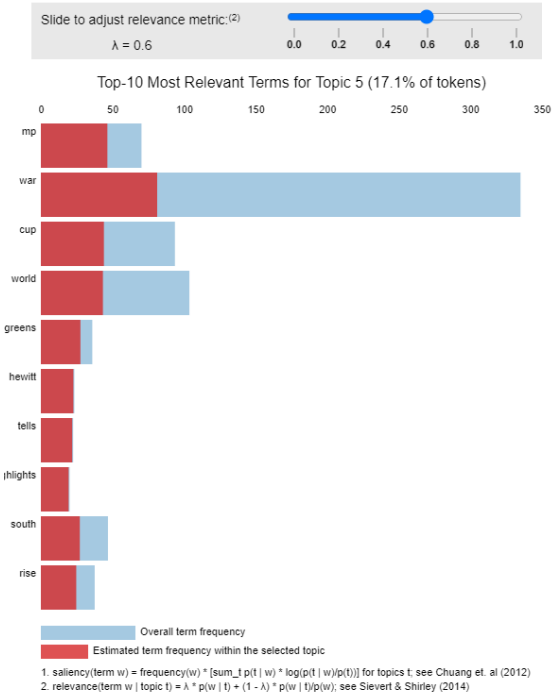
- LDA: understanding the results
  - The right panel shows two metrics
    - **Salience**
      - Thresholding method for selecting which terms are included in the visualization
      - We are selecting the 10 most salient words



MIAMI UNIVERSITY

Topic Modeling

- LDA: understanding the results
  - The right panel shows the **relevance** metric
    - Defined by a parameter lambda λ
      - Value close to 0 selects potentially rare but more exclusive terms for the selected topic
      - Values close to 1 selects more frequently occurring terms in the document that might not be exclusive to the topic
      - Technically λ weighs the probability of a term within a topic relative to its lift
        - For more information search for the paper "*LDAvis: A method for visualizing and interpreting topics*"



MIAMI UNIVERSITY

Topic Modeling

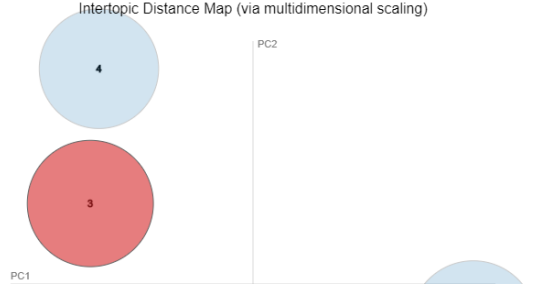
- LDA: understanding the results
  - The optimal value of λ is context-dependent
    - The authors of this visualization tool got good results with $\lambda = 0.6$
  - So, what are the topics in our corpus?
    - It depends upon the data analyst to find similar characteristics between the documents of one cluster and assign it an appropriate label or topic
    - Consequently, it is extremely difficult to evaluate the performance of topic modeling since there are no right answers

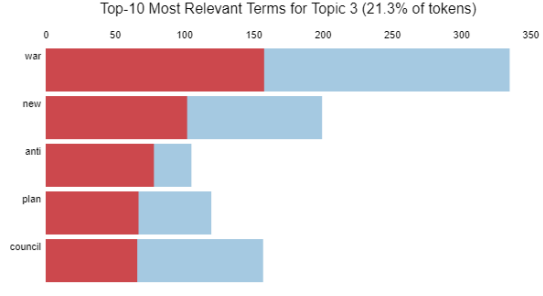
Topic Modeling

- LDA: understanding the results
  - The optimal value of λ is context-dependent
    - The authors of this visualization tool got good results with $\lambda = 0.6$
  - When $\lambda = 1$, we get the same results we obtained before in Slide 18
    - **VERY IMPORTANT POINT:** the topic numbers in LDAVis are not the same as in the LDA model
      - Example:

```
First topic:
['killed', 'crash', 'forces', 'claims', 'baghdad', 'council', 'plan', 'anti', 'new', 'war']
```

The first topic of the LDA model is the 3rd in pyLDAvis





Topic Modeling

- LDA: understanding the results
  - Before moving on, it is important to map each topic from our LDA model onto the topics in pyLDAvis
    - Set $\lambda = 1$ and determine matches between the top 10 words
    - Example always build a table like this:

LDA	pyLDAvis
Topic 1	Topic 3
Topic 2	Topic 5
Topic 3	Topic 2
Topic 4	Topic 1
Topic 5	Topic 4

Topic Modeling

- LDA: understanding the results
  - Let's assume we agree on the following topics in our corpus

LDA	pyLDAvis	Topic Label
Topic 1	Topic 3	Rally
Topic 2	Topic 5	?
Topic 3	Topic 2	International
Topic 4	Topic 1	Trial
Topic 5	Topic 4	Security

Topic Modeling

- LDA: understanding the results
  - One can always evaluate the appropriateness of the topic labels and numbers by going back to the original data
    - Let's apply our LDA model to our DTM to assign a topic to each document

      ```
      topic_values = LDA.transform(dtm)
      ```
  - The above command results in a distribution over topics for each documents
    - Example for the first document: [0.03 0.04 0.53 0.03 0.37]

Topic Modeling

- LDA: understanding the results
  - Which label (topic) should we assign to the documents?
    - Traditional approach: the topic associated with the highest probability
  - Technical note: all the arrays we are working with today are of a special type from the numpy module
    - N-dimensional array (**ndarray**)
    - **ndarray** comes with several handy functions
      - argmax: returns the index of the maximum value in a row
      - astype: transforms all the elements of the array to a single type

```
max_topic_values = topic_values.argmax(axis=1)
max_topic_values = max_topic_values.astype("str")
```

Topic Modeling

- LDA: understanding the results
  - It is time now to replace topic labels
    - That is, instead of number, let's use meaningful words (see slide 25)
      - Recall that 0 is the first index in Python

```
max_topic_values[max_topic_values == "0"] = "Rally"
max_topic_values[max_topic_values == "1"] = "?"
max_topic_values[max_topic_values == "2"] = "International"
max_topic_values[max_topic_values == "3"] = "Trial"
max_topic_values[max_topic_values == "4"] = "Security"
```

LDA	Topic Label
Topic 1	Rally
Topic 2	?
Topic 3	International
Topic 4	Trial
Topic 5	Security

Topic Modeling

- LDA: understanding the results
  - Now, we have topics, and they are appropriately labeled
  - Let's merge these labels with our original, raw data
    - We will then be able to observe whether the labels/topics are good

```
data_w_topic = pandas.DataFrame()
data_w_topic["Text"]  = raw_data.iloc[0:10000,1]
data_w_topic["Topic"] = max_topic_values
```

Topic Modeling

- A trained LDA model can be used to extract topics from never-before-seen documents
  - Same procedure as before
    - Create a DTM with the new data
    - Apply the model to the DTM
    - Assign the topic with the highest probability to each document
  - Look at today's notebook for an example involving two documents that do not belong to the training data

Topic Modeling

- Keep in mind that topic modeling is not a one-shot approach
  - Our results are nowhere close to be perfect
  - We could try the following:
    - Remove some more words (stop words)
    - Perform n-gram (2-gram)
    - Plot more than 10 documents per document
    - Increase the amount of data used for training
      - We only used 10,000
    - Increase the number of topics
      - We assumed only 5
    - Play with some arguments of LDA
    - …

Homework #8

- Let's retrain our LDA model by using a different corpus
  - This will not necessarily improve your model, but it will help you understand how to better prepare a corpus for LDA training
  - Task 1: create vector having 10000 headlines (same data we used today)
  - Task 2: add the word "man" to the stop list
    - Hint: import sklearn.feature_extraction.text.ENGLISH_STOP_WORDS and create a new stop list as follows: new_stop_list = ENGLISH_STOP_WORDS.union(["man"])
  - Task 3: create a CountVectorizer transformation using:
    1. The new stop list;
    2. 1- and 2-grams (parameter ngram_range),
    3. Using only words present in at most 80% of the corpus (parameter max_df)
    4. Using words present in at least 2 documents (parameter min_df)
    - Hint: read the documentation at https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

Homework #8

- Let's retrain our LDA model by using a different corpus
  - Task 4: create a DTM
  - Task 5: fit an LDA model to the DTM having 5 topics
  - Task 6: use pyLDAvis to help you determine the topics inside each text
  - Task 7: answer the following questions on Canvas:
    1. Based on your analysis, what are the 5 resulting topics?
    2. You will likely have overlapping circles in the left panel produced by pyLDAvis. What does that mean?
    3. Why do you have terms that are made up of two words in the right panel of pyLDAvis?

Summary

- We learned another technique to analyze textual data
  - Topic modeling

- Next class
  - Social media analytics and the midterm project