

## Lab Assignment 2

CIS 660 Data Mining  
SS Chung

### Part 1:

#### Preprocessing to Build Document Vectors for Web Page Content Analysis

A document can be represented as a **Bag of Words** that is transformed to a set of terms in thousands of terms (features) as a term dictionary with the *frequency* of each term/word occurring in each document. – For this, you have to build an Inverted Index (which is a Term Look\_Up Table or Term Dictionary with Term Frequency (TF) and Document Frequency (DF). See how to build TF-IDF Inverted Index in Slides 10 – 14 in the Lecture Notes here.

<http://eecs.csuohio.edu/~sschung/CIS660/L03InvertedIndex.pdf>

For text analysis tasks, for example, to categorize (cluster) each document in a collection into a certain number of topics or to find the Top N most related documents in a collection per a given user query (topics) in a Question Answering (QA) System, each document can be transformed to be represented as a vector of topic terms (topic words, keywords or phrases like bi-gram or a tri-gram) occurring in the document with their frequencies.

Construct a document vector with the 6 given topics below with their frequency count for each topic word in the following 5 webpages. The 6 terms for topics are the following 4 keywords and 2 phrases (bi-grams) as below.

Topic words (Keywords) to Analyze:

engineering, research, data, mining, data mining, machine learning

Doc1:

<http://cis.csuohio.edu/~sschung/>

Doc2:

<https://en.wikipedia.org/wiki/Engineering>

Doc3:

<http://my.clevelandclinic.org/research>

Doc4:

[https://en.wikipedia.org/wiki/Data\\_mining](https://en.wikipedia.org/wiki/Data_mining)

Doc5;

[https://en.wikipedia.org/wiki/Data\\_mining#Data\\_mining](https://en.wikipedia.org/wiki/Data_mining#Data_mining)

Right Click on each webpage -> view sourcing then save the html file as .txt file. Those 5 text files are your input files to process to count each word frequency to construct 5 document vectors for the 5 doc as below.

engineering, research, data, mining, data mining, machine learning  
doc1  
doc2  
doc3  
doc4  
doc5

1. We want to count only the words appeared in the Webpage Text as the Content of the page, not the words included inside any tags <.....> or any system generated scripts or html codes. You can count the words appeared in the title bar as well as in the Menu in the webpage
2. We do not want to count any subexpressions that are a part of another words.  
  
“Spin” should not be counted as “pin”
3. However, No case sensitive: Insert, insert, INSERT, insert are all counted as a same word.
4. The words from a same stem are counted as a same word.  
For example, program, programming, programed, programmable are all counted as “program”.  
You can directly add OR conditions with all the variations of the words that are from a same stem to count all as a same word.
5. We want to count for a phrase (bi-grams or tri-grams) by counting occurring of ‘data mining’, for example, when ‘data’ immediately followed by ‘mining’.  
This is usually done to add a discovered bi-gram or tri-gram in the term dictionary as a single term, for example, ‘data mining’ is added as a single term with ‘data\_mining’ in the term dictionary with its frequency.
6. See FAQ for Lab2 on the webpage for more guides.

### Common NLP Preprocessing Procedures for Text Analysis

#### Minimum Requirements:

1. Remove all the special symbols like punctuation mark, question mark using the character deletion step of translate
2. Remove all stop words (Search for Stop word Lists or Python or R Libs)
3. Do Stemming to Reduce inflected (or sometimes derived) words to their word stem.
4. Convert uppercase to lowercase

For more accurate Text Preprocessing, see Lab2 Section.

You can build your term dictionary with each term frequency (Inverted Index) first to construct 5 document vectors for the given Topics. Although DF is highly recommended to add, you can ignore document frequency for simplicity for this lab.

You may omit to build an Inverted Index (Term Dictionary) with TF-IDF.

You can directly construct 5 document vectors to count from your parsing script for this lab.

Inverted Index (Term Dictionary) Construction with TF-IDF will be counted as an Extra Credit.

- You can use or adopt any online word count program for this Lab if you want. For example, `import java.util.StringTokenizer;`
- You can make any assumptions to simplify the program.
- Briefly make notes on these in your report.

You can create a Term Dictionary with TF (and DF) in a Table format or in MongoDB with the scheme. Your Term Dictionary with TF and DF would look like either one of those tables below.

[illegible]

For an Inverted Index in Mongo DB, See [the Common NLP Tasks](#) in Lab2 section.

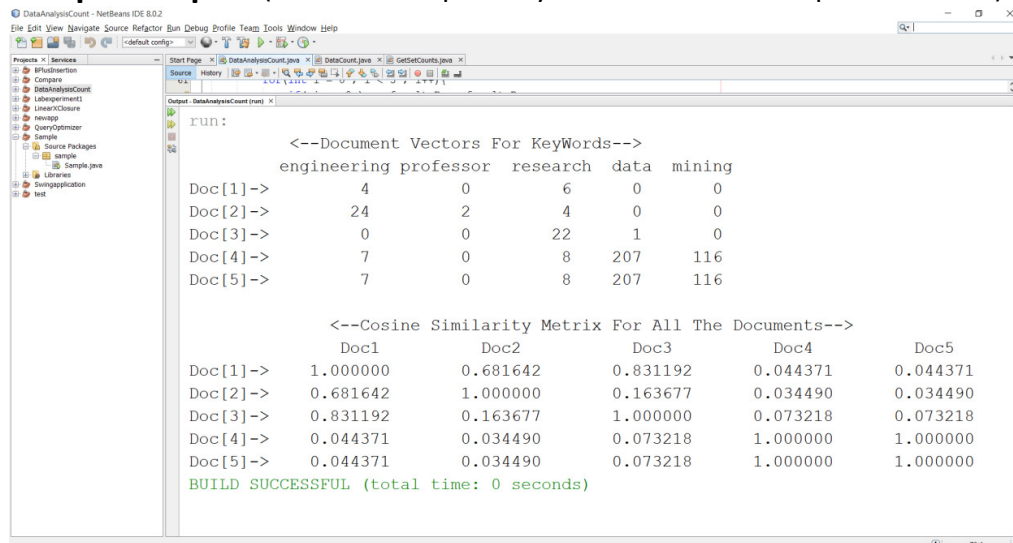
## Part 2: Data Transformation for Topic Analysis of Documents (Webpages)

Using the 5 document vectors, construct a cosine similarity matrix like the one below to indicate a cosine similarity between every pair of document doc1 – doc5. You may want to normalize each document vector first to calculate each  $\cos(d_i, d_j)$  as in the example in the Lecture Notes on Information Retrieval.

- Cosine measure: If  $d_1$  and  $d_2$  are two vectors (e.g., term-frequency vectors), then  $\cos(d_1, d_2) = (d_1 \bullet d_2) / (||d_1|| \cdot ||d_2||)$ , where  $\bullet$  indicates vector dot product,  $||d||$ : the length of vector  $d$

doc1 doc2 doc3 doc4 doc5  
doc1  
doc2  
doc3  
doc4  
doc5

**Sample Output:** (Note this output may not be a correct output of this lab)



```
run:
    <--Document Vectors For Keywords-->
    engineering professor    research  data  mining
Doc[1]->      4      0      6      0      0
Doc[2]->     24      2      4      0      0
Doc[3]->      0      0     22      1      0
Doc[4]->      7      0      8     207     116
Doc[5]->      7      0      8     207     116

    <--Cosine Similarity Matrix For All The Documents-->
    Doc1      Doc2      Doc3      Doc4      Doc5
Doc[1]->  1.000000  0.681642  0.831192  0.044371  0.044371
Doc[2]->  0.681642  1.000000  0.163677  0.034490  0.034490
Doc[3]->  0.831192  0.163677  1.000000  0.073218  0.073218
Doc[4]->  0.044371  0.034490  0.073218  1.000000  1.000000
Doc[5]->  0.044371  0.034490  0.073218  1.000000  1.000000
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Analysis and Discussion

Discuss briefly about your topic analysis from your cosine similarity focusing on whether the indications by the values of your Cosine Sim are all correct?

The Topics of Doc1 is similar to the Topics of Doc 4 and 5? Explain Why or Why Not in terms of 6 TFs? If not, what are the reasons?