Lab Course: Distributed Data Analytics
Exercise Sheet 2

Prof. Dr. Dr. Schmidt-Thieme, Daniela Thyssens
Information Systems and Machine Learning Lab
University of Hildesheim
Submission deadline: Friday May 13, 23:59PM (on LearnWeb, course code: 3116)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit two items a) a zipped file containing python scripts and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission needs to be made before the deadline, only through learnweb.
4. Unless explicitly stated, you are not allowed to use scikit, sklearn or any other library for solving any part. All implementations must be done by yourself.
5. In each lab you have to show your solution works for $P = \{2, 4, 6, 8 \ldots\}$ and provide the timing results (whether it is stated in the question or not)

Complex Data Lab: Processing Text Data in a Distributed Setting

In this exercise sheet, you are going to apply distributed computing concepts using Message Passing Interface API provided by Python (mpi4py) to Natural Language Processing (NLP) techniques using complex data in the form of text documents. The NLP application uses machine learning models to understand the human language and speech. The text data is usually large and consists of complex structures. In this lab you will use MPI framework to process large natural language corpora.

More precisely, you are going to do some basic tasks in NLP including data cleaning, text tokenization and convert words into their Term Frequency, Inverse Document Frequency (TFIDF) scores.

Dataset and a scikit-learn

You will use "Twenty Newsgroups" corpus (data set) available at UCI repository `https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups`. It consists of 20 subfolders each belong to a specify topic. Each subfolder has multiple documents.

```
alt.atheism           comp.sys.ibm.pc.hardware  misc.forsale    rec.sport.baseball  sci.electronics  soc.religion.christian  talk.politics.misc
comp.graphics         comp.sys.mac.hardware     rec.autos       rec.sport.hockey    sci.med          talk.politics.guns      talk.religion.misc
comp.os.ms-windows.misc  comp.windows.x         rec.motorcycles  sci.crypt          sci.space        talk.politics.mideast
```

You can look at the the following blog post to get yourself familiarized with TFIDF calculation available in scikit-learn (`https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit`
Note: This is just a tutorial you will not use scikit-learn to solve the final task.

Exercise 1: Data cleaning and text tokenization (5 points)

In a text document you encounter words that are not helpful for your final model. For example, punctuations and numbers, meaningless words, common English stopwords etc. Your first task is to preprocess your data so you can remove these words from the documents. Your solution should be based on MPI framework. You have to develop (write code) a distributed algorithm that cleans the data by removing

the above mentioned words/numbers. You can take help of some python libraries i.e. NLTK. The developed program should take a set of documents in raw format as input, and outputs a set of documents that are cleaned and tokenized.
Please explain your solution and how you distribute work among workers.

1. Cleaning: remove all punctuations, numbers and The list of common English stopwords used in this exercise sheet can be found in the reference [4].
2. Tokenize: Tokenize your documents so it is easy to process in the next task i.e. Tokenize words and output as a comma separated document.

Exercise 2: Calculate Term Frequency (TF) (5 points)

The Term Frequency (TF) is calculated by counting the number of times a token occurs in the document. This TF score is relative to a specific document, therefore you need to normalized it by dividing with the total number of tokens appearing in the document. A normalized TF score for a specific token t in a document d can be calculated as,

$$\mathrm{TF}(t,d) = \frac{n^d(t)}{\sum_{t' \in d} n^d(t')} \ , \tag{1}$$

where $n^d(t)$ is the number of times a token t appears in a document d and $|d|$ is the total number of tokens in the document d.
Develop an solution using MPI framework and write code. Please explain how you parallelize (or distribute) TF(t,d) calculation. Also explain your strategy from the data division and calculation division point of view as well. Perform a small experiment by varying number workers i.e. $P = \{2, 4, 8\}$. Also verify if you get the same result at the end.

Exercise 3: Calculate Inverse Document Frequency ((IDF) (5 points)

The Inverse Document Frequency ((IDF) is counting the number of documents in the corpus and counting the number of documents that contain a token. While the TF is calculated on a per-document basis, the IDF is computed on the basis of the entire corpus. The final IDF score of a token t in the corpus C is obtained by taking logarithm of document count in the corpus divided by the number of documents in the corpus that contain a particular token. The IDF formula is given as

$$\mathrm{IDF}(t) = \log \frac{|C|}{\sum_{d \in C} \mathbb{1}(t,d)} \ , \tag{2}$$

where $\mathbb{1}(t,d)$ is an indicator function which returns 1 if a token t exists in document d. $|C|$ is the total number of documents in the corpus.

Develop an solution using MPI framework and write cod. Please explain how you parallelize (or distribute) IDF(t,d) calculation. Also explain your strategy from the data division and calculation division point of view as well. Perform a small experiment by varying number workers i.e. $P = \{2, 4, 8\}$. Also verify if you get the same result at the end.

Exercise 4: Calculate Term Frequency Inverse Document Frequency (TF-IDF) scores (5 points)

In this exercise you will find the TF-IDF(t,d) for a given token t and a document d in the corpus C is the product of TF(t,d) and IDF(t), which is represented as,

$$\textit{TF-IDF}(t,d) = \textit{TF}(t,d) \times \textit{IDF}(t) \ , \tag{3}$$

You have already calculated $\textit{TF}(t,d)$ in Exercise 2 and $\textit{IDF}(t)$ is Exercise 3.
In this exercise you have to think about how you can combine the complete pipeline in a single parallel/distributed program that can run worker $P = \{2, 4, 8\}$. Develop an solution using MPI framework

and write code. Please explain how you parallelize (or distribute) the complete pipeline. Also explain your strategy from the data division and calculation division point of view as well. Perform a small experiment by varying number workers i.e. $P = \{2, 4, 8\}$. Also verify if you get the same result at the end.

Annex

1. Introduction to Natural Language Processing (NLP):
   `http://blog.algorithmia.com/introduction-natural-language-processing-nlp/`
2. TFIDF `http://www.tfidf.com/` and `https://en.wikipedia.org/wiki/Tf\%E2\%80\%93idf`
3. Tutorial: Finding Important Words in Text Using TF-IDF
   `http://stevenloria.com/finding-important-words-in-a-document-using-tf-idf/`
4. Common English stopwords: `http://www.textfixer.com/tutorials/common-english-words.txt`
5. Data-Intensive Text Processing with MapReduce
   `http://www.umiacs.umd.edu/~jimmylin/MapReduce-book-final.pdf`