Text Analytics
Overview of Topics covered to date, and how they relate to CA work

1 GETTING THE DATA

1.1 WORKING WITH TEXT FILES

Organise text files into folders by topic, and import to Rapidminer using **PROCESS DOCUMENTS FROM FILE**

1.2 WORKING WITH WEB FILES

1.2.1 USING A WEB CRAWLER

Using the operator **WEB CRAWL**. Supply a root URL, and limit the search using regular expression. The operator will save pages found to a specified folder. Use a different folder for each topic. A separate process can read these pages into Rapidminer using **PROCESS DOCUMENTS FROM FILE**

1.2.2 Using GET PAGES

Provide a separate URL for all the pages you want to use. Each page will be downloaded from the web, and saved as an example set. The output is passed into **PROCESS DOCUMENTS FROM DATA**

2 Generating word clouds

There are a number of online tools that generate word clouds and other visualisation such as http://www.wordle.net.

3 Generating a bag of words (operators embedded in Process Documents)

3.1 For web data only

Use **EXTRACT CONTENT** to strip HTML tags and identify blocks of text based on minimum text block length. Optionally use **EXTRACT INFORMATION** to extract additional fields, specified using regular expression or XPath

3.2 For all data . . .

3.2.1 Tokenise

For most texts, use the **TOKENISE** operator. If the quality of the text is very poor, and the texts are short, use **GENERATE N-GRAMS (CHARACTERs)** instead.

3.2.2 Decide on what Filters to use

Options here include filtering predefined stop words (**FILTER STOP WORDS (ENGLISH)**), filter your own list of stop words (**FILTER STOP WORDS (DICTIONARY)**). You can also filter tokens based on their length, or their Parts of Speech (POS). Also see additional filtering based on term counts in section 3.2 below. The objective is to remove terms that will not be useful in distinguishing between documents about different topics.

3.2.3 Identifying phrases

If combinations of terms are likely to be more useful than individual tokens, **GENERATE N-GRAMS (TERMS)** will combine adjacent tokens.

3.2.4 Decide on Stemmers to use

Try providing your own list of synonyms (**STEM (DICTIONARY)**), and experiment with Porters and Lovins stemmers as well. The objective is to convert similar terms into a single token.

4 Creating the document vector

4.1 Term counts

How terms are counted (frequent, TF-IDF etc.) is defined by the **PROCESS DOCUMENTS** operators. Select the term count using the Vector Creation parameter.

4.2 Additional filters

The **PROCESS DOCUMENTS** operators also facilitates further pruning of terms based on their counts. There are two types of counts:

- **Prune by rank** is based on the total number of times a term appears across all the documents in the collection.
- **Prune by percent** **or** **Prune by percent absolute** is based on the total number of documents a term appears in, regardless of how often it appears in a particular document document.

5 Building models

The output from the **PROCESS DOCUMENTS** operators is a dataset (also called a document vector) than can be modelled just like any other dataset, using classification or clustering algorithms

6 Applying a model to a new collection of documents

To apply the model to a new collection of documents, you need to use:

1. The same bag of words. Store the bag of words from the training process and use it as input to the test process (see lab 3)
2. The same model. As above, store the model from the training process and use it as input to the test process (see lab 3)
3. Same processing steps:
    - All stemmers applied to the training dataset must also be applied to the test dataset
    - Use the same type of term count (TF-IDF etc.)
    - Filters will happen anyway because of the bag of words provided.