Assignment 5 – Text mining (Sentiment-mining)
Due: Nov 25th, 2015

This assignment is based on movie-review data for sentiment analyses. It is based on a collection of movie-review documents labeled with respect to their overall sentiment polarity (positive or negative). We will build and evaluate the effectiveness of classification models – models that can help predict the sentiment polarity.

(More details on the corpus is available from http://www.cs.cornell.edu/people/pabo/movie-review-data/. Note – we use the sentiment polarity dataset of 1000 'positive' and 1000 'negative' reviews.)

The text mining process involves the "bag of word" approach, with standard steps for creating the document-term matrix (word vectors for each document; each row as document) - with either binary term presence/absence values, term occurrences, or tf-idf values. The steps are:

- Tokenize
- Filter stopwords
- Transform case (to all lower/upper)
- Filter tokens by length - say, min 4 and max 15.
- (optionally) Filter tokens by content – if they match a 'dictionary' of terms
- (optionally) Stemming

You may also wish to deselect words that occur in too few documents and/or in most of the documents.

The optional steps above will be what you experiment with to determine what works best.

1. Read a few of the movie-reviews. If you were to classify manually, are there certain words/terms indicative of 'positive' and 'negative' that you would use? Provide lists ('dictionaries') of 'positive' and 'negative' terms. Now consider a manual classification process – you may count the number of 'positive' and 'negative' terms (matching your dictionaries) and classify based on majority count. How effective is this (accuracy?)?

2. Using automated text-mining: perform the steps noted above to get a document-term matrix. How many terms do you have? With and without stemming?
   (Note: processing the corpus to get the document-term matrix is computationally expensive and can take time. Rather than repeat these steps for experiments with different learning algorithms/parameters, you may want to store the document-term matrix, and use this stored 'data'.)

3. The sentiment polarity of a document provides the 'label'/target variables. Learn knn and naïve-Bayes models to classify for sentiment polarity.
   To evaluate effectiveness in prediction, we will consider Split-Validation, with 50% of the documents used for training and 50% for test (you can set a local random number seed for the Split-Validation operator to get the same split on the data across experiments).

   (a) Develop and evaluate knn and naïve-Bayes models. What values of k work best? Evaluation is based on the confusion matrix – overall accuracy, accuracies on positive and negative, precision, specificity, sensitivity, etc.

You should experiment with the following to determine how they affects performance:
(i) measures defining the document term matrix – binary term occurrence, term occurrence, term frequency, tf-idf (please define these four term as part of the assignment)
(ii) which words to include
- without and with stemming
- using words that match specific POS tags – you need to determine which POS tags you want to include (there are multiple websites that give the list of POS tags)
- pruning words based on min and max of the number of documents they appear in

(b) For any one setting from above (for example, using tf-idf, stemmed terms, and using some reasonable pruning), build a support vector machine model . Try SVM with dot-kernel. How does the performance of the SVM model compare with k-nn and naïve-Bayes models? Does using a radial basis function kernel (nonlinear model) perform better? Also try a random forest model and compare performance. What do you conclude overall?

(c) Use the Harvard-IV dictionary of positive and negative terms - Filter Tokens by Content to keep only these terms. How many terms do you get? (Note: for this, the document terms should not be stemmed, since they would otherwise not match the unstemmed dictionary terms. Does it make sense to use pruning of terms or POS tags here?).
Use just the count of positive and negative terms to classify sentiment of reviews. How do you do this, and what performance do you find?
Develop and evaluate a naïve-Bayes, a SVM (either with dot kernel or other kernel based on what you found to perform better in part (b) above), and a random forest model, using only the positive and negative words.

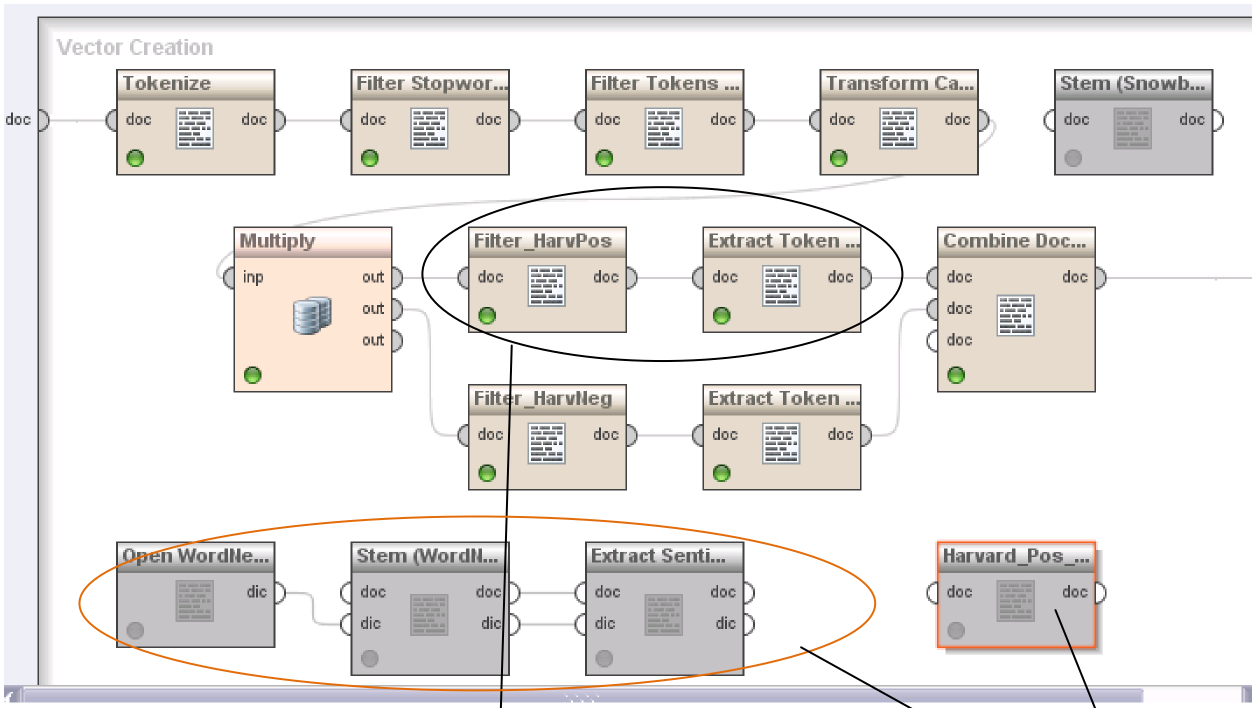
(d) Next, use Wordnet and Senti-Wordnet to determine the sentiment polarity of the reviews. Provide a description of how this process works – i.e. how WordNet and Senti-Wordnet operates (there are various resources on the web on this; please cite your references, and please remember that we cannot copy-paste without quotes…! Ask Sid if you have any questions about this).
Use the Senti-WordNet sentiment scores to classify the reviews. What performance do you find?

(d) How does performance in (a) - (d) above compare? What can you conclude? Does the smaller list of 'sentiment' words provide similar performance levels as models built on a broader set of terms? How does the performance of Senti-WordNet compare? Does it help to include Senti-WordNet scores in a model with sentiment-words or with the broader set of terms?Which is your 'best' model?

The Harvard-IV dictionaries are available for various 'categories'. Details are available in http://www.wjh.harvard.edu/~inquirer. The list of positive and negative terms are given in the attached txt files.
In RapidMiner, the Filter Token by content operator requires the dictionary terms specified as (for example): abide|ability|able|…. (this is a regular expression, with '|' indicating 'or'; there should be no 'blanks'). The operator with these words in included in the rmp file (see description below).

For using Wordnet, you will need to download the Wordnet dictionary to a folder, and specify the folder location in the Open WordNet Dictionary operator. You will also need to use the Stem(WordNet) operator and the Extract Sentiment (English) Wordnet operator. The Extract Sentiment operator provides a sentiment score in [-1,1] for each document. You can use this sentiment 'score' directly. You can also build a model where this score is taken as another attribute, along with other terms in the corpus.

Inside the Process Documents operator:



Filter (keep) on the words corresponding to Harvard Positive words list.

Extract Token Number: extracts the number of positive tokens in the document. This is then stored as meta-data for each document.

Similarly for the Harvard Negative words in the operators below this.

Operators to use Senti-WordNet:
1. Open WordNet Dictionary (specify the folder where you have downloaded and unzipped the dictionary)

2. Use WordNet based Stemming

3. Extract Sentiment for the document based on Senti-WordNet.

This can be used to filter (keep) all words in the Harvard Positive and Negative words list.