Quantitative Text Analysis
Exercise 7: Supervised Scaling

29th July 2014, Essex Summer School

Kenneth Benoit and Paul Nulty

In today's lab we will use R to apply Wordscores scaling to a set of amicus briefs from US Supreme Court cases on affirmative action in college admissions. (Evans et al 2007).

Amicus curiae are persons or organizations not party to a legal case who are permitted by the court to offer it advice in the form of an *amicus brief*. The amicus briefs in this corpus are from an affirmative action case in which an applicant to a university who was denied a place petitioned the Supreme Court, claiming that they were unfairly rejected because of affirmative action policies.

Amicus curiae could advise the court either in support of the petitioner, therefore opposing affirmative action, or in favour of the respondent — the University— therefore supporting affirmative action.

Bolinger case

Instructions

This lab will be similar to yesterday's, but with the order of the parts reversed. First, we will use the built-in classification functions to predict using wordscores, and examine the confusion matrix and the quantities returned by the training function. Then in part two, we'll compute the wordscore manually.

1. The amicus corpus comes built-in to `quanteda`, but lets revise the process of building it from the texts.

   (a) This code loads the texts, extracts labels (P for Petitioner, R for Respondent) from the filenames, and makes a new corpus:

   ```
   library(quanteda)
   data(amicusTexts)
   # set training class
   trainclass <- factor(c("P", "R", rep(NA, length(amicusTexts)-2)))
   # set test class
   testclass  <- rep(NA, length(amicusTexts))
   testclass[grep("AP", names(amicusTexts))] <- "AP"
   testclass[grep("AR", names(amicusTexts))] <- "AR"
   amicusCorpus <- corpusCreate(amicusTexts,
        attribs=list(trainclass=trainclass, testclass=testclass))
   ```

   (b) Make a document-feature matrix (dfm) from the corpus.

   (c) Use the `naiveBayesText` function to train the model, and the `predict` function to make predictions. The syntax for this is the same as yesterday's lab — the `naiveBayesText` function also contains the Wordscores, and the predict function also performs Wordscores predictions. The code for applying this to the amicus briefs is also available on the main `quanteda` github page.

(d) Examine the object returned by the `predict` function (hint (use the `names` command).

(e) Look at the confusion matrixes for the wordscores and naive bayes predictions, and compare the accuracies and f-scores.

```
nbrestab <- table(amicus.nbp$docs$nb.predicted, amicusCorpus$attribs$testclass )
wsrestab <- table(amicus.nbp$docs$ws.predicted, amicusCorpus$attribs$testclass )
```

2. Now we compute the Wordscores manually from the training examples.

(a) The training texts used in this example were the first two texts in the amicus corpus. Make a new dfm from them (use the `dfm` command, with the two texts as an argument:

```
trainTexts <- c(amicusTexts[[1]],amicusTexts[[2]])
trainDfm <- dfm(trainTexts)
```

(b) Use a reference score of -1.0 for the Petitioner text and 1.0 for the Respondent text.

(c) Compute the Wordscores. Remember to normalize the frequency dfm, either like we did yesterday with the sum of the row, or just use the `tf` command on the dfm.

(d) Sort the Wordscores and examine the highest and lowest scores. Are they predictive for substantive or procedural reasons? Find examples of words that would have good prediction on this corpus, but would not transfer to a generalized liberal/conservative scale.