15.4 Latent Semantic Analysis

Information retrieval systems (like search engines) and people doing computational text analysis often represent documents as what are called **bags of words**: documents are represented as vectors, where each component counts how many times each word in the dictionary appears in the text. This throws away information about word order, but gives us something we can work with mathematically. Part of the representation of one document might look like:

```
 a abandoned abc ability able about above abroad absorbed absorbing abstract
43         0   0       0    0    10     0      0        0         0        1
```

and so on through to "zebra", "zoology", "zygote", etc. to the end of the dictionary. These vectors are very, very large! At least in English and similar languages, these bag-of-word vectors have three outstanding properties:

1. Most words do not appear in most documents; the bag-of-words vectors are very **sparse** (most entries are zero).
2. A small number of words appear many times in almost all documents; these words tell us almost nothing about what the document is about. (Examples: "the", "is", "of", "for", "at", "a", "and", "here", "was", etc.)
3. Apart from those hyper-common words, most words' counts are correlated with some but not all other words; words tend to come in bunches which appear together.

Taken together, this suggests that we do not really get a lot of value from keeping around *all* the words. We would be better off if we could project down a smaller number of new variables, which we can think of as combinations of words that tend to appear together in the documents, or not at all. But this tendency needn't be absolute — it can be partial because the words mean slightly different things, or because of stylistic differences, etc. This is *exactly* what principal components analysis does.

To see how this can be useful, imagine we have a collection of documents (a **corpus**), which we want to search for documents about agriculture. It's entirely possible that many documents on this topic don't actually contain the *word* "agriculture", just closely related words like "farming". A simple search on "agriculture" will miss them. But it's very likely that the occurrence of these related words is well-correlated with the occurrence of "agriculture". This means that all these words will have similar projections on to the principal components, and it will be easy to find documents whose principal components projection is like that for a query about agriculture. This is called **latent semantic indexing**.

To see why this is *indexing*, think about what goes into coming up with an index for a book by hand. Someone draws up a list of topics and then goes through the book noting all the passages which refer to the topic, and maybe a little bit of what they say there. For example, here's the start of the entry for "Agriculture" in the index to Adam Smith's *The Wealth of Nations*:

> Agriculture, the labour of, does not admit of such subdivisions as manufactures, 6; this

impossibility of separation, prevents agriculture from improving equally with manufactures, 6; natural state of, in a new colony, 92; requires more knowledge and experience than most mechanical professions, and yet is carried on without any restrictions, 127; the terms of rent, how adjusted between landlord and tenant, 144; is extended by good roads and navigable canals, 147; under what circumstances pasture land is more valuable than arable, 149; gardening not a very gainful employment, 152–3; vines the most profitable article of culture, 154; estimates of profit from projects, very fallacious, *ib.*; cattle and tillage mutually improve each other, 220; ...

and so on. (Agriculture is an important topic in *The Wealth of Nations*.) It's asking a lot to hope for a computer to be able to do something like this, but we could at least hope for a list of pages like "6, 92, 126, 144, 147, 152–3, 154, 220,..." . One could imagine doing this by treating each page as its own document, forming its bag-of-words vector, and then returning the list of pages with a non-zero entry for "agriculture". This will fail: only two of those nine pages actually contains that word, and this is pretty typical. On the other hand, they are full of words strongly correlated with "agriculture", so asking for the pages which are most similar in their principal components projection to that word will work great.[13]

At first glance, and maybe even second, this seems like a wonderful trick for extracting meaning, or **semantics**, from pure correlations. Of course there are also all sorts of ways it can fail, not least from spurious correlations. If our training corpus happens to contain lots of documents which mention "farming" and "Kansas", as well as "farming" and "agriculture", latent semantic indexing will not make a big distinction between the relationship between "agriculture" and "farming" (which is genuinely semantic, about the meaning of the words) and that between "Kansas" and "farming" (which reflects non-linguistic facts about the world, and probably wouldn't show up in, say, a corpus collected from Australia).

Despite this susceptibility to spurious correlations, latent semantic indexing is an *extremely* useful technique in practice, and the foundational papers (Deerwester *et al.*, 1990; Landauer and Dumais, 1997) are worth reading.

15.4.1 Principal Components of the New York Times

To get a more concrete sense of how latent semantic analysis works, and how it reveals semantic information, let's apply it to some data. The accompanying R file and R workspace contains some news stories taken from the New York *Times* Annotated Corpus (Sandhaus, 2008), which consists of about 1.8 million stories from the *Times*, from 1987 to 2007, which have been hand-annotated by actual human beings with standardized machine-readable information about their contents. From this corpus, I have randomly selected 57 stories about art and 45 stories about music, and turned them into a bag-of-words data frame, one row per story, one column per word; plus an indicator in the first column of whether

[13] Or it should anyway; I haven't actually done the experiment with this book.

the story is one about art or one about music.[14] The original data frame thus has 102 rows, and 4432 columns: the categorical label, and 4431 columns with counts for every distinct word that appears in at least one of the stories.[15]

The PCA is done as it would be for any other data:

[[ATTN: Why isn't loading from the course website working?]]

```
load("~/teaching/ADAfaEPoV/data/pca-examples.Rdata")
nyt.pca <- prcomp(nyt.frame[, -1])
nyt.latent.sem <- nyt.pca$rotation
```

We need to omit the first column in the first command because it contains categorical variables, and PCA doesn't apply to them. The second command just picks out the matrix of projections of the variables on to the components — this is called `rotation` because it can be thought of as rotating the coordinate axes in feature-vector space.

Now that we've done this, let's look at what the leading components are.

```
signif(sort(nyt.latent.sem[, 1], decreasing = TRUE)[1:30], 2)
##       music        trio     theater   orchestra   composers       opera
##       0.110       0.084       0.083       0.067       0.059       0.058
##    theaters           m    festival        east     program           y
##       0.055       0.054       0.051       0.049       0.048       0.048
##      jersey     players   committee      sunday        june     concert
##       0.047       0.047       0.046       0.045       0.045       0.045
##    symphony       organ     matinee   misstated instruments           p
##       0.044       0.044       0.043       0.042       0.041       0.041
##         X.d       april      samuel        jazz     pianist     society
##       0.041       0.040       0.040       0.039       0.038       0.038
signif(sort(nyt.latent.sem[, 1], decreasing = FALSE)[1:30], 2)
##       she       her        ms         i      said    mother    cooper
##    -0.260    -0.240    -0.200    -0.150    -0.130    -0.110    -0.100
##        my  painting   process paintings        im        he       mrs
##    -0.094    -0.088    -0.071    -0.070    -0.068    -0.065    -0.065
##        me  gagosian       was   picasso     image sculpture      baby
##    -0.063    -0.062    -0.058    -0.057    -0.056    -0.056    -0.055
##   artists      work    photos       you    nature    studio       out
##    -0.055    -0.054    -0.051    -0.051    -0.050    -0.050    -0.050
##      says      like
##    -0.050    -0.049
```

These are the thirty words with the largest positive and negative projections on to the first component.[16] The words with positive projections are mostly associated with music, those with negative components with the visual arts. The letters "m" and "p" show up with music because of the combination "p.m", which our parsing breaks into two single-letter words, and because stories about music give

[14] Actually, following standard practice in language processing, I've normalized the bag-of-word vectors so that documents of different lengths are comparable, and used "inverse document-frequency weighting" to de-emphasize hyper-common words like "the" and emphasize more informative words. See the lecture notes for data mining if you're interested.

[15] If we were trying to work with the complete corpus, we should expect at least 50000 words, and perhaps more.

[16] Which direction is positive and which negative is of course arbitrary; basically it depends on internal choices in the algorithm.

show-times more often than do stories about art. Personal pronouns appear with art stories because more of those quote people, such as artists or collectors.[17]

What about the second component?

```
signif(sort(nyt.latent.sem[, 2], decreasing = TRUE)[1:30], 2)
##         art      museum      images     artists   donations     museums
##       0.150       0.120       0.095       0.092       0.075       0.073
##    painting         tax   paintings   sculpture     gallery  sculptures
##       0.073       0.070       0.065       0.060       0.055       0.051
##     painted       white    patterns      artist      nature     service
##       0.050       0.050       0.047       0.047       0.046       0.046
##  decorative        feet     digital      statue       color    computer
##       0.043       0.043       0.043       0.042       0.042       0.041
##       paris         war collections     diamond       stone     dealers
##       0.041       0.041       0.041       0.041       0.041       0.040
signif(sort(nyt.latent.sem[, 2], decreasing = FALSE)[1:30], 2)
##          her          she      theater        opera           ms
##       -0.220       -0.220       -0.160       -0.130       -0.130
##            i         hour   production         sang     festival
##       -0.083       -0.081       -0.075       -0.075       -0.074
##        music      musical        songs        vocal    orchestra
##       -0.070       -0.070       -0.068       -0.067       -0.067
##           la      singing      matinee  performance         band
##       -0.065       -0.065       -0.061       -0.061       -0.060
##       awards    composers         says           my           im
##       -0.058       -0.058       -0.058       -0.056       -0.056
##         play     broadway       singer       cooper performances
##       -0.056       -0.055       -0.052       -0.051       -0.051
```

Here the positive words are about art, but more focused on acquiring and trading ("collections", "dealers", "donations", "dealers") than on talking with artists or about them. The negative words are musical, specifically about musical theater and vocal performances.

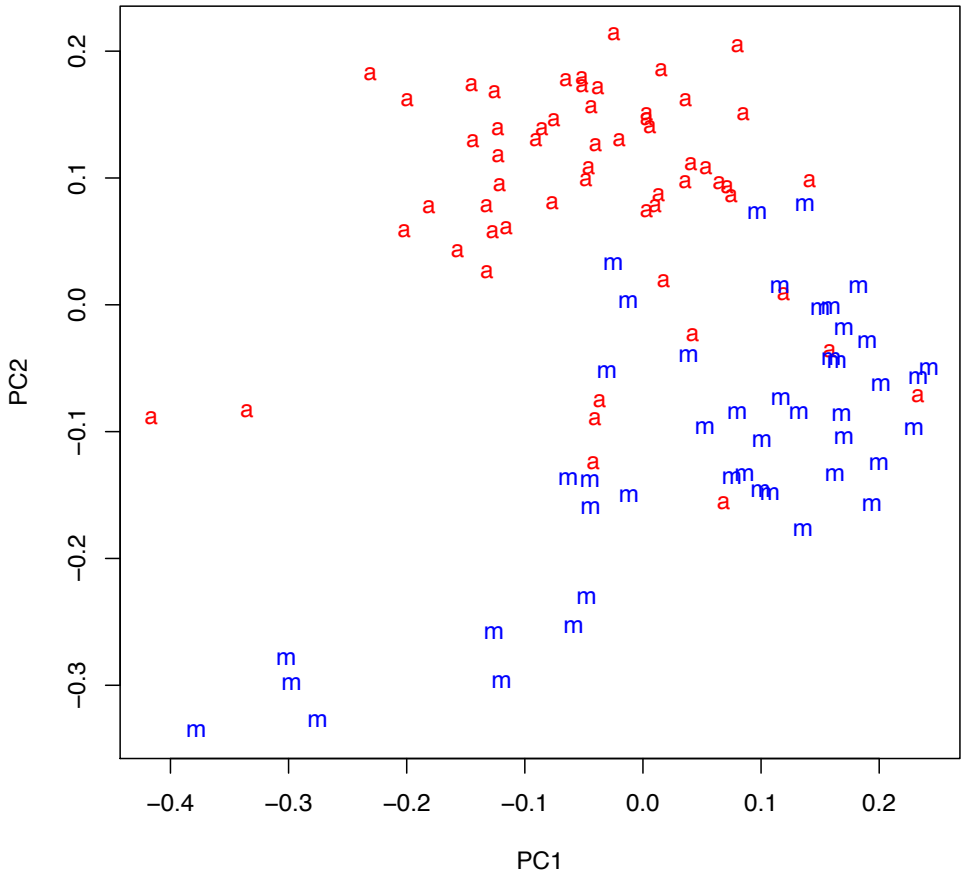
I could go on, but by this point you get the idea.

15.5 PCA for Visualization

Let's try displaying the *Times* stories using the principal components (Figure 15.6).

Notice that even though we have gone from 4431 dimensions to 2, and so thrown away a lot of information, we could draw a line across this plot and have most of the art stories on one side of it and all the music stories on the other. If we let ourselves use the first four or five principal components, we'd still have a thousand-fold savings in dimensions, but we'd be able to get almost-perfect separation between the two classes. This is a sign that PCA is really doing a good job at summarizing the information in the word-count vectors, and in turn that the bags of words give us a lot of information about the meaning of the stories.

[17] You should check out these explanations for yourself. The raw stories are part of the R workspace.




```
plot(nyt.pca$x[, 1:2], pch = ifelse(nyt.frame[, "class.labels"] == "music",
    "m", "a"), col = ifelse(nyt.frame[, "class.labels"] == "music", "blue",
    "red"))
```

Figure 15.6 Projection of the *Times* stories on to the first two principal components. Music stories are marked with a blue "m", art stories with a red "a".

Multidimensional scaling

Figure 15.6 also illustrates the idea of **multidimensional scaling**, which means finding low-dimensional points to represent high-dimensional data by preserving the distances between the points. If we write the original vectors as $\vec{x}_1, \vec{x}_2, \ldots \vec{x}_n$, and their images as $\vec{y}_1, \vec{y}_2, \ldots \vec{y}_n$, then the MDS problem is to pick the images to