Discovery with Text as Data

Review of Two Types of Machine Learning Methods

1. Unspervised
   - kmeans, for example
   - unknown categories
   - often for discovery
2. Supervised
   - what we did in class
   - pre-known, pre-defined categories of interest
   - often for prediction

Today we'll do kmeans again, but with text as data.

Building our document term matrix

Turning language into data. Text is complicated, so we might want to simplify it to “clean” the data. We might want to:

- split into single words (sometimes phrases)
- lower case all text
- remove all punctuation
- remove stop words
- stem
- etc.

Sample documents

This data set contains responses from Gadarian and Albertson (2014) where subjects in the treatment group wrote about what made them anxious about immigration, and those in the control simply wrote about immigration.

Sample documents

[1] “wide open borders that allow terrorists to enter at will increased costs of health care due to illegal immigrants taxing the hospital system talk of amnesty causing a greater influx of illegal immigrants on the other hand, tough immigration practices causing families to be torn apart when illegals are sent home prices rising dramatically due to the loss of inexpensive labor”

[2] “job security, wreckless driving, auto accidents.”

[3] “job loss to those who would work for les pay.”

Building our document term matrix

	job	loss	immigra	illegal
d1	0	1	3	3
d2	1	0	0	0
d3	1	1	0	0
...				

- What assumptions are we making here?
- What are the pros & cons of these assumptions?

Term frequency - inverse document frequency

$$tfidf(w,d) = tf(w,d) \cdot idf(w)$$

where

$$idf(w) = \log\left(\frac{N}{df(w)}\right)$$

N = number of documents

$df(w)$ = number of documents that contain word w

In R

```
dtm_mat[1:5, 1:5]
```

```
##      Terms
## Docs abl alien america american back
##    1   0     0       0        0    0
##    2   0     0       0        0    0
##    3   0     0       0        0    0
##    4   0     0       0        1    0
##    5   0     0       1        0    0
```

In R

```
dtm_tfidf <- as.matrix(weightTfIdf(dtm))
dtm_tfidf[1:5, 1:5]
```

```
##     Terms
## Docs abl alien  america american back
##    1   0     0 0.000000 0.000000    0
##    2   0     0 0.000000 0.000000    0
##    3   0     0 0.000000 0.000000    0
##    4   0     0 0.000000 0.154585    0
##    5   0     0 4.713188 0.000000    0
```

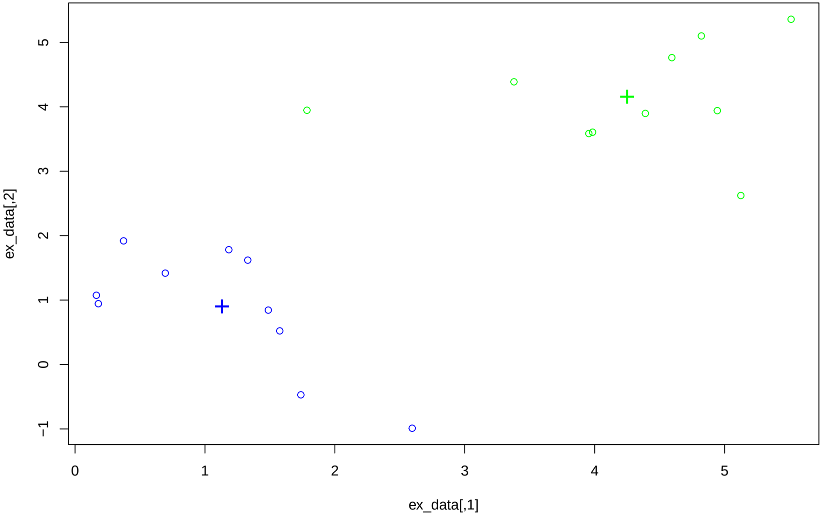
Extending kmeans beyond 2 dimensions

So far, we've always been able to plot and visualize kmeans clusters

```r
set.seed(1)
ex_data <- cbind(c(rnorm(10, 1, 1), rnorm(10, 4, 1)),
                 c(rnorm(10, 1, 1), rnorm(10, 4, 1)))
results <- kmeans(ex_data, centers = 2)
```

Extending kmeans beyond 2 dimensions

```
plot(ex_data, col = ifelse(results$cluster == 1, "blue", "green"))
points(results$centers, col = c("blue", "green"), pch = "+", cex = 2)
```



Now, let's use kmeans to cluster similar documents

- kmeans can be extended to cluster observations with > 2 variables
- imagine we want to cluster similar documents
- instead of using (X,Y) coordinates, we use their word counts or tf-idf measures
- impossible to visualize

Centroids

Now the centroid is defined by each word's mean tfidf value among the documents assigned to the cluster

```
set.seed(2342)
doc_results <- kmeans(dtm_tfidf, centers = 2)

head(doc_results$centers[1,]) ## first centroid
```

```
##        abl      alien    america   american       back
## 0.01004186 0.01477696 0.04648535 0.07158787 0.04999256 0
```

```
table(doc_results$cluster)
```

```
##
##   1   2
## 299  42
```

Clusters

We can see what words are most associated with the cluster

```
colnames(dtm)[order(doc_results$centers[1,], decreasing = TRUE)][1:10]
```

```
##  [1] "immigr"   "job"      "peopl"    "countri"  "come"     "america
##  [7] "think"    "tax"      "need"     "problem"
```

```
colnames(dtm)[order(doc_results$centers[2,], decreasing = TRUE)][1:10]
```

```
##  [1] "illeg"     "mexico"    "border"    "fenc"      "mexican"
##  [6] "worker"    "immigr"    "difficult" "legal"     "alien"
```