TOPIC MODELLING

Riya Agarwal

12/4/2016

In machine learning and natural language processing, a topic model is a type of statistical model for discovering the abstract “topics” that occur in a collection of documents.
The algorithm used for this is Latent Dirichlet Allocation (LDA). It treats each document as a mixture of topics, and each topic as a mixture of words. This allows documents to overlap each other in terms of content, rather than being separated into discrete groups.

Source data:

I will be using the **AssociatedPress** dataset which is a document-term matrix of a collection of 2246 news articles from an American news agency, mostly published around 1988.

Let us load them into R:

```
library(tm)
```

```
## Loading required package: NLP
```

```
data("AssociatedPress", package = "topicmodels")
AssociatedPress
```

```
## <<DocumentTermMatrix (documents: 2246, terms: 10473)>>
## Non-/sparse entries: 302031/23220327
## Sparsity           : 99%
## Maximal term length: 18
## Weighting          : term frequency (tf)
```

We see that this dataset contains documents (each of them an AP article) and terms (words).

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidytext)
```

```
## Warning: package 'tidytext' was built under R version 3.4.2
```

```
a_td <- tidy(AssociatedPress)
a_td
```

```
## # A tibble: 302,031 x 3
##    document       term count
##       <int>      <chr> <dbl>
##  1        1     adding     1
##  2        1      adult     2
##  3        1        ago     1
##  4        1    alcohol     1
##  5        1  allegedly     1
##  6        1      allen     1
##  7        1 apparently     2
##  8        1   appeared     1
##  9        1   arrested     1
## 10        1    assault     1
## # ... with 302,021 more rows
```

The data I have is originally a document-term matrix, exactly what I need for topic modeling.I tidy it because the original document term matrix contains stop words which I want to remove before I model the data. Let us remove the stop words, then cast the data back into a document-term matrix.

```
a_dtm <- a_td %>%
  anti_join(stop_words, by = c(term = "word")) %>%
  cast_dtm(document, term, count)
a_dtm
```

```
## <<DocumentTermMatrix (documents: 2246, terms: 10134)>>
## Non-/sparse entries: 259208/22501756
## Sparsity           : 99%
## Maximal term length: 18
## Weighting          : term frequency (tf)
```

Data exploration

I will plot a histogram of the most common words remaining in the data after the cleaning. The plot below shows a histogram of the ten most frequent words in the corpus. The most frequent word is ‘percent’

```
dtm.matrix <- as.matrix(a_dtm)
wordcount <- colSums(dtm.matrix)
topten <- head(sort(wordcount, decreasing=TRUE), 10)
```

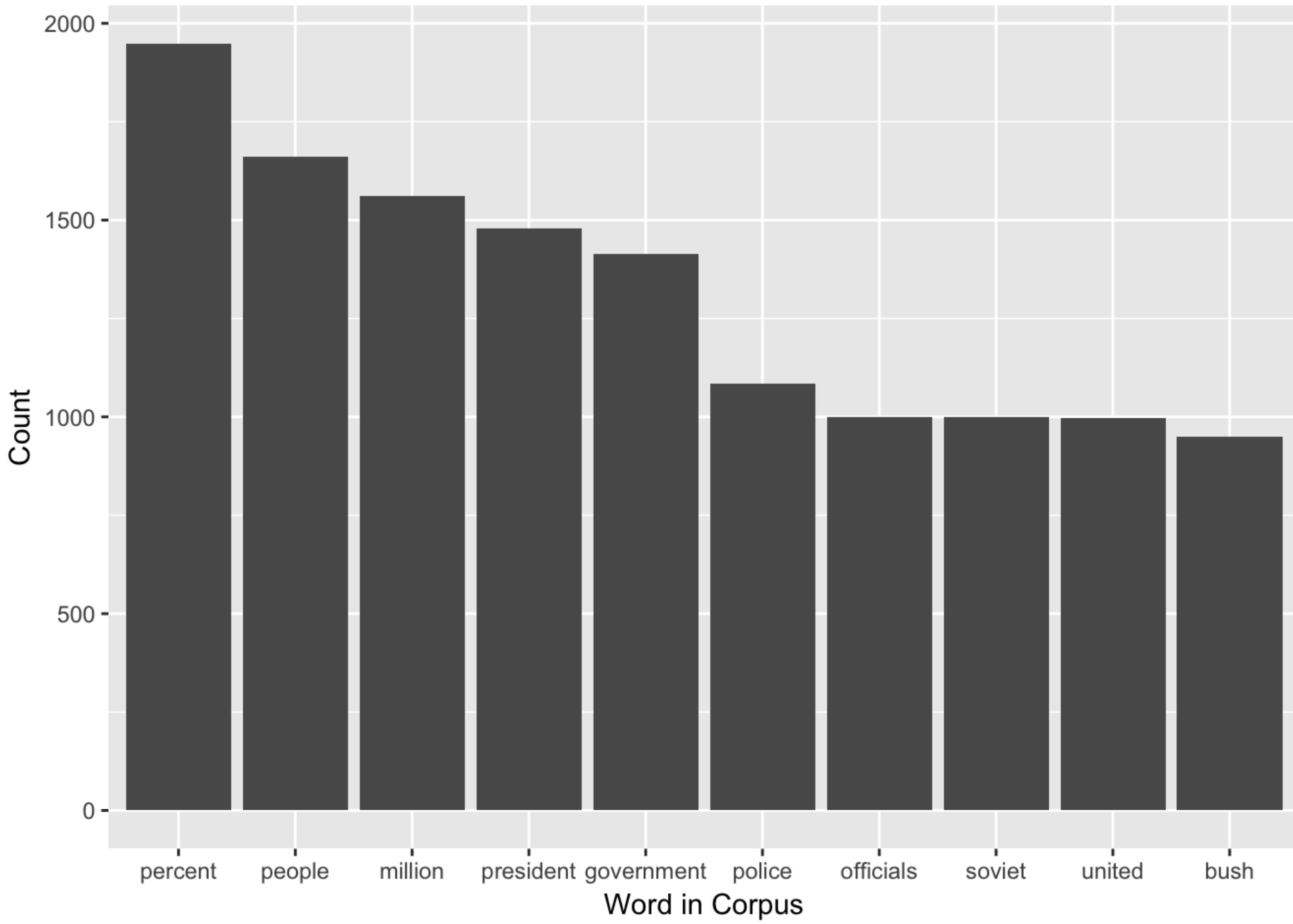
```
library(reshape2)
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
dfplot <- as.data.frame(melt(topten))
dfplot$word <- dimnames(dfplot)[[1]]
dfplot$word <- factor(dfplot$word,
                      levels=dfplot$word[order(dfplot$value,
                                               decreasing=TRUE)])

fig <- ggplot(dfplot, aes(x=word, y=value)) + geom_bar(stat="identity")
fig <- fig + xlab("Word in Corpus")
fig <- fig + ylab("Count")
print(fig)
```



Determine k number of topics

One aspect of LDA, is you need to know the k number of optimal topics for the documents.

k=4

Let us estimate an LDA model for the Associated Press articles, setting k=4.

Time to run the Model

To run the model, I am using the LDA function in the topicmodels package. You pass the document term matrix, optimal number of topics(k), a seed number if you want to be able to replicate the results.

```
library(topicmodels)
a_lda4 <- LDA(a_dtm, k = 4, control = list(seed = 11091988))
a_lda4
```

```
## A LDA_VEM topic model with 4 topics.
```

Let’s see what the top terms for each of these topics look like!

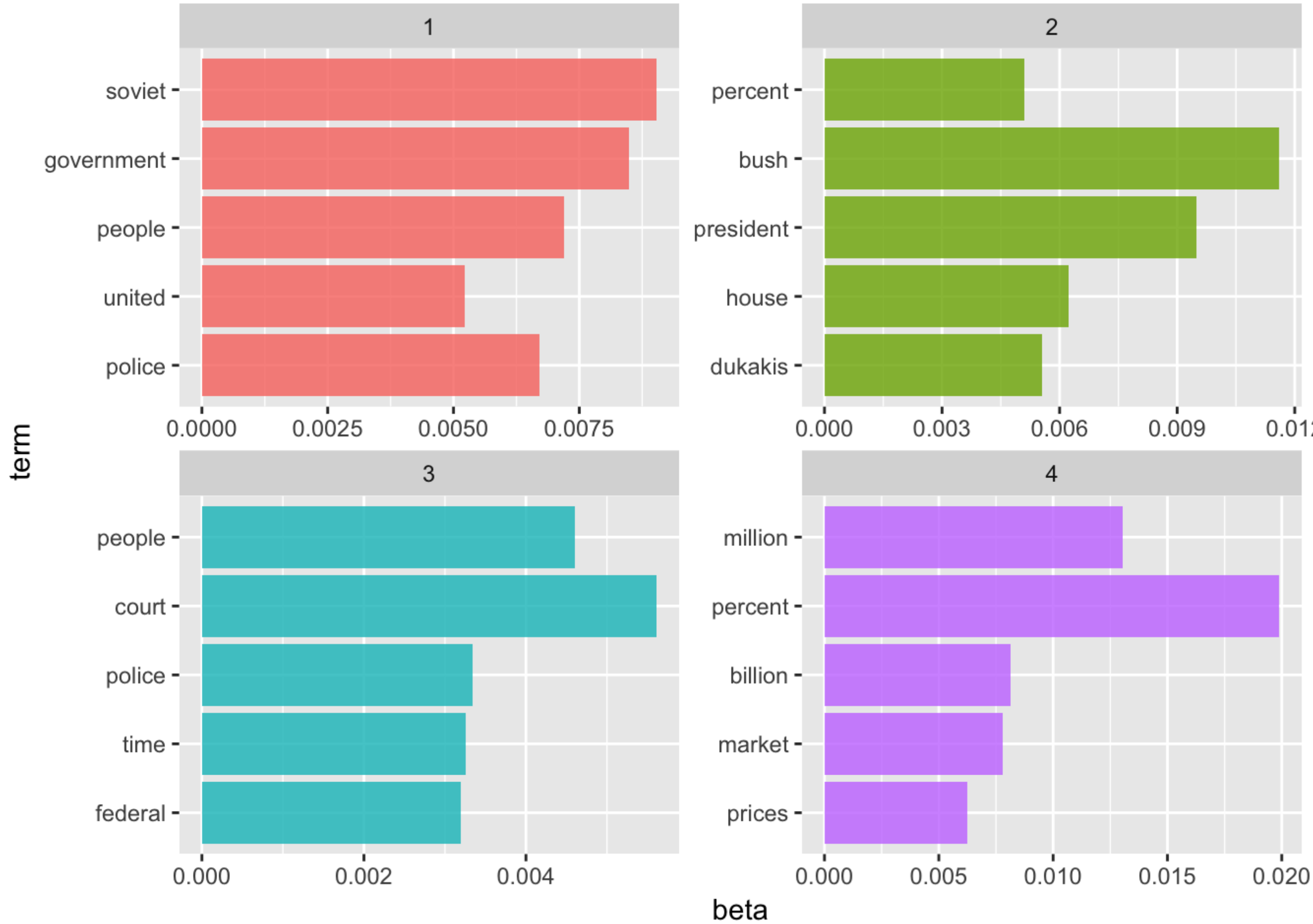
```
a_lda_td4 <- tidy(a_lda4)

top_terms4 <- a_lda_td4 %>%
  group_by(topic) %>%
  top_n(5, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
top_terms4
```

```
## # A tibble: 20 x 3
##    topic       term        beta
##    <int>      <chr>       <dbl>
##  1     1     soviet 0.009039278
##  2     1 government 0.008485694
##  3     1     people 0.007197636
##  4     1     police 0.006714883
##  5     1     united 0.005216493
##  6     2       bush 0.011606595
##  7     2  president 0.009498895
##  8     2      house 0.006230417
##  9     2    dukakis 0.005546771
## 10     2    percent 0.005091372
## 11     3      court 0.005614746
## 12     3     people 0.004601156
## 13     3     police 0.003336351
## 14     3       time 0.003252129
## 15     3    federal 0.003195206
## 16     4    percent 0.019893090
## 17     4    million 0.013024261
## 18     4    billion 0.008153693
## 19     4     market 0.007773469
## 20     4     prices 0.006233089
```

Let’s visualize to see what’s going on:

```
top_terms4 %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(alpha = 0.8, stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 2) +
  coord_flip()
```



k=10

Let us try a different number of topics this time and see what happens!

```
a_lda10 <- LDA(a_dtm, k = 10, control = list(seed = 11091987))
a_lda10
```

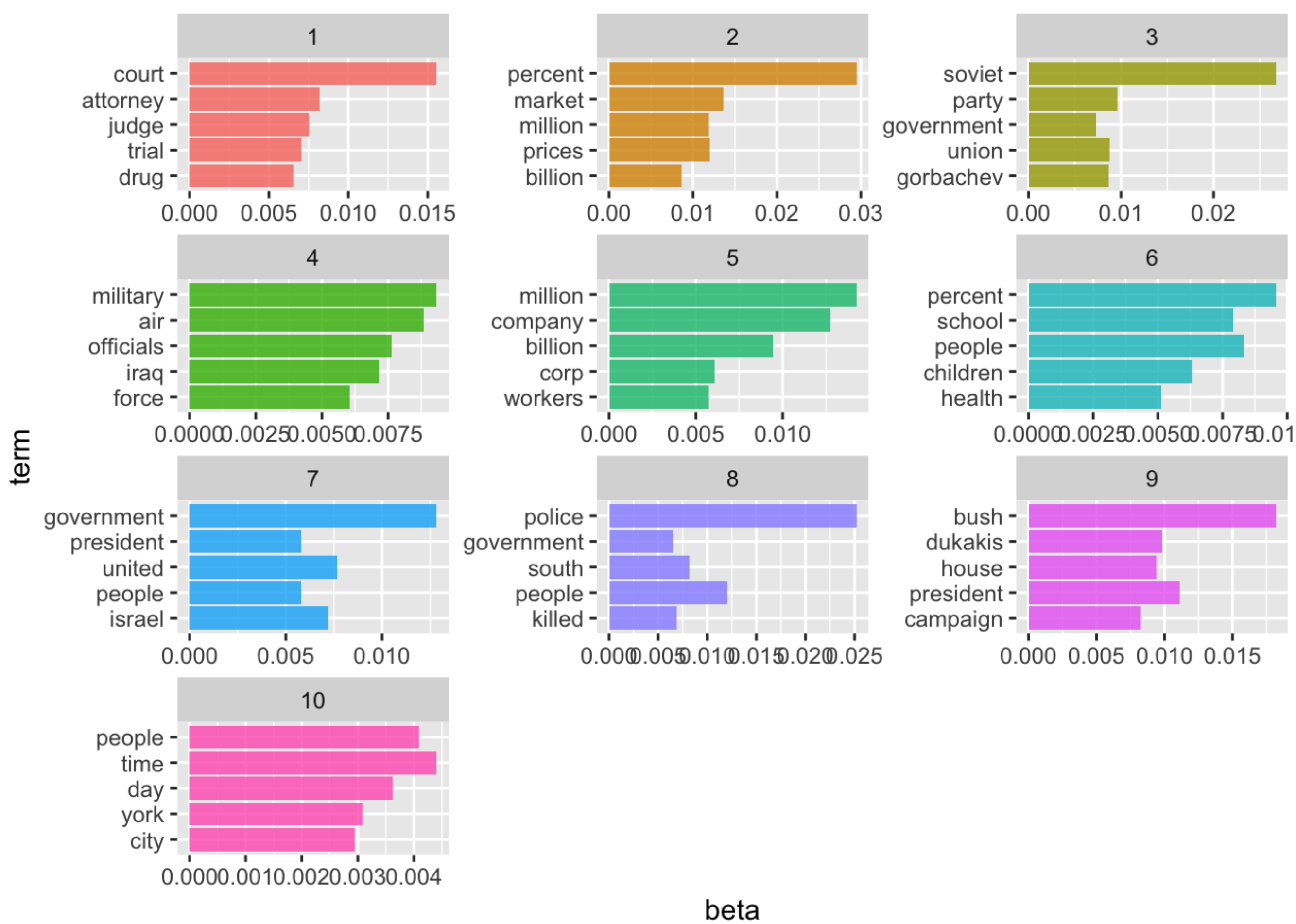
```
## A LDA_VEM topic model with 10 topics.
```

```
a_lda_td10 <- tidy(a_lda10)

top_terms10 <- a_lda_td10 %>%
  group_by(topic) %>%
  top_n(5, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
top_terms10
```

```
## # A tibble: 50 x 3
##    topic     term        beta
##    <int>    <chr>       <dbl>
##  1     1    court 0.015570374
##  2     1 attorney 0.008193595
##  3     1    judge 0.007498139
##  4     1    trial 0.007002726
##  5     1     drug 0.006562663
##  6     2  percent 0.029448754
##  7     2   market 0.013582254
##  8     2   prices 0.012014158
##  9     2  million 0.011900016
## 10     2  billion 0.008615622
## # ... with 40 more rows
```

```
top_terms10 %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(alpha = 0.8, stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 3) +
  coord_flip()
```

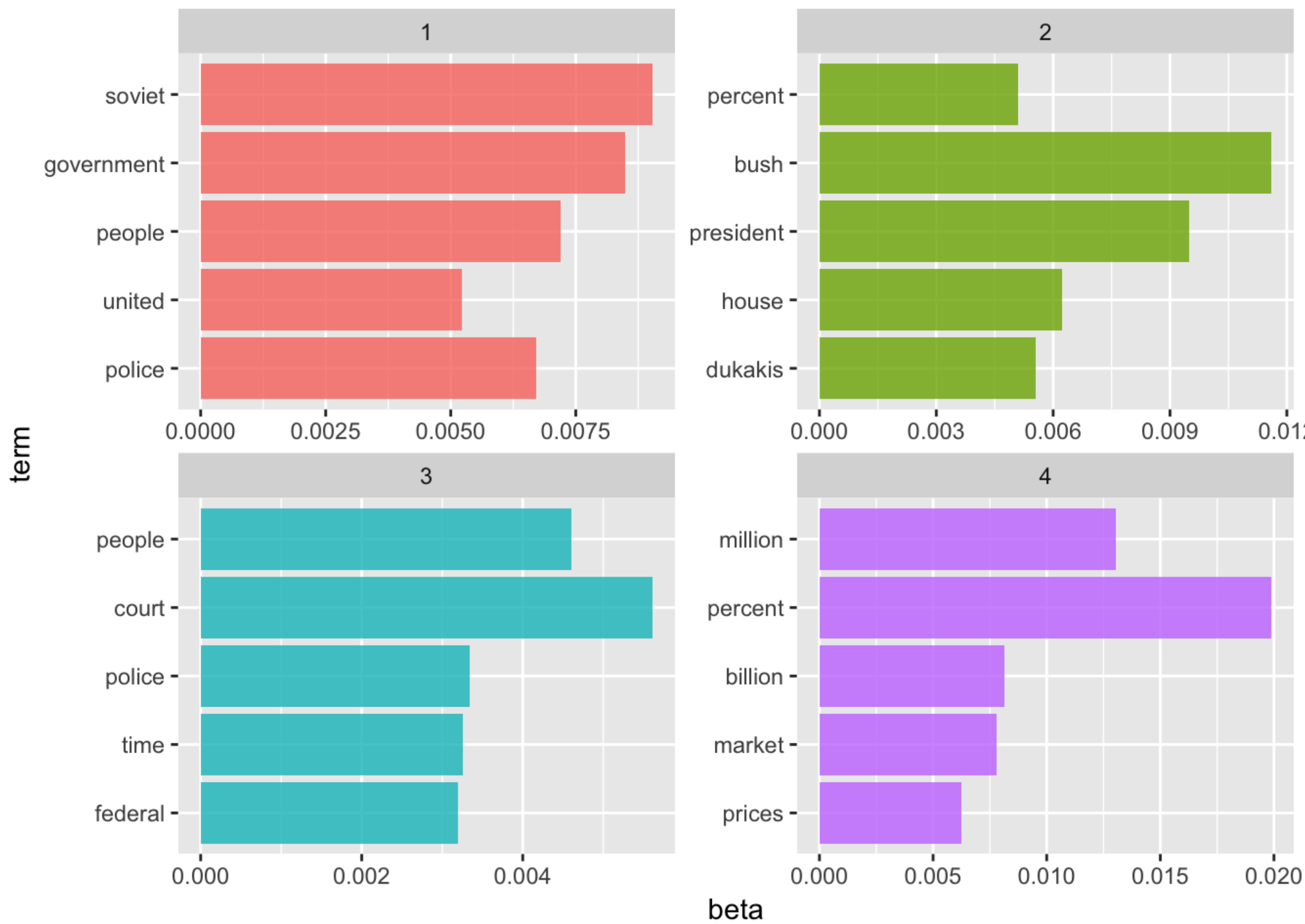


Summary

Increasing k we sacrifice clarity.Inorder to determine the optimal number of topics we also can use a statistical method- Perplexity. Perplexity is a statistical measure of how well a probability model predicts a sample. The benefit of this statistic comes in comparing perplexity across different models with varying ks. The model with the lowest perplexity is generally considered to be thes best

However, for the simplicity of this project I decide to stick with k=4 Let's visualize the 4 Topics again:

```
top_terms4 %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_bar(alpha = 0.8, stat = "identity", show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free", ncol = 2) +
  coord_flip()
```



My interpretation of these topics:

Topic 1 stands out to be about American-Soviet relations.

Topic 2 seems to be focussed on the Bush Administration discussions.

Topic 3 seems to be focused on crime and justice.

Topic 4 clearly relates to the economy.