MODULE 19

The module discusses the two topics covered in the Session 2 of the final class:

- Naive Bayes
- Topic Models

Naive Bayes

Lets say we have estimated the probability vectors for two corpora - w1, w2 where the documents in the two corpora were authored by two authors. The probability vector estimates for the two corpora are as given below:

Probability Vector Estimate for Corpus 1:

$$\hat{w}_1 = \left[\hat{w}_{11}, \hat{w}_{12} ... \hat{w}_{1D}\right]$$

Probability Vector Estimate for Corpus 2:

$$\hat{w}_2 = \left[\hat{w}_{21}, \hat{w}_{22} ... \hat{w}_{2D}\right]$$

We derive these probability estimates by drawing frequencies and/or by applying laplace smoothing discussed in the previous module.

Say X* is a test document (a bag of words representation) which we haven't used in the probability vector estimates above and X* is given as

$$X = \begin{bmatrix} X_1^* & \dots & X_D^* \end{bmatrix}$$

where 1 to D are the words in the document and X^* is a bag of words representation with X_1^* being the count of word 1 in test document to X_D^*, count of the Dth word in the document.

Each corpus has documents that are authored by a single author. To answer the question, from which of the two corpora did the document come from or who was the author of the document we use the Naive Bayes Model.

If the true generative process are 'the bag of words model' and if they belong to the first corpus then we can use the multinomial probability model to compute the probability as given below:

$$P(X^*|\hat{w}_1) = \frac{N^*!}{(X_1^*!X_2^*!...X_D^*!)}(\hat{w}_{11}^{X_1^*}.\hat{w}_{12}^{X_1^*}...\hat{w}_{1D}^{X_1^*})$$

where $\hat{w}_1$ is the probability vector estimate for corpus 1 and

$$N^* = \sum_{j=1}^{D} X_j^*$$

Similarly we can compute the probability/likelihood that the document came from the second corpus as:

$$P(X^*|\hat{w}_2) = \frac{N^*!}{(X_1^*!X_2^*!...X_D^*!)}(\hat{w}_{21}^{X_1^*}.\hat{w}_{22}^{X_1^*}...\hat{w}_{2D}^{X_1^*})$$

where $hatw_2$ is the probability vector estimate for corpus 2 and

$$N^* = \sum_{j=1}^{D} X_j^*$$

To decide on which corpora we use the Bayes Rule given below along with the probabilities estimated above.

For two hypothesis Y1 and Y2 and data X, the probability of Y1 given data (X) is given by:

$$P(Y_1|X) = P(Y_1) * P(X|Y_1)/P(X)$$

And similarly for Y2 given X:

$$P(Y_2|X) = P(Y_2) * P(X|Y_2)/P(X)$$

Two key points from the discussion so far:

The reason for using Bayes Rule is that Bayes rule allows us to compute prior probabilities using the Bayes Rule which is unaccounted for in the Multinomial Probability model. The Naive Bayes model is based on the assumption that the word count of the individual words in a document are independent.

We can simplify the model developed using the Multinomial Probability, we can ignore the following term from the comparison as they are common in both the probabilities:

$$\frac{N^*!}{(X_1^*!X_2^*!...X_D^*!)}$$

Also, while taking a product of small fractions, we often face a problem of numerical underflow. Numerical underflow is a condition in a computer program where the result of a calculation is a smaller number than the computer can actually store in memory. To avoid this we take the represent the probability in the log scale as below. Please note that the expression is also linear in parameters after taking the log.

$$\log(P(X^*|\hat{w}_1)) = \sum_{=1}^{k} X_j^* \log(\hat{w}_{ij})$$

R Script for Naive Bayes

Remember to source in the "readerPlain" wrapper function which is available in the tm.examples.R file and the code snippet as below:

```r
readerPlain = function(fname){
                readPlain(elem=list(content=readLines(fname)),
                            id=fname, language='en') }
```

The code below is used to roll the documents from the directory below into two corpora and each representing the documents written by two different authors - Aaron Pressman - (AP) and Alan Crosby (AC).

```r
library(tm)
```

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

```r
author_dirs = Sys.glob('F:/Predictive Modelling(JScott)/STA380-master/STA380-master/data/ReutersC50/C50t
author_dirs = author_dirs[1:2] # first two authors
file_list = NULL
labels = NULL
for(author in author_dirs) {
  author_name = substring(author, first=29)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  labels = append(labels, rep(author_name, length(files_to_add))) #  having labels of these vectors
}
```

The snippet below stores the contents of documents in the my_corpus variable and the code also retrieves the files names for each corpus and assigns it to the corresponding corpus.

```r
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

my_corpus = Corpus(VectorSource(all_docs))
names(my_corpus) = file_list # rather than name being 1:100 assign names to the document `
```

The data in the documents are preprocessed and the following transformations are made - make the document lowercase, remove numbers, remove punctutations, remove excess white-space and 'SMART' stopwords are removed.

```r
options(warn=-1)
my_corpus = tm_map(my_corpus, content_transformer(tolower)) # make everything lowercase
my_corpus = tm_map(my_corpus, content_transformer(removeNumbers)) # remove numbers
my_corpus = tm_map(my_corpus, content_transformer(removePunctuation)) # remove punctuation
my_corpus = tm_map(my_corpus, content_transformer(stripWhitespace)) ## remove excess white-space
my_corpus = tm_map(my_corpus, content_transformer(removeWords), stopwords("SMART"))

DTM = DocumentTermMatrix(my_corpus)
DTM # some basic summary statistics
```

```
## <<DocumentTermMatrix (documents: 100, terms: 4655)>>
## Non-/sparse entries: 16048/449452
## Sparsity           : 97%
## Maximal term length: 36
## Weighting          : term frequency (tf)
```

```r
class(DTM)  # a special kind of sparse matrix format
```

```
## [1] "DocumentTermMatrix"    "simple_triplet_matrix"
```

The summary shows 100 documents of two different authors having sparsity of 93 %

We remove the sparse terms below:

```r
options(warn=-1)
DTM = removeSparseTerms(DTM, 0.975)
DTM
```

```
## <<DocumentTermMatrix (documents: 100, terms: 1760)>>
## Non-/sparse entries: 12224/163776
## Sparsity           : 93%
## Maximal term length: 18
## Weighting          : term frequency (tf)
```

Creating a dense matrix from a sparse matrix as some function will not work on sparse matrix. The con side is that we are wasting memory space.

```
# Now a dense matrix
X = as.matrix(DTM)
```

We create a training set for each author using the first 45 docuents under each author.

```
# Naive Bayes
AP_train = X[1:45,] #5 as test set from each author
AC_train = X[51:95,] #5 as test set from each author
```

Creating the multinomial probability vector for each author Aaron Pressman - (AP) and Alan Crosby (AC) using the smoothing factor to account for the occurences of a word that is not available in training corpus.

```
smooth_count = 1/nrow(X)
w_AP = colSums(AP_train + smooth_count)
w_AP = w_AP/sum(w_AP)
```

```
# AC's multinomial probability vector
w_AC = colSums(AC_train + smooth_count)
w_AC = w_AC/sum(w_AC)
```

Choose a test document and check using the Naive Bayes model to determine whether it belongs to author Aaron Pressman - (AP) or Alan Crosby (AC).

The test document is chosen from the 5 documents that werent a part of the training documents.

```
# Let's take a specific test document
x_test = X[49,]
head(sort(x_test, decreasing=TRUE), 25)
```

```
##     foreign        cftc    exchange     trading    currency       court 
##          15          12          10          10           9           8 
##   amendment     supreme derivatives     futures     options    treasury 
##           7           6           5           5           5           5 
##       banks        case  commission    congress      courts      market 
##           4           4           4           4           4           4 
##      argued    decision        dunn       issue         law   oversight 
##           3           3           3           3           3           3 
##       trade 
##           3
```

```
# Compare log probabilities under the Naive Bayes model
sum(x_test*log(w_AP))
```

```
## [1] -1888.263
```

```
sum(x_test*log(w_AC))
```

```
## [1] -2253.32
```

```
# Another test document
x_test2 = X[99,]
head(sort(x_test2, decreasing=TRUE), 25)
```

```
##       week     market      index   thursday   analysts     points
##          9          8          6          6          5          5
##     prices     bourse bratislava      close  investors  ljubljana
##          5          4          4          4          4          4
##    percent     prague      stock      trend     warsaw  bucharest
##          4          4          4          4          4          3
##   budapest  coalition        end   exchange government   minister
##          3          3          3          3          3          3
##      prime
##          3
```

```
sum(x_test2*log(w_AP))
```

```
## [1] -2808.106
```

```
sum(x_test2*log(w_AC))
```

```
## [1] -2236.782
```

From the above we see that the log probabilities for the first test document is higher for author AP - indicating the first document could belong the to the author.

The second test has a higher log probability for author AC indicating it could belong to this author.

Topic Model

Topic model can be defined as a model of text built out of a mixture of superposition of simple bag of words models that can be used to determine the theme of the document. For example in articles related to Austin we will find topics such as football, barbeque which will have a set of words associated with them.

If we can take the analogy to documents, the components or constituents are the bag of words model and we use use them to create rich distributions to determine theme.

Bag of words can be regarded as set of word with thematic coherence and documents being a superposition of these bag of words.

We will explain the *topic model* with the two example below

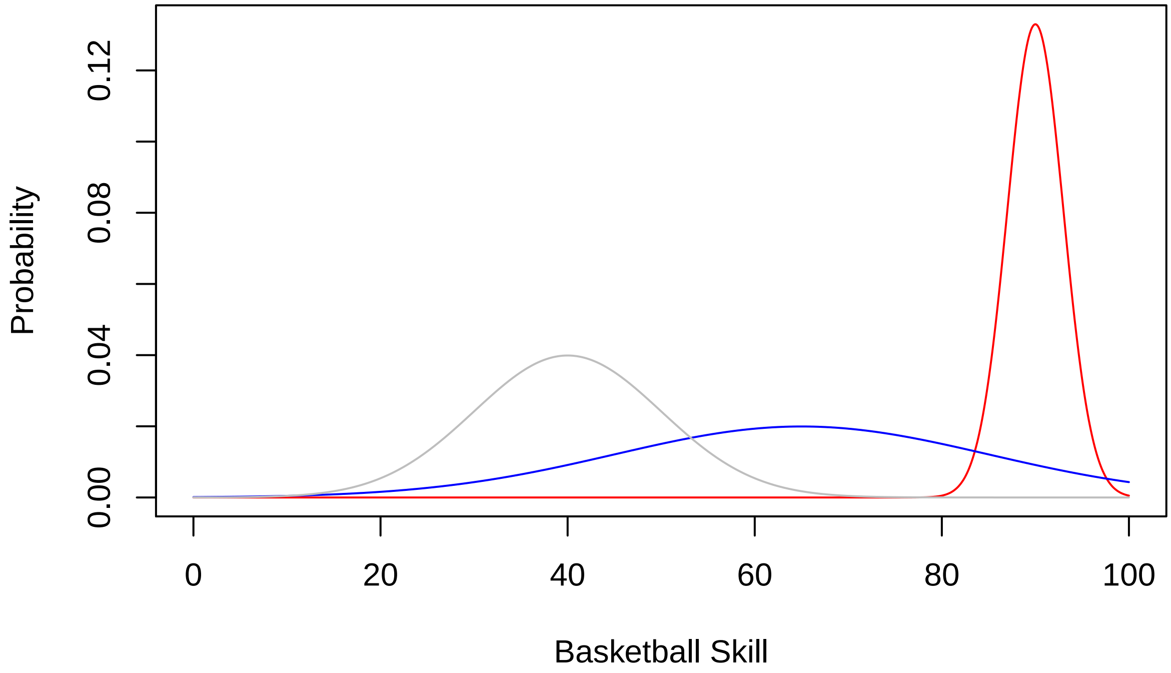
Simple_Mixture - An example

There are three kinds of basketball players having an arbitrary measure of their skills Topic modeling can be thought of as mixture modeling and in this case the three basketball players are mixture components. We are trying to form a basketball team based on distribution of each component. Hence, We will get a mixture component distribution based on distribution of individual components

```
#normally distributed around 40
mu1 = 40
sigma1 = 10
#normally distributed around 65
mu2 = 65
sigma2 = 20
#normally distributed around 90
mu3 = 90
sigma3 = 3
```

The three mixture components

```
curve(dnorm(x, mu3, sigma3), from=0, to=100, n=1001, col='red', xlab='Basketball Skill', ylab='Probabili
curve(dnorm(x, mu2, sigma2), from=0, to=100, n=1001, col='blue', add=TRUE)
curve(dnorm(x, mu1, sigma1), from=0, to=100, n=1001, col='grey', add=TRUE)
```



The weights for each class of basketball players are defined as 30% ,60% , 10%. The combination of the three class of players gives a basketball team which has its own distribution

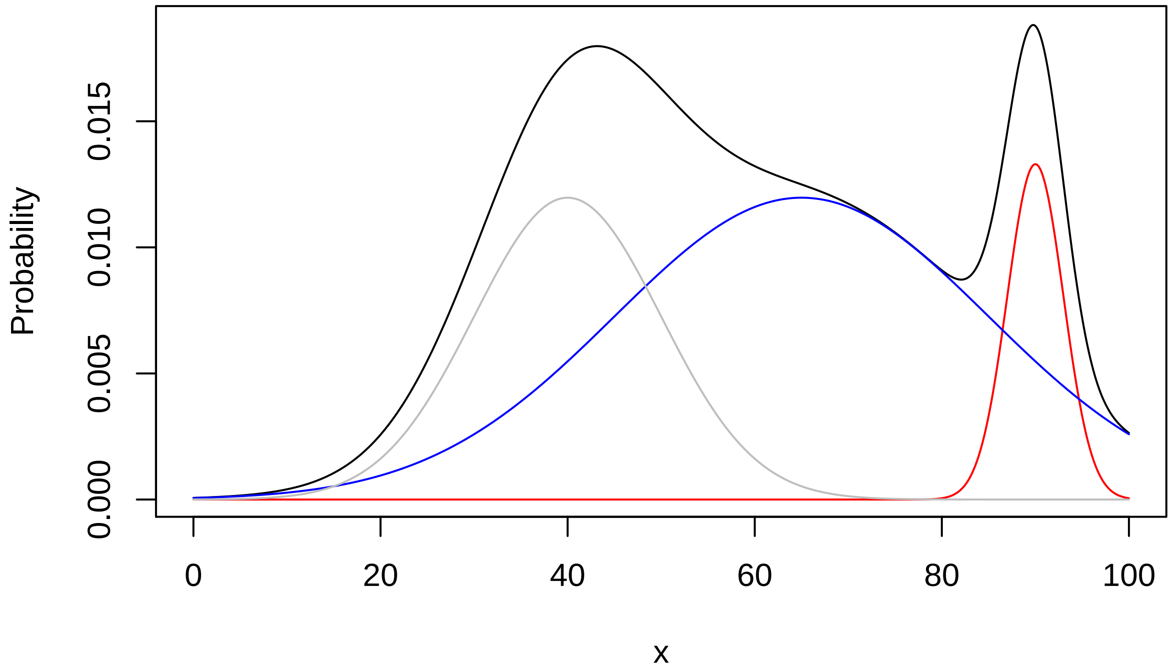
Each point in the basketball team is a combination of weights of points of individual class of basketball players. Same goes for area under the curve for basketball team.

Based on the graph we can see that we get a rich distribution of mixture(basketball team) based on the collection of small constituents(Class of basketball players)

```
weights = c(0.3, 0.6, 0.1)

curve(weights[1]*dnorm(x, mu1, sigma1) + weights[2]*dnorm(x, mu2, sigma2)  + weights[3]*dnorm(x, mu3, s
      from=0, to=100, ylab='Probability', n=1001)

curve(weights[3]*dnorm(x, mu3, sigma3), col='red', n=1001, add=TRUE)
curve(weights[2]*dnorm(x, mu2, sigma2), from=0, to=100, n=1001, col='blue', add=TRUE)
curve(weights[1]*dnorm(x, mu1, sigma1), from=0, to=100, n=1001, col='grey', add=TRUE)
```



The Congressmen - Example-2

The second example is the one with the congressman that had been discussed as a part of the PCA analysis. The dataset used here is a processed document term matrix from the original congressmen dataset.

```
library(maptpx) # Posterior maximization for topic models
```

```
## Loading required package: slam
```

```r
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```r
# both files are document term matrix
countdata = read.csv("F:/Predictive Modelling(JScott)/STA380-master/STA380-master/data/congress109.csv"
memberdata = read.csv("F:/Predictive Modelling(JScott)/STA380-master/STA380-master/data/congress109membe
```

Like kmeans clustering, here we would have to predefine number of topics

```r
tm1 = topics(countdata, 10)
```

```
##
## Estimating on a 529 document collection.
## Fitting the 10 topic model.
## log posterior increase: 7802.8, 1806, 682.3, 191.2, done.
```

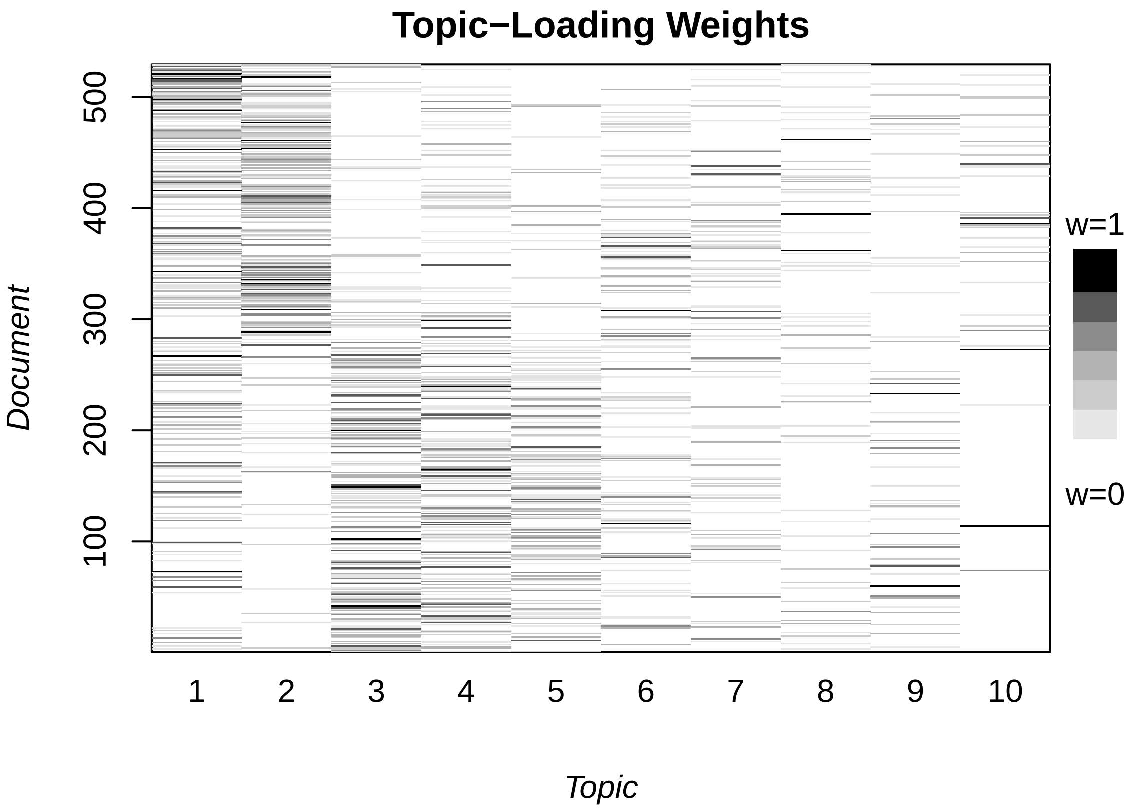
```r
summary(tm1)
```

```
##
## Top 5 phrases by topic-over-null term lift (and usage %):
##
## [1] 'ryan.white.care', 'white.care.act', 'victim.domestic', 'victim.domestic.violence', 'african.amer
## [2] 'united.airline.employe', 'cut.child.support', 'fund.tax.cut', 'student.loan.cut', 'permanent.ta
## [3] 'near.retirement.age', 'death.tax', 'repeal.death.tax', 'medic.liability.crisi', 'increase.taxe'
## [4] 'commonly.prescribed.drug', 'near.earth.object', 'iwo.jima', 'southeast.texa', 'postal.service'
## [5] 'ready.mixed.concrete', 'indian.art.craft', 'date.time', 'assistant.secretary.commerce', 'amnesty
## [6] 'low.cost.reliable', 'wild.bird', 'suppli.natural.ga', 'driver.education', 'fuel.efficiency.stan
## [7] 'credit.card.industry', 'increase.minimum.wage', 'supreme.court.rejected', 'protect.minority.rig
## [8] 'american.fre.trade', 'central.american.fre', 'north.american.fre', 'financial.accounting.standa
## [9] 'regional.training.cent', 'national.ad.campaign', 'pluripotent.stem.cel', 'cel.stem.cel', 'produ
## [10] 'able.buy.gun', 'change.heart.mind', 'gun.safety.law', 'caliber.sniper.rifle', 'hate.crime.legi
##
## Dispersion = 2.99
```

From the summary we would find not the frequent words but the most surprising one with the highest lift in comparison to their baseline probability

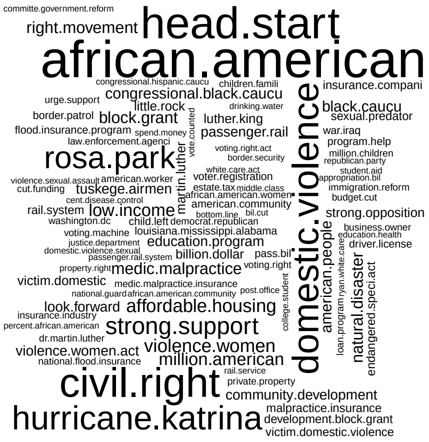
Plotting the weight of each topic in each document using 'barcode graph'.

```r
plot(tm1)
```

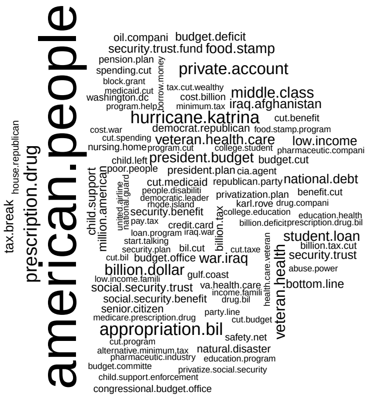


Analyze the tokens used with the help of wordcloud.

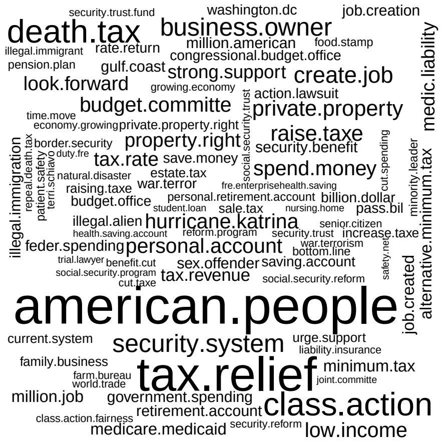
```
options(warn=-1)
wordcloud(colnames(countdata), tm1$theta[,1], min.freq=0, max.words=100)
```



```r
wordcloud(colnames(countdata), tm1$theta[,2], min.freq=0, max.words=100)
```



```r
wordcloud(colnames(countdata), tm1$theta[,3], min.freq=0, max.words=100)
```



Selecting K

```r
tm_select = topics(countdata, K=c(5:15))
```

```
##
## Estimating on a 529 document collection.
## Fit and Bayes Factor Estimation for K = 5 ... 15
## log posterior increase: 6251.1, 1240.2, 504.8, 395.4, done.
## log BF( 5 ) = 57506.56
## log posterior increase: 2914.3, done.
## log BF( 6 ) = 67701.89
## log posterior increase: 2220.5, 157, done.
## log BF( 7 ) = 69245.55
## log posterior increase: 1823, 180.4, 150.7, 116.6, done.
## log BF( 8 ) = 71999.82
## log posterior increase: 1980.7, done.
## log BF( 9 ) = 78576.42
## log posterior increase: 1602.8, 143.7, 84.6, done.
## log BF( 10 ) = 76964.33
## log posterior increase: 1638.3, 210.9, 210, done.
## log BF( 11 ) = 80168.42
## log posterior increase: 1464.6, 167.6, done.
## log BF( 12 ) = 83035.33
## log posterior increase: 1321.6, 88.7, done.
## log BF( 13 ) = 84182.02
## log posterior increase: 1332.1, 111.6, done.
## log BF( 14 ) = 82032.79
## log posterior increase: 1086.9, 52.9, done.
## log BF( 15 ) = 83306.13
```

```
summary(tm_select)
```

```
##
## Top 5 phrases by topic-over-null term lift (and usage %):
##
## [1] 'national.homeownership.month', 'southeast.texa', 'million.illegal.alien', 'temporary.worker.prog
## [2] 'national.heritage.corridor', 'asian.pacific.american', 'american.heritage.month', 'pacific.ameri
## [3] 'united.airline.employe', 'spending.cut.bil', 'issue.facing.american', 'student.loan.cut', 'outin
## [4] 'near.retirement.age', 'medic.liability.crisi', 'increase.taxe', 'gifted.talented.student', 'tax
## [5] 'winning.war.iraq', 'near.earth.object', 'troop.bring.home', 'bring.troop.home', 'bless.america'
## [6] 'wild.bird', 'ready.mixed.concrete', 'witness.testify', 'timber.sale', 'president.announce' (8)
## [7] 'columbia.river.gorge', 'passenger.rail.service', 'postal.service', 'reverse.robin.hood', 'rail.
## [8] 'judicial.confirmation.process', 'chief.justice.rehnquist', 'john.robert', 'justice.janice.roger
## [9] 'western.energy.crisi', 'republic.cypru', 'violent.sexual.predator', 'sole.source.contract', 'se
## [10] 'pluripotent.stem.cel', 'national.ad.campaign', 'cel.stem.cel', 'stem.cel.line', 'regional.trai
## [11] 'increase.minimum.wage', 'credit.card.industry', 'dropout.prevention.program', 'raise.minimum.wa
## [12] 'north.american.fre', 'financial.accounting.standard', 'american.fre.trade', 'central.american.
## [13] 'able.buy.gun', 'hate.crime.legislation', 'hate.crime.law', 'driver.education', 'global.gag.rul
##
## Log Bayes factor and estimated dispersion, by number of topics:
##
##              5        6        7        8        9       10       11
## logBF 57506.56 67701.89 69245.55 71999.82 78576.42 76964.33 80168.42
## Disp      3.65     3.30     3.11     3.09     2.94     2.84     2.77
##             12       13       14       15
## logBF 83035.33 84182.02 82032.79 83306.13
## Disp      2.66     2.53     2.48     2.40
##
## Selected the K = 13 topic model
```