Assignment 5

Statistics and Data Science 365/565

Due: October 23 (before 11:59 pm)

Question 1: Probabilities for topic modeling (10 points)

Consider a latent Dirichlet allocation topic model with $K = 2$ topics. Given the topics β_k, compute in closed form the likelihood of the 3-word document $d =$ `"seize the data"` assuming each word is in the vocabulary. That is, compute the probability:

$$p(W_{d,1} = seize, W_{d,2} = the, W_{d,3} = data \mid \beta_{1:2}, \alpha)$$

Solution:

We know that the following is how you calculate this probablitiy value:

$$p(W \mid \beta_{1:2}, \alpha, \alpha) = \int p(\theta \mid \alpha, \alpha) * \prod_{n=1}^{N} \sum_{Z_n} p(z_n \mid \theta) p(w_n \mid z_n, \beta_{1:2}) d\theta$$

Here, we have 2 topics aand 3 words in our document. With two topics,$p(z_1 \mid \theta) = \theta$ and $p(z_2 \mid \theta) = 1 - \theta$ and the α values are equal. Additionally, $p(w_n \mid z_n, \beta_{1:2})$ is equal to β_{1,w_n} and β_{2,w_n} for each value of β. Then we just multiply for each word. So we get:

$$\int p(\theta \mid \alpha, \alpha) * \prod_{n=1}^{N} (\theta\beta_{1,w_n} + (1-\theta)\beta_{2,w_n}) d\theta =$$

$$\int p(\theta \mid \alpha, \alpha) * (\theta\beta_{1,seize} + (1-\theta)\beta_{2,seize}) * (\theta\beta_{1,the} + (1-\theta)\beta_{2,the}) * (\theta\beta_{1,data} + (1-\theta)\beta_{2,data}) d\theta$$

By multiplying out these terms, we get the following:

$$\int p(\theta \mid \alpha, \alpha) * \big[(\theta^3 \beta_{1,seize}\beta_{1,the}\beta_{1,data} + \theta^2(1-\theta)\beta_{1,seize}\beta_{1,the}\beta_{2,data} + \theta^2(1-\theta)\beta_{1,seize}\beta_{2,the}\beta_{1,data} +$$

$$\theta^2(1-\theta)\beta_{2,seize}\beta_{1,the}\beta_{1,data} + \theta(1-\theta)^2\beta_{1,seize}\beta_{2,the}\beta_{2,data} + \theta(1-\theta)^2\beta_{2,seize}\beta_{2,the}\beta_{1,data} +$$
$$\theta(1-\theta)^2\beta_{2,seize}\beta_{1,the}\beta_{2,data} + ((1-\theta)^3\beta_{2,seize}\beta_{2,the}\beta_{2,data}\big] d\theta$$

Note that when we multiply $p(\theta \mid \alpha)$ (which is a β distribution) across all the θ and $(1-\theta)$ terms we get the integral of the probability density function of another β function:

$$\int p(\theta \mid \alpha, \alpha) * \theta^{\alpha'} (1-\theta)^{\alpha''} d\theta = \int \frac{\Gamma(\alpha+\alpha)}{\Gamma(\alpha)\Gamma(\alpha)} \theta^{\alpha-1} (1-\theta)^{\alpha} \theta^{\alpha'} (1-\theta)^{\alpha''} d\theta = \frac{\Gamma(\alpha+\alpha)}{\Gamma(\alpha)\Gamma(\alpha)} \int \theta^{\alpha+\alpha'-1} (1-\theta)^{\alpha+\alpha''-1} d\theta$$

$$= \frac{\Gamma(\alpha+\alpha)}{\Gamma(\alpha)\Gamma(\alpha)} * \frac{\Gamma(\alpha+\alpha')\Gamma(\alpha+\alpha'')}{\Gamma(2\alpha+\alpha'+\alpha'')}$$

Where α' and α'' are the values for each term inside the expanded integral above. This also comes from the integration of the pdf of the β function. These functions of the gamma function will be the coefficients for each of the β terms in the previous integrals. Now, let's calculate the coefficients for each of these terms:

First for θ^3, $\alpha' = 3$ and $\alpha'' = 0$. And for $(1-\theta)^3$, $\alpha' = 0$ and $\alpha'' = 3$. This means that either topic one or topic two was choosen three times. We can see these are equivalent and solve for the coefficents:

$$\frac{\Gamma(\alpha+\alpha')\Gamma(\alpha+\alpha'')}{\Gamma(2\alpha+\alpha'+\alpha'')} = \frac{\Gamma(\alpha+3)\Gamma(\alpha+0)}{\Gamma(2\alpha+0+3)} = \frac{\Gamma(\alpha+3)\Gamma(\alpha)}{\Gamma(2\alpha+3)}$$

Next, we want to look at instances where one of the topics is only chosen for one of the three words. This is what we see for all of the middle terms. For the first three, $\alpha' = 2$ and $\alpha'' = 1$. For the second three, $\alpha' = 1$ and $\alpha'' = 2$. Again, we see these are equal and can solve for the coeffecients:

$$\frac{\Gamma(\alpha+\alpha')\Gamma(\alpha+\alpha'')}{\Gamma(2\alpha+\alpha'+\alpha'')} = \frac{\Gamma(\alpha+2)\Gamma(\alpha+1)}{\Gamma(2\alpha+2+1)} = \frac{\Gamma(\alpha+2)\Gamma(\alpha+1)}{\Gamma(2\alpha+3)}$$

With all of this, we can factor out the terms from the integral and write the expression for the entire probabilty:

$$p(W \mid \beta_{1:2}, \alpha) = \frac{\Gamma(\alpha+\alpha)}{\Gamma(\alpha)\Gamma(\alpha)} \frac{\Gamma(\alpha+2)\Gamma(\alpha+1)}{\Gamma(2\alpha+3)} * (\beta_{1,seize}\beta_{1,the}\beta_{2,data} + \beta_{1,seize}\beta_{2,the}\beta_{1,data} + \beta_{2,seize}\beta_{1,the}\beta_{1,data} +$$

$$\beta_{1,seize}\beta_{2,the}\beta_{2,data} + \beta_{2,seize}\beta_{2,the}\beta_{1,data} + \beta_{2,seize}\beta_{1,the}\beta_{2,data})$$

$$+ \frac{\Gamma(\alpha+\alpha)}{\Gamma(\alpha)\Gamma(\alpha)} \frac{\Gamma(\alpha+3)\Gamma(\alpha)}{\Gamma(2\alpha+3)} * (\beta_{1,seize}\beta_{1,the}\beta_{1,data} + \beta_{2,seize}\beta_{2,the}\beta_{2,data})$$

In the following problems, you will model the statistics and machine learning repository of the online question and answer site "StackExchange," called "CrossValidated."

Question 2: Topic modeling of CrossValidated (50 points)

Our data were taken from the December 15, 2016 Stack Exchange data dump[1]. You will find two files,

```
stackexchange/20161215StatsPostsRaw.csv
stackexchange/20161215StatsPostsMerged.csv
```

The cleaned file has 92,335 documents, created by combining questions and associated answers, then removing HTML, LaTeX, code, and stopwords. See the `README` file for further details. You can use the raw data file if you wish to, but it's not necessary.

Here is part of an entry from the cleaned up version of the collection:

```
124,"Statistical classification of text I'm a programmer without
statistical background, and I'm currently looking at different
classification methods for a large number of different documents that
I want to classify into pre-defined categories. I've been reading
about kNN, SVM and NN. However, I have some trouble getting
started. What resources do you recommend? I do know single variable
and multi variable calculus quite well, so my math should be strong
enough. I also own Bishop's book on Neural Networks, but it has proven
to be a bit dense as an introduction. [...]
```

For the entirety of this problem use only the first 200 words of each document, so as to avoid potential issues with memory on your computer.

a. Process the data to determine a word vocabulary. You should get a vocabulary of size around 10,000 words or so—it's up to you to decide. You will need to write a parser that maps each entry to a sequence of word-id/count pairs. Describe the steps you take to process the data and the criteria you use to select the vocabulary.

{For this part (only) you may share code with others. Please post any code that is shared to Piazza, so that it is available to everyone in the class.} Code sharing is being allowed here because this kind of text processing is more tedious in R than in some other languages, such as Python, and it's a bit separate from the main focus on ML.

```
## Parsed with column specification:
## cols(
##   Id = col_integer(),
##   CleanBody = col_character()
## )
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 105 parsing failures.
## row # A tibble: 5 x 5 col     row col   expected               actual file
## ... .................. ... ...........................................................................
## See problems(...) for more details.
```

```
set.seed(1)
## Get first 200 words
data[, 2] <- apply(data[, 2], 1, function(x) word(x, start = 1,
    end = min(200, length(strsplit(as.character(x), split = " ")[[1]])),
    sep = fixed(" ")))
```

[1] Licensed under Creative Commons Share Alike 3.0, https://creativecommons.org/licenses/by-sa/3.0/

```
# Sample training/test data separation
train_indices <- sample(1:nrow(data), round(nrow(data) * 0.9),
    replace = FALSE)
train <- data[train_indices, ]
test <- data[-c(train_indices), ]

tokens <- train$CleanBody %>% tolower %>% word_tokenizer
it <- itoken(tokens, ids = train$Id, progressbar = FALSE)

#Here we are just filtering the vocabulary.
v <-  create_vocabulary(it, stopwords = as.vector(stop_words$word) ) %>% #filter stop words
  prune_vocabulary(term_count_min = 10) #filter for word count less than 10

#keep words with no digits
v <- v[!str_detect(v$term,"\\d"),]
#remove most punctuation
v <- v[!str_detect(v$term,"[\\. \\_ \\% \\& \\^ \\! \\* \\: \\; \\? \\< \\>]"),]
#keep words with greater than 3 characters in the word
#I imagine we don't have many words of value here.
v <- v[nchar(v$term) > 3,]

vectorizer <- vocab_vectorizer(v)
dtm <- create_dtm(it, vectorizer, type = "dgTMatrix")
```

b. Now fit topic models on the collection. Divide the corpus into training and validation documents–use a 90%/10% split, holding out about 9,000 documents.

You may use the LDA implementation in the library `topicmodels` *in R. The following resources may be helpful:*

https://goo.gl/6xLoky

http://tidytextmining.com/topicmodeling.html

Train topic models using different numbers of topics; a good starting point would be around 30 topics. Display the top 10 or so words (in decreasing order of probability β_{kw}) in each topic. Comment on the "meaning" or interpretation of several of the topics.

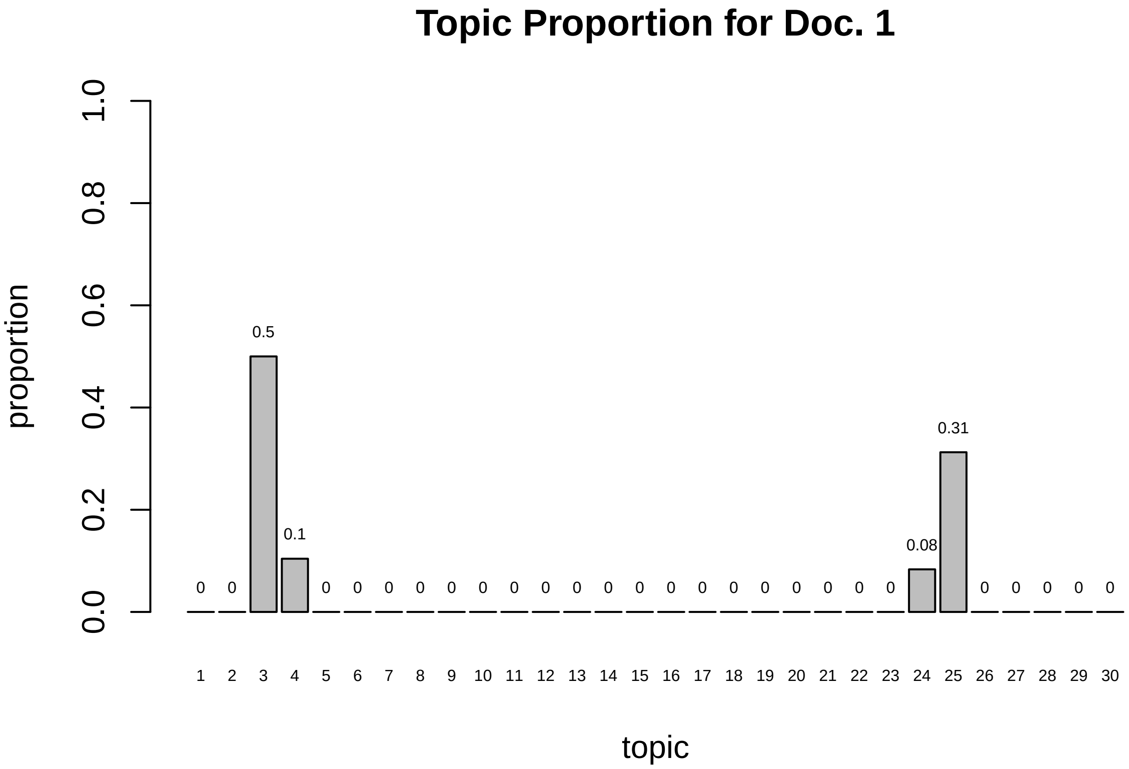
Select several documents, and display the most probable topics for each of them (according to the posterior distribution over θ). Do the assigned topics make sense? Comment on your findings.

You will need to read the documentation for the implementation that you choose (for example `topicmodels`*), to learn how to carry out these steps.*

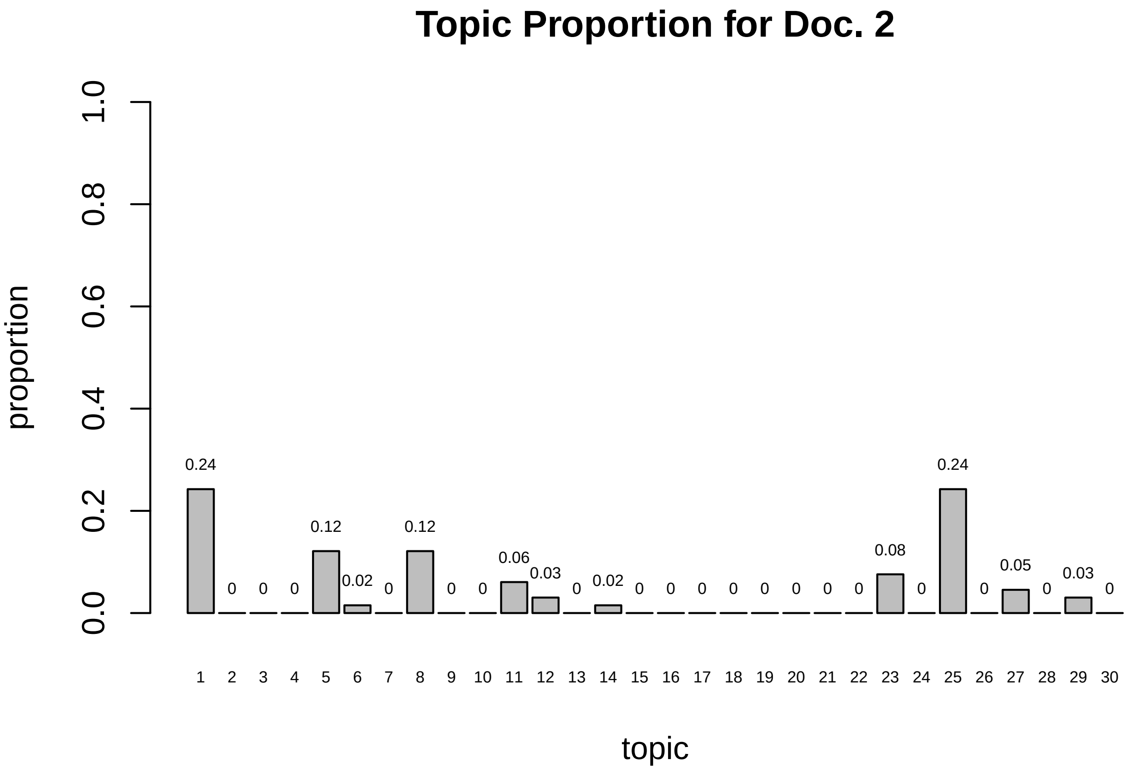
```
set.seed(1)

lda_model <- LDA$new(n_topics = 30, doc_topic_prior = 0.1, topic_word_prior = 0.01)
doc_topic_distr <- lda_model$fit_transform(x = dtm, n_iter = 5000,
    convergence_tol = 0.001, n_check_convergence = 5, progressbar = FALSE)
```

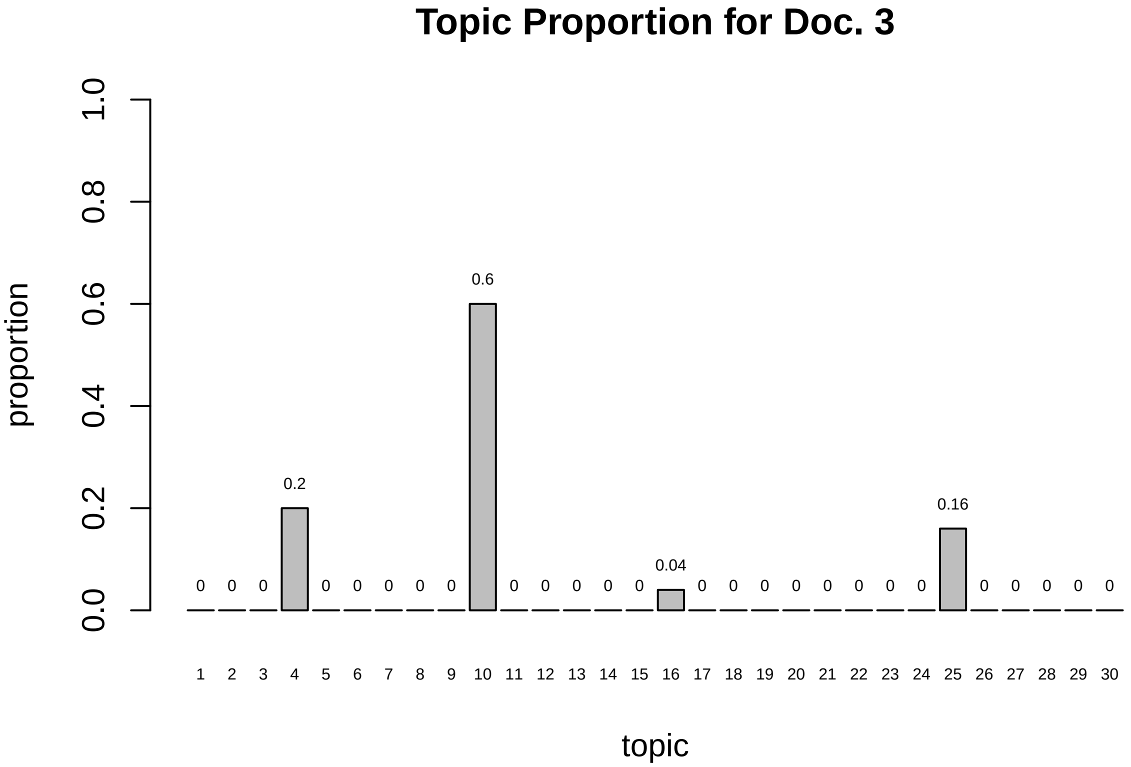
```
##      [,1]        [,2]         [,3]          [,4]            [,5]
##  [1,] "plot"      "distance"   "time"        "distribution"  "features"
##  [2,] "data"      "clustering" "series"      "probability"   "feature"
##  [3,] "line"      "cluster"    "data"        "random"        "data"
##  [4,] "graph"     "means"      "model"       "function"      "dataset"
##  [5,] "outliers"  "clusters"   "forecast"    "variable"      "selection"
##  [6,] "plots"     "data"       "arima"       "theorem"       "words"
##  [7,] "axis"      "based"      "forecasting" "distributions" "word"
##  [8,] "curve"     "similarity" "period"      "question"      "text"
##  [9,] "values"    "score"      "price"       "median"        "document"
## [10,] "histogram" "measure"    "month"       "continuous"    "subset"
##       [,6]           [,7]         [,8]             [,9]
##  [1,] "bayesian"     "treatment"  "data"           "function"
##  [2,] "prior"        "data"       "values"         "likelihood"
##  [3,] "probability"  "control"    "missing"        "parameters"
##  [4,] "distribution" "experiment" "variables"      "maximum"
##  [5,] "model"        "time"       "dataset"        "equation"
##  [6,] "ratio"        "test"       "variable"       "algorithm"
##  [7,] "binomial"     "design"     "scale"          "parameter"
##  [8,] "likelihood"   "subjects"   "transformation" "form"
##  [9,] "posterior"    "subject"    "analysis"       "solution"
## [10,] "event"        "repeated"   "column"         "understand"
##       [,10]           [,11]        [,12]       [,13]
##  [1,] "probability"   "model"      "data"      "correlation"
##  [2,] "random"        "regression" "score"     "variables"
##  [3,] "distribution"  "models"     "students"  "correlated"
##  [4,] "conditional"   "logistic"   "average"   "variable"
##  [5,] "density"       "data"       "student"   "coefficient"
##  [6,] "probabilities" "linear"     "question"  "relationship"
##  [7,] "markov"        "package"    "rate"      "correlations"
##  [8,] "sequence"      "parameters" "company"   "independent"
##  [9,] "variables"     "selection"  "customers" "significant"
## [10,] "question"      "function"   "amount"    "values"
##       [,14]            [,15]         [,16]      [,17]        [,18]
##  [1,] "class"          "variable"    "factor"   "variance"   "effect"
##  [2,] "classification" "variables"   "analysis" "estimate"   "variance"
##  [3,] "data"           "model"       "anova"    "estimator"  "analysis"
##  [4,] "classifier"     "regression"  "factors"  "covariance" "measure"
##  [5,] "accuracy"       "effect"      "scale"    "matrix"     "difference"
##  [6,] "random"         "categorical" "scores"   "error"      "standard"
##  [7,] "classes"        "effects"     "items"    "estimation" "error"
##  [8,] "tree"           "dependent"   "spss"     "parameter"  "data"
##  [9,] "training"       "interaction" "measures" "parameters" "measurement"
## [10,] "decision"       "continuous"  "score"    "estimates"  "meta"
##       [,19]          [,20]             [,21]         [,22]
##  [1,] "regression"   "time"            "statistics"  "training"
##  [2,] "linear"       "data"            "book"        "validation"
##  [3,] "model"        "model"           "package"     "cross"
##  [4,] "variable"     "series"          "code"        "network"
##  [5,] "coefficients" "effects"         "statistical" "neural"
##  [6,] "residuals"    "stationary"      "analysis"    "output"
##  [7,] "coefficient"  "panel"           "software"    "input"
##  [8,] "variables"    "fixed"           "read"        "data"
##  [9,] "error"        "autocorrelation" "python"      "error"
## [10,] "equation"     "random"          "found"       "layer"
##       [,23]        [,24]        [,25]           [,26]
##  [1,] "values"     "sample"     "distribution"  "test"
##  [2,] "confidence" "size"       "normal"        "hypothesis"
##  [3,] "interval"   "standard"   "random"        "tests"
##  [4,] "function"   "population" "distributed"   "null"
##  [5,] "intervals"  "samples"    "distributions" "testing"
##  [6,] "code"       "deviation"  "gaussian"      "significant"
##  [7,] "data"         "sampling"    "variance"       "difference"
##  [8,] "calculate"    "error"       "variables"      "data"
##  [9,] "error"        "power"       "multivariate"   "significance"
## [10,] "package"      "calculate"   "variable"       "statistic"
##       [,27]          [,28]         [,29]          [,30]
##  [1,] "matrix"       "people"      "analysis"     "learning"
##  [2,] "vector"       "probability" "question"     "machine"
##  [3,] "vectors"      "question"    "data"         "algorithm"
##  [4,] "data"         "person"      "methods"      "data"
##  [5,] "space"        "patients"    "method"       "algorithms"
##  [6,] "components"   "survey"      "paper"        "methods"
##  [7,] "kernel"       "outcome"     "statistical"  "based"
##  [8,] "component"    "disease"     "statistics"   "techniques"
##  [9,] "dimensional"  "risk"        "answer"       "system"
## [10,] "principal"    "population"  "research"     "mining"
```



```
## [1] "M/GI/inf queue in stationary distribution, how to get queue size distribution at the arrival ti
```



```
## [1] "Outlier detection with ROBPCA for multivariate poisson/non-normal data It is stated here[1] tha
```



```
## [1] "Question in conditional probability Let   and   be two independent random variables of discrete
```

Comments:

Looking at just the first four topics and the associated words, we see the following topics: 1) data exploration and visualization, 2) clustering, 3) time series forecasting 4) probability theory. All the words in each grouping are clearly related to each other and these topics make sense to assume from these words. Other topics include classification (14) and machine learning (30).

Document one has a sparse topic distribution, centered mainly around topic 3 (0.5) and topic 25 (0.31). Topic 3 is related to time series forecasting and topic 25 is related to distributions. When looking at the actual first 200 words of text, we see that many of the words are related to distributions and their means and variances (Poisson specifically). However, we do not see anything really indicative of topic 3, time series forecasting. However, the words "times" and "time" appear alot, because they are related to Poisson distributions and what kind of data they can model. This means that the document is likely being misidentified as being related to time series forecasting because it includes these words.

Document two has a more uniform topic distribution. No topic has a proportion higher than 0.24 and there are 11 different topics represented in the document. The two most prevelent topics are topics 1 and 25, data exploration & visualization and distributions. When looking at the document, we see that posting heavily discusses multivariate data and normal distributions and identifying outliers. This makes sense because they are related to the topics with the highest proportions. However, we also see some discussion of R packages, poisson distributions, and more. It then makes sense that the distribution of topics is more uniform than that of document one.

Finally, document three again has a sparse topic distribution, with the highest proportions for topic 10 and topic 4. Topic 4 is probability theory and topic 10 is probability theory as well, these topics appear very similar and many of the top words are actually the same. The text clearly shows the posting is about conditional probability and distributions. It makes sense that we have a very sparse topic distribution where these two topics comprise over 75% of the topic proportions.

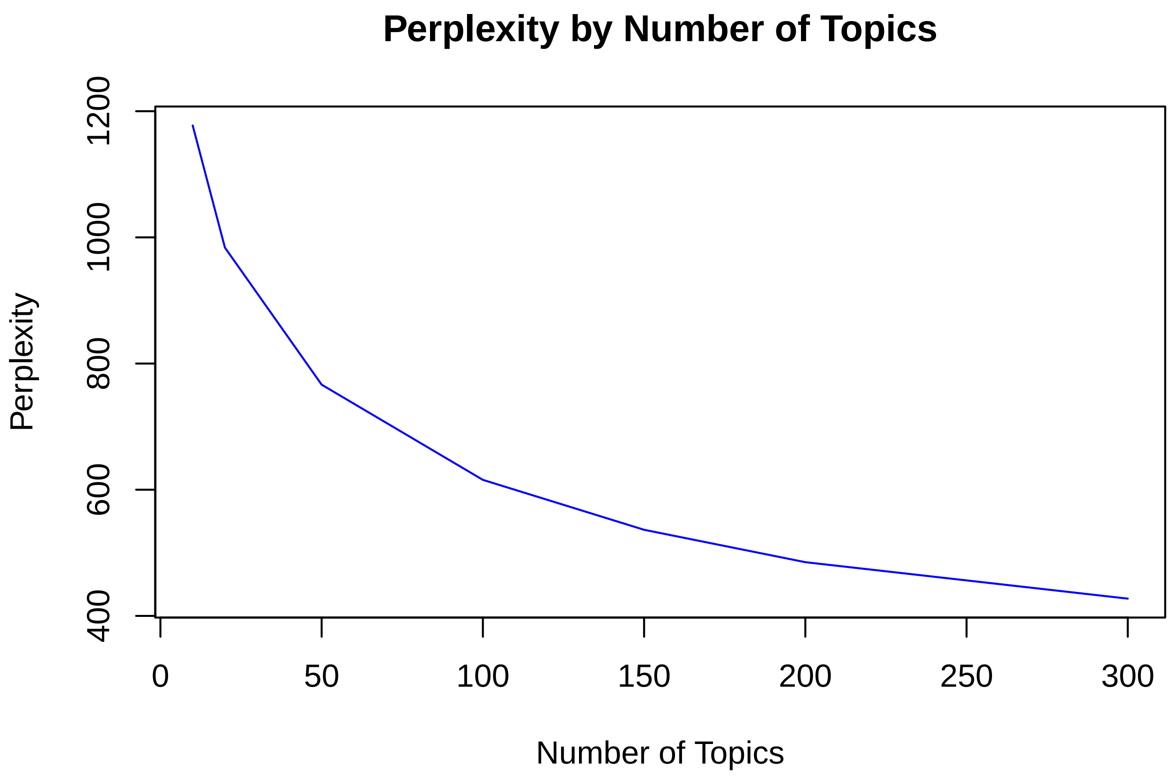
c. *Now you will evaluate the test set perplexity using the function* `perplexity` *in the* `topicmodels` *library in R. Perplexity is a measurement of how well our probability model predicts a set of data. It's essentially the analog of the MSE (mean squared error) for probability models. Analyze the test set perplexity for a range of models, fit with $K = 10, 20, \ldots, 200$ topics (or an appropriate range of your own choice). Plot the test set perplexity as a function of number of topics. Which is the best model? Do you notice any qualitative difference in the topics as K increases? Comment on your overall findings.*

```r
set.seed(1)

test_dtm <- itoken(test$CleanBody, tolower, word_tokenizer, ids = test$Id) %>%
    create_dtm(vectorizer, type = "dgTMatrix")

number_of_topics <- c(10, 20, 50, 100, 150, 200, 300)
perplexities <- rep(0, length(number_of_topics))
for (i in 1:length(number_of_topics)) {
    print(i)
    lda_model <- LDA$new(n_topics = number_of_topics[i], doc_topic_prior = 0.1,
        topic_word_prior = 0.01)
    doc_topic_distr <- lda_model$fit_transform(x = dtm, n_iter = 5000,
        convergence_tol = 0.001, n_check_convergence = 5, progressbar = FALSE)
    new_doc_topic_distr <- lda_model$transform(test_dtm)
    perplexities[i] <- perplexity(test_dtm, topic_word_distribution = lda_model$topic_word_distribution
        doc_topic_distribution = new_doc_topic_distr)
    assign(paste0("topics", number_of_topics[i]), lda_model$get_top_words())
}

plot(number_of_topics, perplexities, ylab = "Perplexity", xlab = "Number of Topics",
    main = "Perplexity by Number of Topics", type = "l", col = "blue")
```



```
# Print the topics for the model with 10 topics and the model
# with 300 topics.
print("K=10 top topics and top words")
```

```
## [1] "K=10 top topics and top words"
```

```
topics10[, 1:10]
```

```
##       [,1]       [,2]            [,3]             [,4]
##  [1,] "time"     "distribution"  "data"           "function"
##  [2,] "series"   "probability"   "training"       "likelihood"
##  [3,] "data"     "random"        "model"          "parameters"
##  [4,] "model"    "test"          "class"          "model"
##  [5,] "values"   "question"      "classification" "bayesian"
##  [6,] "process"  "hypothesis"    "learning"       "prior"
##  [7,] "forecast" "distributions" "features"       "parameter"
##  [8,] "arima"    "function"      "validation"     "distribution"
##  [9,] "trend"    "independent"   "cross"          "gaussian"
## [10,] "function" "poisson"       "feature"        "algorithm"
##       [,5]         [,6]         [,7]          [,8]           [,9]
##  [1,] "model"      "test"       "matrix"      "sample"       "data"
##  [2,] "regression" "data"       "data"        "distribution" "time"
##  [3,] "linear"     "analysis"   "correlation" "standard"     "people"
##  [4,] "variable"   "sample"     "distance"    "variance"     "question"
##  [5,] "test"       "anova"      "covariance"  "confidence"   "statistics"
##  [6,] "variables"  "difference" "clustering"  "normal"       "statistical"
##  [7,] "models"     "treatment"  "variables"   "interval"     "plot"
##  [8,] "values"     "effect"     "cluster"     "random"       "analysis"
##  [9,] "data"       "tests"      "vector"      "values"       "average"
## [10,] "effects"    "size"       "analysis"    "estimate"     "based"
##       [,10]
##  [1,] "variables"
##  [2,] "variable"
##  [3,] "regression"
##  [4,] "model"
##  [5,] "data"
##  [6,] "logistic"
##  [7,] "categorical"
##  [8,] "dependent"
##  [9,] "continuous"
## [10,] "analysis"
```

```
print("K=300 top topics and top words")
```

```
## [1] "K=300 top topics and top words"
```

```
topics300[, 1:10]
```

```
##       [,1]         [,2]       [,3]            [,4]           [,5]
##  [1,] "line"       "week"     "distribution"  "component"    "random"
##  [2,] "baseline"   "data"     "distributions" "components"   "model"
##  [3,] "function"   "days"     "uniform"       "analysis"     "effects"
##  [4,] "cumulative" "weeks"    "shape"         "principal"    "nested"
##  [5,] "straight"   "service"  "empirical"     "discriminant" "effect"
##  [6,] "represent"  "time"     "weibull"       "original"     "fixed"
##  [7,] "values"     "counts"   "distributed"   "rotation"     "mixed"
##  [8,] "called"     "count"    "discrete"      "variance"     "multilevel"
##  [9,] "horizontal" "weekly"   "pareto"        "variables"    "level"
## [10,] "bottom"     "readings" "continuous"    "orthogonal"   "lmer"
##       [,6]             [,7]          [,8]          [,9]
##  [1,] "model"          "score"       "determine"   "data"
##  [2,] "linear"         "scores"      "sense"       "outliers"
##  [3,] "models"         "scoring"     "based"       "identify"
##  [4,] "relationship"   "scored"      "makes"       "outlier"
##  [5,] "include"        "performance" "quantity"    "detect"
##  [6,] "includes"       "assessment"  "determining" "detection"
##  [7,] "discrimination" "severity"    "question"    "extreme"
##  [8,] "nested"         "symptom"     "community"   "dataset"
##  [9,] "including"      "reviews"     "role"        "identifying"
## [10,] "complex"        "judges"      "approach"    "boxplot"
##       [,10]
##  [1,] "distribution"
##  [2,] "gamma"
##  [3,] "parameters"
##  [4,] "parameter"
##  [5,] "inverse"
##  [6,] "unknown"
##  [7,] "normal"
##  [8,] "truncated"
##  [9,] "shape"
## [10,] "scale"
```

Comments: *As the number of topics increases, the perplexity decreases. This initially makes sense because, analgous to MSE, it should decrease as the number of topics increase. Therefore, the model with 300 topics is the best model with the lowest perplexity. However we would imagine perplexity would eventually increase due to overfitting. We do not see this, but maybe that is because we have not reached a number of topics where the perplexity hits an inflection point and begins to increase again.*

Then we can look at the most probably topics and their top words. For the model with fewer topics (k=10), the topics are more general. With more topics, each topic is more niche and becomes more distinguishable from each other. We see this in part b as well where many topics contain identical words and are generally very similar. When we are modeling more topics, we expect each topic to be very specific and the words to be quite different. We can see in the k=10 topics that topics 5 and 10 are both related to regression. However, for k=300, none of the top topics appear very similar.