The Analytics Edge 2017

Test Your Knowledge of Text Analytics, Clustering in R

1. Nearly every email user has at some point encountered a "spam" email, which is an unsolicited message often advertising a product, containing links to malware, or attempting to scam the recipient. Roughly 80-90% of more than 100 billion emails sent each day are spam emails, most being sent from botnets of malware-infected computers. The remainder of emails are called "ham" emails.

   As a result of the huge number of spam emails being sent across the Internet each day, most email providers offer a spam filter that automatically flags likely spam messages and separates them from the ham. Though these filters use a number of techniques (e.g. looking up the sender in a so-called "Blackhole List" that contains IP addresses of likely spammers), most rely heavily on the analysis of the contents of an email via text analytics.

   In this problem, you will build and evaluate a spam filter using a publicly available dataset first described in the 2006 conference paper "Spam Filtering with Naive Bayes – Which Naive Bayes?" by V. Metsis, I. Androutsopoulos, and G. Paliouras. The "ham" messages in this dataset come from the inbox of former Enron Managing Director for Research Vincent Kaminski, one of the inboxes in the Enron Corpus. One source of spam messages in this dataset is the SpamAssassin corpus, which contains hand-labeled spam messages contributed by Internet users. The remaining spam was collected by Project Honey Pot, a project that collects spam messages and identifies spammers by publishing email address that humans would know not to contact but that bots might target with spam. The full dataset we will use was constructed as roughly a 75/25 mix of the ham and spam messages. The dataset contains just two fields:

   - **text:** The text of the email
   - **spam:** A binary variable, 1 indicating if the email was spam and 0 otherwise

   (a) Begin by loading the dataset **emails.csv** into a data frame called **emails**. Remember to pass the stringsAsFactors=FALSE option when loading the data. How many emails are in the dataset? How many of the emails are spam?

   (b) Which word appears at the beginning of every email in the dataset?

   (c) Could a spam classifier potentially benefit from including the frequency of the word that appears in every email?

      i. No - the word appears in every email so this variable would not help us differentiate spam from ham.

      ii. Yes – the number of times the word appears might help us differentiate spam from ham.

(d) The nchar() function counts the number of characters in a piece of text. How many characters are in the longest email in the dataset (where longest is measured in terms of the maximum number of characters)?

(e) Which row contains the shortest email in the dataset? (Just like in the previous problem, shortest is measured in terms of the fewest number of characters). Write down the corresponding email.

(f) Follow the standard steps to build and pre-process the corpus:

- Load the tm package.
- Build a new corpus variable called corpus.
- Using tm_map, convert the text to lowercase.
- Using tm_map, remove all punctuation from the corpus.
- Using tm_map, remove all English stopwords from the corpus.
- Using tm_map, stem the words in the corpus.
- Build a document term matrix from the corpus, called dtm

How many terms are in dtm?

(g) To obtain a more reasonable number of terms, limit dtm to contain terms appearing in at least 5% of documents, and store this result as spdtm (don't overwrite dtm, because we will use it later). How many terms are in spdtm?

(h) Build a data frame called emailsSparse from spdtm, and use the make.names function to make the variable names of emailsSparse valid. colSums() is an R function that returns the sum of values for each variable in our data frame. Our data frame contains the number of times each word stem (columns) appeared in each email (rows). Therefore, colSums(emailsSparse) returns the number of times a word stem appeared across all the emails in the dataset. What is the word stem that shows up most frequently across all the emails in the dataset?

(i) Add a variable called "spam" to emailsSparse containing the email spam labels. How many word stems appear at least 5000 times in the ham emails in the dataset? Which word stems are these?

(j) How many word stems appear at least 1000 times in the spam emails in the dataset? Which word stems are these?

(k) The lists of most common words are significantly different between the spam and ham emails. What does this likely imply?

i. The frequencies of these most common words are unlikely to help differentiate between spam and ham.

ii. The frequencies of these most common words are likely to help differentiate between spam and ham.

(l) Several of the most common word stems from the ham documents, such as "enron", "hou" (short for Houston), "vinc" (the word stem of "Vince") and "kaminski", are likely specific

to Vincent Kaminski's inbox. What does this mean about the applicability of the text analytics models we will train for the spam filtering problem?

i. The models we build are still very general, and are likely to perform well as a spam filter for nearly any other person.

ii. The models we build are personalized, and would need to be further tested before being used as a spam filter for another person.

(m) First, convert the dependent variable to a factor with
> emailsSparse$spam <− as.factor(emailsSparse$spam)
Next, set the random seed to 123 and use the sample.split function to split emailsSparse 70-30 into a training set called "train" and a testing set called "test". Make sure to perform this step on emailsSparse instead of emails. Using the training set, train the following three models. The models should predict the dependent variable "spam", using all other available variables as independent variables. Please be patient, as these models may take a few minutes to train.

- A logistic regression model called spamLog. You may see a warning message here - we'll discuss this more later.
- A CART model called spamCART, using the default parameters to train the model. Directly before training the CART model, set the random seed to 123.
- A random forest model called spamRF, using the default parameters to train the model. Directly before training the random forest model, set the random seed to 123 (even though we've already done this earlier in the problem, it's important to set the seed right before training the model so we all obtain the same results. Keep in mind though that on certain operating systems, your results might still be slightly different).

For each model, obtain the predicted spam probabilities for the training set.
You may have noticed that training the logistic regression model yielded the messages "algorithm did not converge" and "fitted probabilities numerically 0 or 1 occurred". Both of these messages often indicate overfitting and in some case corresponds to severe overfitting, often to the point that the training set observations are fit perfectly by the model. Let's investigate the predicted probabilities from the logistic regression model.
How many of the training set predicted probabilities from spamLog are less than 0.00001?
How many of the training set predicted probabilities from spamLog are more than 0.99999?
How many of the training set predicted probabilities from spamLog are between 0.00001 and 0.99999?

(n) How many variables are labeled as significant (at the p=0.05 level) in the logistic regression summary output?

(o) How many of the word stems "enron", "hou", "vinc", and "kaminski" appear in the CART tree? Recall that we suspect these word stems are specific to Vincent Kaminski and might affect the generalizability of a spam filter built with his ham data.

(p) What is the training set accuracy of spamLog, using a threshold of 0.5 for predictions? What is the training set AUC of spamLog?

(q) What is the training set accuracy of spamCART, using a threshold of 0.5 for predictions?

(r) What is the training set AUC of spamCART? (Remember that you have to pass the prediction function predicted probabilities.)

(s) What is the training set accuracy of spamRF, using a threshold of 0.5 for predictions? (Remember that your have to use type="prob" in your prediction for random forest.)

(t) What is the training set AUC of spamRF? (Remember to pass the argument type="prob" to the predict function to get predicted probabilities for a random forest model. The probabilities will be the second column of the output.)

(u) Which of the models have the best training set performance, in terms of accuracy and AUC?

- Logistic regression
- CART
- Random forest

(v) Obtain predicted probabilities for the testing set for each of the models, again ensuring that probabilities instead of classes are obtained. What is the testing set accuracy of spamLog, using a threshold of 0.5 for predictions?

(w) What is the testing set AUC of spamLog? What is the testing set accuracy of spamCART, using a threshold of 0.5 for predictions? What is the testing set AUC of spamCART? What is the testing set accuracy of spamRF, using a threshold of 0.5 for predictions? What is the testing set AUC of spamRF?

(x) Which model had the best testing set performance, in terms of accuracy and AUC?

- Logistic regression
- CART
- Random forest

(y) Which model demonstrated the greatest degree of overfitting?

- Logistic regression
- CART
- Random forest

2. In this question, we will use cluster-then-predict, a methodology in which you first cluster observations and then build cluster-specific prediction models. In this assignment, we'll use cluster-then-predict to predict future stock prices using historical stock data. When selecting which stocks to invest in, investors seek to obtain good future returns. In this problem, we will first use clustering to identify clusters of stocks that have similar returns over time. Then, we'll use logistic regression to predict whether or not the stocks will have positive future returns. For this problem, we'll use StocksCluster.csv, which contains monthly stock returns from the NASDAQ stock exchange. The NASDAQ is the second-largest stock exchange in the world, and it lists many technology companies. The stock price data used in this problem was obtained from infochimps, a website providing access to many datasets. Each observation in the dataset is the monthly returns of a particular company in a particular year. The years included are 2000-2009. The companies are limited to tickers that were listed on the exchange for the entire period 2000-2009, and whose stock price never fell below $1. So, for example, one observation is for Yahoo in 2000, and another observation is for Yahoo in 2001. Our goal will be to predict whether or not the stock return in December will be positive, using the stock returns for the first 11 months of the year. This dataset contains the following variables:

   - **ReturnJan:** the return for the company's stock during January (in the year of the observation)
   - **ReturnFeb:** the return for the company's stock during February (in the year of the observation)
   - **ReturnMar:** the return for the company's stock during March (in the year of the observation)
   - **ReturnApr:** the return for the company's stock during April (in the year of the observation)
   - **ReturnMay:** the return for the company's stock during May (in the year of the observation)
   - **ReturnJune:** the return for the company's stock during June (in the year of the observation)
   - **ReturnJuly:** the return for the company's stock during July (in the year of the observation)
   - **ReturnAug:** the return for the company's stock during August (in the year of the observation)
   - **ReturnSep:** the return for the company's stock during September (in the year of the observation)
   - **ReturnOct:** the return for the company's stock during October (in the year of the observation)
   - **ReturnNov:** the return for the company's stock during November (in the year of the observation)

- **PositiveDec:** whether or not the company's stock had a positive return in December (in the year of the observation). This variable takes value 1 if the return was positive, and value 0 if the return was not positive.

For the first 11 variables, the value stored is a proportional change in stock value during that month. For instance, a value of 0.05 means the stock increased in value 5% during the month, while a value of -0.02 means the stock decreased in value 2% during the month.

(a) Load **StocksCluster.csv** into a data frame called **stocks**. How many observations are in the dataset?

(b) What proportion of the observations have positive returns in December?

(c) What is the maximum correlation between any two return variables in the dataset? You should look at the pairwise correlations between ReturnJan, ReturnFeb, ReturnMar, ReturnApr, ReturnMay, ReturnJune, ReturnJuly, ReturnAug, ReturnSep, ReturnOct, and ReturnNov

(d) Which month (from January through November) has the largest mean return across all observations in the dataset? Which month (from January through November) has the smallest mean return across all observations in the dataset?

(e) Run the following commands to split the data into a training set and testing set, putting 70% of the data in the training set and 30% of the data in the testing set:
> set.seed(144)
> spl <− sample.split(stocks$PositiveDec, SplitRatio = 0.7)
> stocksTrain <− subset(stocks, spl == TRUE)
> stocksTest <− subset(stocks, spl == FALSE)
Then, use the stocksTrain data frame to train a logistic regression model (name it StocksModel) to predict PositiveDec using all the other variables as independent variables. What is the overall accuracy on the training set, using a threshold of 0.5?

(f) Now obtain test set predictions from StocksModel. What is the overall accuracy of the model on the test, again using a threshold of 0.5?

(g) What is the accuracy on the test set of a baseline model that always predicts the most common outcome in the training set?

(h) Now, let's cluster the stocks. The first step in this process is to remove the dependent variable using the following commands:
> limitedTrain <− stocksTrain
> limitedTrain$PositiveDec <− NULL
> limitedTest <− stocksTest
> limitedTest$PositiveDec <− NULL
Why do we need to remove the dependent variable in the clustering phase of the cluster-then-predict methodology?

i. Leaving in the dependent variable might lead to unbalanced clusters

ii. Removing the dependent variable decreases the computational effort needed to cluster

iii. Needing to know the dependent variable value to assign an observation to a cluster defeats the purpose of the methodology

(i) In some cases where we have a training and testing set, we might want to normalize by the mean and standard deviation of the variables in the training set. We can do this by using the **caret** package passing just the training set to the preProcess function, which normalizes variables by subtracting by the mean and dividing by the standard deviation.
> library(caret)
> preproc <− preProcess(limitedTrain)
> normTrain <− predict(preproc, limitedTrain)
> normTest <− predict(preproc, limitedTest)
What is the mean of the ReturnJan variable in normTrain?
What is the mean of the ReturnJan variable in normTest?

(j) Why is the mean ReturnJan variable much closer to 0 in normTrain than in normTest?

i. Small rounding errors exist in the normalization procedure

ii. The distribution of the ReturnJan variable is different in the training and testing set

iii. The distribution of the dependent variable is different in the training and testing set

(k) Set the random seed to 144 (it is important to do this again, even though we did it earlier). Run k-means clustering with 3 clusters on normTrain, storing the result in an object called km. Which cluster has the largest number of observations?

i. Cluster 1

ii. Cluster 2

iii. Cluster 3

(l) In this question, we use the flexclust package to obtain training set and testing set cluster assignments for our observations and to do the predictions. Use the following commands:
> library(flexclust)
> km.kcca <− as.kcca(km, normTrain)
> clusterTrain <− predict(km.kcca)
> clusterTest <− predict(km.kcca, newdata=normTest)
How many test-set observations were assigned to Cluster 2?

(m) Using the subset function, build data frames stocksTrain1, stocksTrain2, and stocksTrain3, containing the elements in the stocksTrain data frame assigned to clusters 1, 2, and 3, respectively (be careful to take subsets of stocksTrain, not of normTrain). Similarly build stocksTest1, stocksTest2, and stocksTest3 from the stocksTest data frame. Which training set data frame has the highest average value of the dependent variable?

i. stocksTrain1

ii. stocksTrain2

iii. stocksTrain3

(n) Build logistic regression models StocksModel1, StocksModel2, and StocksModel3, which predict PositiveDec using all the other variables as independent variables. StocksModel1 should be trained on stocksTrain1, StocksModel2 should be trained on stocksTrain2, and StocksModel3 should be trained on stocksTrain3. Which variables have a positive sign for the coefficient in at least one of StocksModel1, StocksModel2, and StocksModel3 and a negative sign for the coefficient in at least one of StocksModel1, StocksModel2, and StocksModel3?

(o) Using StocksModel1, make test-set predictions called PredictTest1 on the data frame stocksTest1. Using StocksModel2, make test-set predictions called PredictTest2 on the data frame stocksTest2. Using StocksModel3, make test-set predictions called PredictTest3 on the data frame stocksTest3.
What is the overall accuracy of StocksModel1 on the test set stocksTest1, using a threshold of 0.5?
What is the overall accuracy of StocksModel2 on the test set stocksTest2, using a threshold of 0.5?
What is the overall accuracy of StocksModel3 on the test set stocksTest3, using a threshold of 0.5?

(p) To compute the overall test-set accuracy of the cluster-then-predict approach, we can combine all the test-set predictions into a single vector and all the true outcomes into a single vector:
> AllPredictions <− c(PredictTest1, PredictTest2, PredictTest3)
> AllOutcomes <− c(stocksTest1$PositiveDec, stocksTest2$PositiveDec, stocksTest3$PositiveDec)
What is the overall test-set accuracy of the cluster-then-predict approach, again using a threshold of 0.5?

3. Clustering is commonly used to divide a broad target market of customers into smaller, similar groups and then to design marketing strategies specifically for each group. In this question, you will use clustering to study publicly available data from the New York Citi Bike sharing program. Citi Bike is the largest bike sharing program in the United States and serves various parts of the New York city. The data is provided in the file **citibike.csv** and contains trip information for the bike rides for the month of July 2013:

- **tripduration**: Time duration of the trip (in seconds)
- **startstation**: Name of the start station
- **endstation**: Name of the end station
- **gender**: Gender of user (1 = male, 2 = female)
- **age**: Age of the user
- **day**: Day on which the trip was started (Mon, Tue, Wed, Thu, Fri, Sat, Sun)
- **starttime**: Start time of the trip in unit of hours (measured from 0 (12am) to 23 (11 pm))

(a) Read the dataset into the dataframe **citi**. How many bike stations are there in this dataset?

(b) On which day of the week, is the average duration of the trips taken by bikers the maximum?

(c) What is the start hour when the maximum number of bikes are rented? What is the start hour when the minimum number of bikes are rented?

(d) In this dataset, what proportion of the bikes are rented by female users?

(e) One of the challenges to do clustering with this data is the presence of the categorical variable for the **day** variable. To tackle this, we will define seven new binary variables (**Mon** to **Sun**), each of which takes a value of 1 if the corresponding trip is taken on that day and 0 otherwise. Write down one such sample R command(s) that you use to do this for a given day of the week.

(f) In clustering data, it is often important to normalize the variables so that they are all on the same scale. If you clustered this dataset without normalizing, which variable would you expect to dominate in the distance calculations?

- **tripduration**
- **gender**
- **age**
- **starttime**
- **Mon**

(g) Normalize the variables **tripduration**, **gender**, **age**, **starttime**, **Mon**, ..., **Sun** by using the **scale()** function in R. We normalize such that each variable has mean 0 and standard deviation 1. What is the maximum value of **tripduration** in the normalized dataset?

(h) We will not use hierarchical clustering for this dataset. Why do you think hierarchical clustering might have a problem with this dataset?

- We have categorical variables in this dataset, so we cant use hierarchical clustering.
- We might have too many variables in the dataset for hierarchical clustering to handle.
- We might have too many observations in the dataset for hierarchical clustering to handle.
- We are sure of the number of clusters in this application, so using hierarchical clustering does not make sense.

(i) Run the k-means algorithm on the normalized dataset with 10 clusters. Use set.seed(100) in R just before running the code. You only want to use the variables **tripduration**, **gender**, **age**, **starttime**, **Mon**, ..., **Sun** in building your clusters. Use the default settings to build your model. How many trips are there in the largest and smallest clusters respectively?

(j) Which cluster best fits the description "trips taken primarily by older users on Saturdays"? You can use the centers of the clusters to answer this question.

(k) Which cluster best fits the description "longer trips taken primarily by female users either on Tuesdays or Wednesdays"? You can use the centers of the clusters to answer this question.

(l) If we ran k-means clustering a second time without making any additional calls to set.seed, we would expect

- Different results from the first k-means clustering
- Identical results to the first k-means clustering

(m) If we ran k-means clustering a second time, again running the command set.seed(100) right before doing the clustering, we would expect

- Different results from the first k-means clustering
- Identical results to the first k-means clustering

(n) Suppose the marketing department at Citi Bike decided that instead of using the days of the week for clustering, they would like to use a single variable **weekday** which took a value of 1 if the trip started on Monday, Tuesday, Wednesday, Thursday, or Friday and 0 if it started on a Saturday or Sunday. Redo the clustering. As before, remember to normalize the **weekday** variable and run the k-means algorithm on the normalized dataset with 10 clusters. Use set.seed(100) in R before running the code. Which cluster best fits the description "longer trips taken by older female users on weekdays"? You can use the centers of the clusters to answer this question.

(o) Which cluster best fits the description "short trips taken by younger male users early on weekdays"? You can use the centers of the clusters to answer this question.