Title: Capstone Yelp Restaurant Review Prediction

Scot Shields

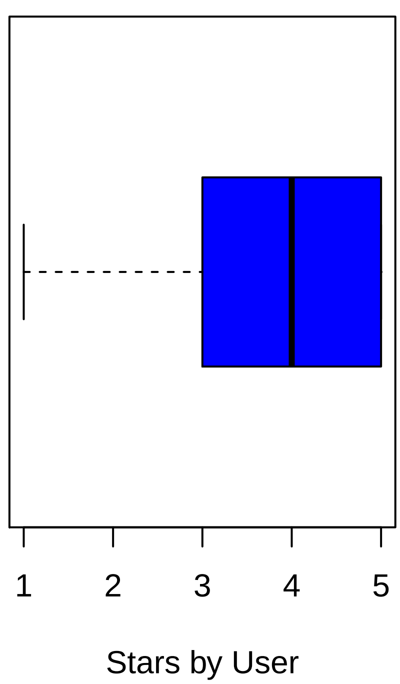
November 21, 2015

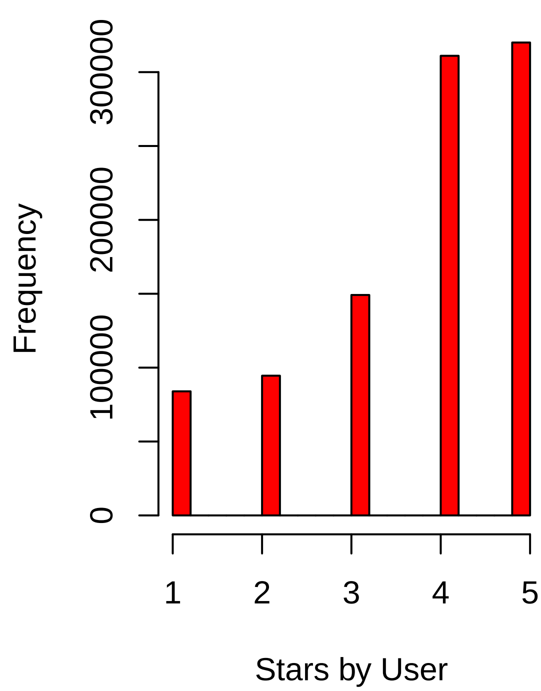
Intro:

The problem I intend to address for my capstone project is creating a model that will predict if a reviewer will like or dislike a restaurant. This prediction will be based on the number of stars the reviewer has given other restaurants, and the features those restaurants had. I will define liking a restaurant as rating it a certain number of stars or greater, anything less than that number of stars would be a dislike. I think this project could be a helpful step towards building a recommender system. It would be of interest to reviewers who use yelp and would like to find new restaurants they may enjoy.

Methods:

exploratory analysis






```
##    user_id           business_id            stars
##  Length:958777      Length:958777      Min.   :1.000
##  Class :character   Class :character   1st Qu.:3.000
##  Mode  :character   Mode  :character   Median :4.000
```

```
##                                         Mean   :3.718
##                                         3rd Qu.:5.000
##                                         Max.   :5.000

## Restaurant Ratings Stars variance:  1.592669

## Restaurant Ratings Stars stddev:  1.26201

## Restaurant Ratings Stars skewness:  -0.799382
```

From the boxplot we see that the majority of restaurants in the data have between 3 and 5 stars. From the histogram and by running the skewness function on the data, we see the data is left skewed pulling the mean towards 3. According to the yelp website, a rating of 3 stars is defined as "A-Ok." So I think it's safe to assume that a rating of 3 stars indicates the restaurant is merely adequate, and ratings of 4 or 5 stars indicate the reviewer "liked" the restaurant. For the model I defined liking a restaurant as rating it 4 or more stars.

Prediction Algorithm:

1) Loaded Yelp dataset into r and flattened it using various packages including jsonlite.

2) filtered the categories field in the business data for records that contained "Restaurants," using grep.
3) Loaded and flattened review data.
4) filtered review data to only include records that had business_ids found in the business data, filtered for restaurants business_id field.
5) filtered the dataframe to only include unique ratings by each user_id for each business_id.
6) filtered review data to find the user_id with the most reviews.
7) joined review data to business data on business_id, using join.
8) added a new field to user_model called like, which labeled each record True or False depending on if it had a rating of 4 stars or more.
9) parsed the list items in the categories field from the user_model, and added new fields to the dataframe which labeled each record true if it contained the list value the field was named for, using mtabulate.
10) removed the user_id, stars, business_id, review_count, and all fields that were not type logical except the "like" field.
11) replaced all "na" values to "FALSE."
12) ran nearZeroVar on the dataframe to find all fields with very low variance. After reviewing the list of fields, I removed them.
13) coerced the "like" field to a factor.
14) partitioned the dataframe into 60% training and 40 % testing data. I trained a randomforest model, with crossvalidation, and 5 folds.
15) Evaluated the model predictions on the test data using confusionMatrix.
16) Calculated the in sample error rate and out of sample error rate.

Special Methods Used:

flatten:

Methods for making an object 'flat', such as creating a non-nested list from a list, and methods for collecting information from list-like objects into a matrix or data frame.

grep:

search for matches to argument pattern within each element of a character vector.

duplicated:

Determines which elements of a vector or data frame are duplicates of elements with smaller subscripts, and returns a logical vector indicating which elements (rows) are duplicates.

join:

Join, like merge, is designed for the types of problems where you would use a sql join.

mtabulate:

Tabulate Frequency Counts for Multiple Vectors.

nearZeroVar:

nearZeroVar diagnoses predictors that have one unique value (i.e. are zero variance predictors) or predictors that have both of the following characteristics: they have very few unique values relative to the number of samples and the ratio of the frequency of the most common value to the frequency of the second most common value is large. checkConditionalX looks at the distribution of the columns of x conditioned on the levels of y and identifies columns of x that are sparse within groups of y.

randomforest:

An ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random forests correct for decision trees' habit of overfitting to their training set.

crossvalidation:

A model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. A model is given a dataset of known data on which training is run, and a dataset of unknown data against which the model is tested.

confusionMatrix:

Calculates a cross-tabulation of observed and predicted classes with associated statistics.

In Sample Error:

The error rate you get on the same data set you used to build your predictor. Sometimes called resubstitution error.

Out of Sample Error:

The error rate you get on a new data set.

cbind:

Take a sequence of vector, matrix or data frames arguments and combine by rows.

Results:

For more info on restaurants the model predicts the reviewer will like/dislike I cbind the model predictions to the Reviewer_Model_tab dataframe, from earlier in the algorithm. This dataframe contains additional variables (ie. "business_id","categories",…) which were removed from the training and testing data. The total test predictions are too many to list in this report. There is a sample of the predictions the model made on the test data below. For this example the test predictions were cbind to the previously mentioned dataframe to add the name of the restaurant.

Sample Model Predictions:

```
##       test_pred  like                                  name
## 3419       True  True                  Stockyards Restaurant
## 5020       True  True Fleming's Prime Steakhouse & Wine Bar
## 5204       True  True                            Arrivederci
## 14142      True  True                       Bonito Michoacan
## 14711      True  True                  Lawry's The Prime Rib
## 14806      True  True                              Carmine's
## 15977      True  True                           Mon Ami Gabi
## 15993      True  True             Tom Colicchio's Craftsteak
## 16124      True False           Bobby's Restaurant and Lounge
## 16361      True  True        Nobhill Tavern by Michael Mina
## 16862      True  True                    Mastro's Ocean Club
## 16885      True  True                             STRIPSTEAK
## 17478      True  True             BJ's Restaurant & Brewhouse
## 22252      True  True                                 Taggia
## 23657      True  True                  Alto Ristorante e Bar
## 26785      True False             Lorenzo's Pizza and Pasta
## 36589      True  True                               Eddie D's
## 38471      True  True                              Orange Sky
## 40767      True False                    SC Prime Steakhouse
## 41074      True  True                     Biaggio's Pizzeria
## 41303      True  True                  Spotted Donkey Cantina
## 51056      True  True                             Ciao Grazie
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction False True
##      False   149   67
##      True      3   19
##
##                Accuracy : 0.7059
##                  95% CI : (0.6436, 0.763)
##     No Information Rate : 0.6387
##     P-Value [Acc > NIR] : 0.01717
##
##                   Kappa : 0.24
##  Mcnemar's Test P-Value : 5.076e-14
##
##             Sensitivity : 0.22093
##             Specificity : 0.98026
##          Pos Pred Value : 0.86364
##          Neg Pred Value : 0.68981
##              Prevalence : 0.36134
##          Detection Rate : 0.07983
##    Detection Prevalence : 0.09244
##       Balanced Accuracy : 0.60060
##
##        'Positive' Class : True
##
```

```
## In Sample Error Rate:  0.2150838
```

```
## Out of Sample Error Rate:  0.2941176
```

Summary of Results:

From Cross Validation we see that the model is about 70% accurate, meaning the model correctly predicted if the reviewer liked or disliked a restuarant about 70% of the time on the test data.

The P-Value from the hypothesis test comparing the accuracy and the NIR is 0.01717 indicating the accuracy is greater than the rate of the largest class.

The In Sample Error is about 22% and the out of sample error is about 29%. So there may be a little bit of over fitting of the model to the training data.

The sensitivity is aproximately 22% and the is specificity is about 98%. So the model correctly predicted that the reviewer would like a restaurant about 22% of the time, and correctly predicted that the reviewer wouldn't like a restaurant about 98% of the time.

The positive predictive value is about 86% and the negative predictive value is about 69%. So when the model predicts that the reviewer likes a restaurant there's 86% chance it's correct and when the model predicts the reviewer doesn't like a restaurant there's an 69% chance it's correct.

Discussion:

The model Accuracy is about 70%. The model seems to do an ok job of identifying restuarants the reviewer won't like (Specificity ~ 98%), not such a good job of identifying restaurants the reviewer will like (Sensitivity ~ 22%), but when it predicts the reviewer will like a restaurant it's pretty likely they will (Positive Predictive Value ~ 86%). The out of sample error rate is higher than the in sample error rate, so there appears to be some overfitting. From the p value we see that the Accuracy of the model is significantly better than the NIR (P-Value [accuracy>NIR] is 0.01717). So in conclusion I have created a model that will predict if a reviewer will like or dislike a restaurant.

- Code Repo: https://github.com/scotsditch/YelpCaspstoneProject