Coursera Data Science Specialization Capstone Project

Boris Badmaev

November 15, 2015

Introduction

This capstone project is the final part of Coursera Data Science specialization course taught by professors of the Johns Hopkins University. The data for this project was provided by Yelp (Yelp dataset challenge). Yelp.com and the Yelp mobile app publish crowd-sourced reviews about local businesses.

The objective of this research is to predict if a "Vegetarian" category can be added to the list of categories of a business (restaurants) based on Yelp reviews, particularly based on a frequency of certain words (a.k.a. marker words) present in the reviews. The prediction algorithm will help Yelp to suggest owners to add "Vegetarian" tag/categories to their business categories in order to attract the vegetarian crowd and/or help vegetarian users search such businesses on Yelp.

Data

The data for this capstone come from Yelp, which is a business founded in 2004 to "help people find great local businesses like dentists, hair stylists and mechanics." As of the second quarter of 2015, Yelp had a monthly average of 83 million unique visitors who visited Yelp via their mobile device and written more than 83 million reviews. There are 5 files in the Yelp Dataset, each file is composed of a single object type, one json-object per-line.

In order to answer our question only two object types will be used: `business` and `review`. To load the data the following code was used:

```
business <- stream_in(file("yelp_academic_dataset_business.json"))
business <- flatten(business)
review <- stream_in(file("yelp_academic_dataset_review.json"))
review <- flatten(review)
```

Methods

There are two major steps in our analysis:

1. Explore reviews and come up with the list of "marker" words that we can used to identify whether a business serves/sells vegetarian food. In addition to that identify the "marker" words that usually associated with carnivorous diet.
2. Build a model that helps predicting whether a business can be associated with vegetarian or non-vegetarian diet based on frequency of "marker" words present in reviews for that business.

Exploratory Analysis and Selection or Marker Words

The **business** data frame contains Categories column that we can use to create two subsets of businesses:

a) Have **Vegetarian|Vegan** tags

b) Have **Steakhouses|Burgers|Barbeque|American (Traditional)** tags

Since these businesses already have been identified by owners as vegetarian/non-vegetarian they can be used for both: selection of vegetarian/non vegetarian marker words and for creation and testing of our prediction models.

```
# selecting vegetarian and carnivorous restraunts:
v<-grepl("Vegetarian|Vegan", business$categories)
c<-grepl("Steakhouses|Burgers|Barbeque|American (Traditional)", business$categories)
business.v<-business[v,]
business.c<-business[c,]
# merging business data with reviews data
data.v<-merge(business.v, review, by="business_id")
data.c<-merge(business.c, review, by="business_id")
```

Then all reviews to be summarized per business

```
# summarizing reviews based on business_id
library(dplyr)
dataV<-data.v%>%group_by(business_id)%>%summarize(review=paste(text, collapse=" "))
dataC<-data.c%>%group_by(business_id)%>%summarize(review=paste(text, collapse=" "))
```

Then we need to create one corpus for Vegetarian reviews and another one for Non-vegetarian reviews.

```
library(tm)
library(wordcloud)
corpus.v<-Corpus(VectorSource(dataV$review))
corpus.c<-Corpus(VectorSource(dataC$review))
```

Next a series of text transformations is to be performed:
- transform words to lower case
- remove so-called “stop words” (English)
- remove punctuation
- get rid of extra spaces
- remove numbers

Through multiple iteration it was found out that there is a lot of “noise” - the words that do not characterize the food or cuisine. Examples of such words would be: “restaurant”, “yelp”, “awesome”, “table”, “staff”, etc. Thus we need to remove these words too.

```
corpus.v<-tm_map(corpus.v, removeWords, words0)
corpus.c<-tm_map(corpus.c, removeWords, words0)
```

After removing these “noisy” words we’re ready to come up with the lists of terms (marker words) from the reviews that are used more often to characterized the vegetarian/non-vegetarian food/cuisine. These terms later will be used in creating the prediction model. Now let’s see what are the most frequent terms and demonstrate the corresponding word clouds. In order to do that we will create Document-Term Matrix and calculate frequency of the all the words. Then the results will be sorted and presented graphically via wordcloud function:

Ok, now we can make an intelligent choice of "marker" words that are frequent in vegetarian vs non-vegetarian reviews:
Vegetarian terms: "chicken", "vegetarian", "vegan", "veggie", "indian", "salad", "healthy", "fresh", "cheese", "hummus", "green", "pizza", "thai", "soup", "eggplant", "curry"
Non-vegetarian terms: "burger", "steak", "fries", "meat", "rib", "bbq", "beef", "pork", "bacon", "grill", "sauce"

Prediction models creation

Now let's start by preparing the training and test datasets.
First, we are going to re-use the same merged datasets we had in the previous steps: data.v (315) and data.c (2460). We will add **vflag** column to both datasets that we want our model to predict. It will have a binomial outcome: vegetarian vs. non-vegetarian. For the business in training/test data the values are known.

In addition to *vflag* we added 27 columns - one "marker" word per column. The numbers in these 27 columns represent frequency of a particular "marker" word in the summarized reviews for a particular business.

Let's mix vegetarian and non-vegetarian datasets together to create Train and Test datasets.

```
library(caret)
inTrain.v<-createDataPartition(y=data_v$vflag, p=0.7, list=FALSE)
train.v<-data_v[inTrain.v,]
test.v<-data_v[-inTrain.v,]
inTrain.c<-createDataPartition(y=data_c$vflag, p=0.7, list=FALSE)
train.c<-data_c[inTrain.c,]
test.c<-data_c[-inTrain.c,]
train<-rbind(train.v, train.c)
```

```
test<-rbind(test.v, test.c)
train$vflag<-as.factor(train$vflag)
test$vflag<-as.factor(test$vflag)
```

Let's split the train dataset further so we can cross-validate the prediction results

```
inTrain<-createDataPartition(y=train$vflag, p=0.7, list=FALSE)
subTrain<-train[inTrain,]
subTest<-train[-inTrain,]
```

Model 1: logistic regression model

```
fit_glm<-glm(vflag~., data=subTrain, family="binomial")
summary(fit_glm)
```

```
##
## Call:
## glm(formula = vflag ~ ., family = "binomial", data = subTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3654  -0.1629  -0.0022   0.0000   3.2973
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.1173027  0.2876769 -10.836  < 2e-16 ***
## vegetarian   1.6592145  0.3522398   4.710 2.47e-06 ***
## burger      -0.2593895  0.0445452  -5.823 5.78e-09 ***
## chicken     -0.0154096  0.0469486  -0.328 0.742744
## vegan        1.6737945  0.3889078   4.304 1.68e-05 ***
## veggie       0.8538935  0.2865537   2.980 0.002884 **
## indian       0.4636995  0.6147274   0.754 0.450659
## salad        0.0753141  0.0916916   0.821 0.411427
## healthy      0.0006413  0.0784053   0.008 0.993474
## fresh        0.2368637  0.0596142   3.973 7.09e-05 ***
## cheese       0.1983301  0.1224981   1.619 0.105437
## hummus       1.3053458  0.3882026   3.363 0.000772 ***
## green       -0.4362336  0.1374706  -3.173 0.001507 **
## pizza        0.0315668  0.0447954   0.705 0.481004
## thai         0.1492594  0.1729976   0.863 0.388257
## soup         0.2001113  0.2011386   0.995 0.319789
## eggplant    -3.2933664  0.6734200  -4.891 1.01e-06 ***
## curry       -0.0646636  0.3053855  -0.212 0.832307
## steak       -1.1604255  0.2477173  -4.684 2.81e-06 ***
## fries        0.1144476  0.0689217   1.661 0.096805 .
## meat        -0.1637806  0.1359267  -1.205 0.228235
## rib         -0.3730141  0.2378864  -1.568 0.116873
## bbq         -0.0336942  0.1430537  -0.236 0.813793
## beef        -0.0300830  0.0425094  -0.708 0.479145
## pork        -0.1892130  0.2055850  -0.920 0.357383
## bacon       -0.2766954  0.1632864  -1.695 0.090163 .
```

```
## grill       -1.0232786  0.2128317  -4.808 1.53e-06 ***
## sauce       -0.0174790  0.0958464  -0.182 0.855296
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 964.88  on 1359  degrees of freedom
## Residual deviance: 164.90  on 1332  degrees of freedom
## AIC: 220.9
##
## Number of Fisher Scoring iterations: 16
```

So based on summary we can say that, for example, for every increase of frequency of "vegetarian" word in the reviews the log odds of having "vegetarian" category for this business increases by 1.66. That's pretty predictable. What surprised me was that "eggplant" in the reviews actually reduced the log odds of having vegetarian tag (to be honest "eggplant" wasn't in the vegetarian cloud - I added it as my favorite vegetarian dish)

Model 2: Random Forrest

```
library(randomForest)
fit_rf<-randomForest(vflag~., data=subTrain, type="class")
```

Results

Let's predict the vflag outcome using our two models

```
prob_glm<-predict(fit_glm, subTest,type="response")
predict_glm<-factor(ifelse(prob_glm>0.5, "vegetarian", "non-vegetarian"))
predict_rf<-predict(fit_rf, subTest)
confusionMatrix(predict_glm, subTest$vflag)
```

```
## Confusion Matrix and Statistics
##
##                 Reference
## Prediction       non-vegetarian vegetarian
##   non-vegetarian            510         13
##   vegetarian                  6         53
##
##                Accuracy : 0.9674
##                  95% CI : (0.9495, 0.9802)
##     No Information Rate : 0.8866
##     P-Value [Acc > NIR] : 1.141e-12
##
##                   Kappa : 0.8298
##  Mcnemar's Test P-Value : 0.1687
##
##             Sensitivity : 0.9884
##             Specificity : 0.8030
```

```
##          Pos Pred Value : 0.9751
##          Neg Pred Value : 0.8983
##              Prevalence : 0.8866
##          Detection Rate : 0.8763
##    Detection Prevalence : 0.8986
##       Balanced Accuracy : 0.8957
##
##        'Positive' Class : non-vegetarian
##
```

```
confusionMatrix(predict_rf, subTest$vflag)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction       non-vegetarian vegetarian
##   non-vegetarian            512         17
##   vegetarian                  4         49
##
##                Accuracy : 0.9639
##                  95% CI : (0.9454, 0.9775)
##     No Information Rate : 0.8866
##     P-Value [Acc > NIR] : 1.462e-11
##
##                   Kappa : 0.8037
##  Mcnemar's Test P-Value : 0.008829
##
##             Sensitivity : 0.9922
##             Specificity : 0.7424
##          Pos Pred Value : 0.9679
##          Neg Pred Value : 0.9245
##              Prevalence : 0.8866
##          Detection Rate : 0.8797
##    Detection Prevalence : 0.9089
##       Balanced Accuracy : 0.8673
##
##        'Positive' Class : non-vegetarian
##
```

Here the Logistic Regression model performed a bit better than Random Forest. Now let's do our final check and see how well our prediction models work on **test** dataset

```
f.prob_glm<-predict(fit_glm, test, type="response")
f.predict_glm<-factor(ifelse(f.prob_glm>0.5, "vegetarian", "non-vegetarian"))
f.predict_rf<-predict(fit_rf, test)
confusionMatrix(f.predict_glm, test$vflag)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction       non-vegetarian vegetarian
##   non-vegetarian            727         22
```

```
##   vegetarian                 10         72
##
##                Accuracy : 0.9615
##                  95% CI : (0.9461, 0.9735)
##     No Information Rate : 0.8869
##     P-Value [Acc > NIR] : 9.189e-15
##
##                   Kappa : 0.7968
##  Mcnemar's Test P-Value : 0.05183
##
##             Sensitivity : 0.9864
##             Specificity : 0.7660
##          Pos Pred Value : 0.9706
##          Neg Pred Value : 0.8780
##              Prevalence : 0.8869
##          Detection Rate : 0.8748
##    Detection Prevalence : 0.9013
##       Balanced Accuracy : 0.8762
##
##        'Positive' Class : non-vegetarian
##
```

```
confusionMatrix(f.predict_rf, test$vflag)
```

```
## Confusion Matrix and Statistics
##
##                 Reference
## Prediction       non-vegetarian vegetarian
##   non-vegetarian            735         23
##   vegetarian                  2         71
##
##                Accuracy : 0.9699
##                  95% CI : (0.9559, 0.9804)
##     No Information Rate : 0.8869
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8339
##  Mcnemar's Test P-Value : 6.334e-05
##
##             Sensitivity : 0.9973
##             Specificity : 0.7553
##          Pos Pred Value : 0.9697
##          Neg Pred Value : 0.9726
##              Prevalence : 0.8869
##          Detection Rate : 0.8845
##    Detection Prevalence : 0.9122
##       Balanced Accuracy : 0.8763
##
##        'Positive' Class : non-vegetarian
##
```

As we can see here both algorithms are pretty close to each other and performed with consistent accuracy (~96%).

To summarize we can say that our models can confidently predict the "vegetarian/non-vegetarian" category based on frequency of "marker" words in the reviews.

Discussion

Do to limitations of this project some of the interesting follow up questions fall out of scope of this paper. I would be interesting to cover the following topics in the future:
- how algorithms can be optimized? For example, choose less "marker" words etc.
- can we include frequency of 2-grams, 3-grams or phrases?
- can similar approach be used for predicting other business categories?
- check how our prediction algorithms work for other businesses (that were not used in "marker" words selection or prediction model creation processes). I did run algorithms to come up with predictions and consistently received the results I expected (e.g. McDonald's: non-vegetarian, P.F.Chang: vegetarian)
- create an app that based on business_id (or name) gives back the vflag (vegetarian/non-vegetarian) outcome and probability.