## Quantifying the Influence of Sentiment

author: Megan Minshew
date: November 15, 2015

**Introduction** Does the sentiment measureable in review text in combination with ratings provide an indication of usefulness to a prospective customer? If we assume that liking a tip or voting on a review is an indicator of influence, can we then reliably identify the prospective customer's positivity or negativity to the business?

## Methods Phase 1: Identification of sentiment in reviews and classification by level of positivity/negativity.

Inital data cleanup is performed in R and prepares data for hive processing

```
library(jsonlite);library(dplyr);library(stringr);library(psych);library(caret);library(randomForest);
# load the review data
review <- stream_in(file("yelp_academic_dataset_review.json"), verbose=FALSE)
# create a data frame and promote the nested votes
tidy_review <- as.data.frame(review$review_id)
tidy_review$stars <- as.factor(review$stars); tidy_review$isfunny <- review$votes$funny;
tidy_review$isuseful <- review$votes$useful; tidy_review$iscool <- review$votes$cool
# reviews are messy and need some basic cleanup before the data is useable
txtreview <- review$text
# clean reviews with gsub removing numbers punctuation and control chars and convert to lower
txtreview <- gsub('[[:punct:]]', '', txtreview); txtreview <- gsub('[[:cntrl:]]', '', txtreview)
txtreview <- gsub('\\d+', '', txtreview); txtreview <- tolower(txtreview)
# add to our dataframe
tidy_review$reviewtext <- txtreview
# output for next environment
write.csv(tidy_review, "tidyreview.csv")
```

Sentiment classification will be using two methods that are complimentary. The first has been entirely developed for this project. It takes each word and uses the star rating for the entire review as the basis of each word's level of positivity or negativity. Words will be assigned scores of -1, 0 or 1 based on the following rules:
**Neutral (0):** *any word that is in all 5 star ratings* any word that is evenly distributed between one + two and four + five stars *any words that is uncategorized by the below rules* ***Negative (-1):*** words that are predominately in one and two stars *words in one, two or three stars and not in four or five stars* ***Positive (1):*** words that are predominately in four and five stars *words in three, four or five stars and not in one or two stars

The second sentiment method classification utilizes a lexicon of positive and negative words available with full source citation at: https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
The files downloaded are named negativeSent.txt and positiveSent.txt

### Sentiment Algorithm Construction

After reviewing available sentiment ranking options, and since the time was available, I chose to construct scoring by considering each individual word used and all star scores associated with a word. My preferred environment for this is Hadoop Hive. As hive queries are not supported in R markdown, queries are described:

create table dsReview (row_num string, reviewid string, stars string,isfunny string, isuseful string, iscool string, reviewtext string) row format delimited fields terminated by ',' stored as textfile;
Load data inpath "hdfs://clusternameremoved//sampledata/tidyreview.csv" into table dsReview;

**split the review text into an array of words and explode the array into rows of words with the stars**
create table dsrawWords as select a.stars, r.wrd as rawWord from dsReview a inner join (select reviewid, SPLIT(reviewtext, '') as reviewwords from dsReview) x lateral view explode(x.reviewwords) r as wrd;

**summarize the stars and counts for multi-instance words**
create table dswordSentiment as select rawWord, length(rawword) as numLetters, count() *as numSeen, max(case when stars = 1 then 1 else 0 end) as hasOne, sum(case when stars = 1 then 1 else 0 end) as numOne, max(case when stars = 2 then 1 else 0 end) as hasTwo, sum(case when stars = 2 then 1 else 0 end) as numTwo, max(case when stars = 3 then 1 else 0 end) as hasThree, sum(case when stars = 3 then 1 else 0 end) as numThree, max(case when stars = 4 then 1 else 0 end) as hasFour, sum(case when stars = 4 then 1 else 0 end) as numFour, max(case when stars = 5 then 1 else 0 end) as hasFive, sum(case when stars = 5 then 1 else 0 end) as numFive, max(case when stars < 1 or stars > 5 then 1 else 0 end) as hasOther, sum(case when stars < 1 or stars > 5 then 1 else 0 end) as numOther, cast(0 as int) as posMatch, cast(0 as int) as negMatch, cast(null as string) as sentiment from dsrawwords group by rawWord, where count() > 1;*

**Assign individual words to positive/negative/neutral as defined in methods, part one**
insert overwrite table dswordsentiment select *, case when hasone + hastwo + hasthree + hasfour + hasfive = 5 then 'neutral' when hasone + hastwo + hasthree > 0 and hasfour + hasfive = 0 then 'negative' when hasone + hastwo = 0 and hasthree + hasfour + hasfive > 0 then 'positive' end as sentiment from dswordsentiment;

**Override the sentiment scoring with the lexicon of positive and negative words**
insert overwrite table dswordsentiment select *, case when sentiment = 'neutral' and posMatch = 1 and negMatch = 0 then 'positive' when sentiment = 'negative' and posMatch = 1 and negMatch = 0 then 'positive' when sentiment = 'neutral' and posMatch = 0 and negMatch = 1 then 'negative' when sentiment = 'positive' and posMatch = 0 and negMatch = 1 then 'negative' else sentiment end as sentiment from dswordsentiment;

**score the contents of the review as a sum of the sentimentscore of all included words**
create table dsresults as select reviewid, stars, isfunny, isuseful, iscool, size(reviewwords) as reviewlength, sentimentscore, isfunny+isuseful+iscool as numinfluence, ROUND(sentimentscore/size(reviewWords), 2) as sentimentweight, (ROUND(sentimentscore/size(reviewWords), 2)*(isfunny+isuseful+iscool)) as influencescore, cast(null as tinyint) as influencestars, cast(null as string) as influencegrade from dssentiment;

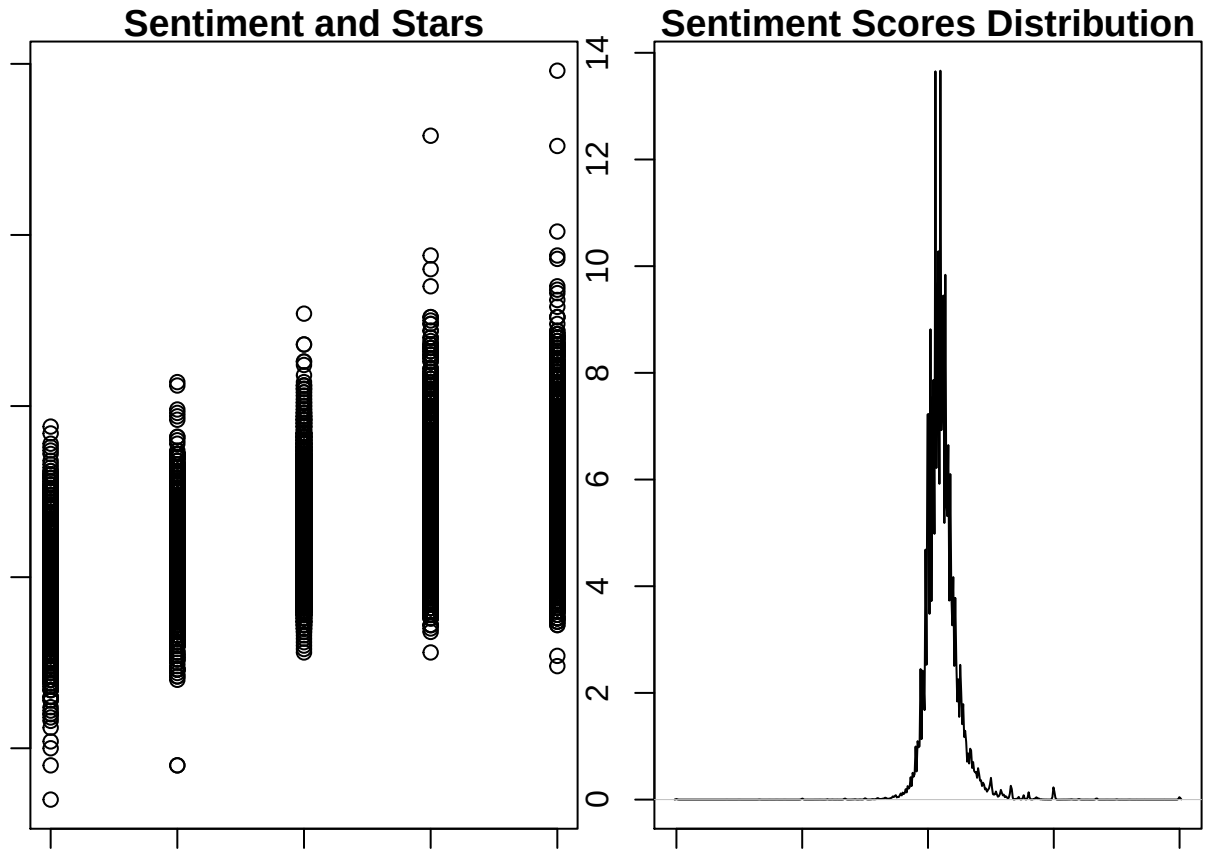
## Methods Phase 2. Prediction of sentiment grade based on classifications produced by step one

The goal of this phase is to define and predict how impactful a review will be to other patrons based on an assumption that the length of the review is as influential as the composition of sentiment. A recursive partitioning model is evaluated for its predictive accuracy classifying the influence grade.

```
data <- read.table("reviewclean.csv", sep="\t")
names(data) <- c("reviewid", "stars", "isfunny", "isuseful", "iscool", "reviewlength", "sentimentscore"
mldata <- data[3:7]; mldata$influencegrade <- data$influencegrade
inTrain <- createDataPartition(y=mldata$influencegrade,p=0.60, list=FALSE)
training <- mldata[inTrain,]; testing <- mldata[-inTrain,]
r_model <- train(influencegrade ~ ., method="rpart", data=training)
#test the model
r_predicted <- predict(r_model, newdata=testing[,-6])
```

**Results Phase 1: Analysis of the distribution of sentiment findings indicates a reasonable approach to the scoring mechanism and an answer to the question as to the ability to find a measure of importance.**

```
par(mfrow=c(1,2), mar= c(1,1,1,1))
plot(data$stars, data$sentimentscore, main="Sentiment and Stars")
plot(density(data$sentimentweight), main="Sentiment Scores Distribution")
```




A review of the words most frequently found in reviews and classified as positive or negative yields good results.

```
sentwords <- read.table("reviewWordFreq.csv", sep="\t")
posWords <- subset(sentwords, sentwords$V2=='positive')
negWords <- subset(sentwords, sentwords$V2=='negative')
par(mfrow=c(1,2), mar= c(0,0,0,0))
wordcloud(posWords$V1, freq=posWords$V3, max.words=250, colors=brewer.pal(8,"Dark2"))
wordcloud(negWords$V1, freq=negWords$V3, max.words=250, colors=brewer.pal(9,"Set1"))
```


**Results Phase 2: Predictive findings for impact of sentiment on consumers is reasonable and possible.**

An attempt follows to use the method of classification and construct the boundaries around the level of influence represented by the sentiment. The manual, hive query based method of scoring the sentiment produces the following results when testing data is trained by the recursive partitioning model.

```
#review the results of the predictive model
r_model$finalModel
```

```
## n= 941560
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 941560 753248 A (0.2 0.2 0.2 0.2 0.2)
##   2) isuseful>=0.5 443874 262535 A (0.41 0.36 0.088 0.029 0.12)
##     4) sentimentscore>=0.5 371541 190202 A (0.49 0.42 0.081 0.0043 0.0023)
##       8) iscool>=0.5 213787  62421 A (0.71 0.27 0.018 0.0038 0.002) *
##       9) iscool< 0.5 157754  57470 B (0.19 0.64 0.17 0.005 0.0027) *
##     5) sentimentscore< 0.5 72333  20027 F (0 0 0.12 0.16 0.72) *
##   3) isuseful< 0.5 497686 322195 D (0.014 0.062 0.3 0.35 0.27) *
```

```
table(r_predicted, testing$influencegrade)
```

```
##
## r_predicted      A      B      C      D      F
##           A 101036  38119   2615    587    253
##           B  19804  67278  17462    593    299
##           C      0      0      0      0      0
##           D   4701  20144  99625 116953  90206
##           F      0      0   5839   7408  34782
```

The results returned by the final model aren't encouraging enough to continue in this direction. There is good correlation in the most and least categories but disappointing results in the middle. As this phase of the report is exploratory in nature, conclusions are that classifying the usefullness of identified sentiment deserves it's own project.

## Discussion

In summary the primary aim of this project was to evaluate text data in reviews and determine if sentiment was identifiable, reasonable and reliable and is there a relationship between the measurable volume of sentiment and the number of customers that respond.

The report evidences that sentiment is identifiable and reasonable and that likely categories of influence level are possible mathmatically. The relationship between the measure of sentiment and the volume of responses was not adequately isolated and must be answered with a maybe.

The final hypothesis of this report resolves with simple logic, If we assume that liking a tip or voting on a review is an indicator of influence and we can reliably identify the review as positive or negative then the prospective customer that responds to the text logically inherits the sentiment of the review or tip.