Modeling

Alex Kan, Jessica Ho, Katherine Wang, Ishan Shah, Svetak Sundhar

April 20, 2018

```
df <- read.csv("data/phoenixAg.csv")

temp <- df %>%
    dplyr::select(-c(businessID, businessName, city))

# Encode average review as 2 if less than 3, or 3 if >= 3
temp$averageReviewBusiness <- ifelse(temp$averageReviewBusiness < 3, 2, 3)
```

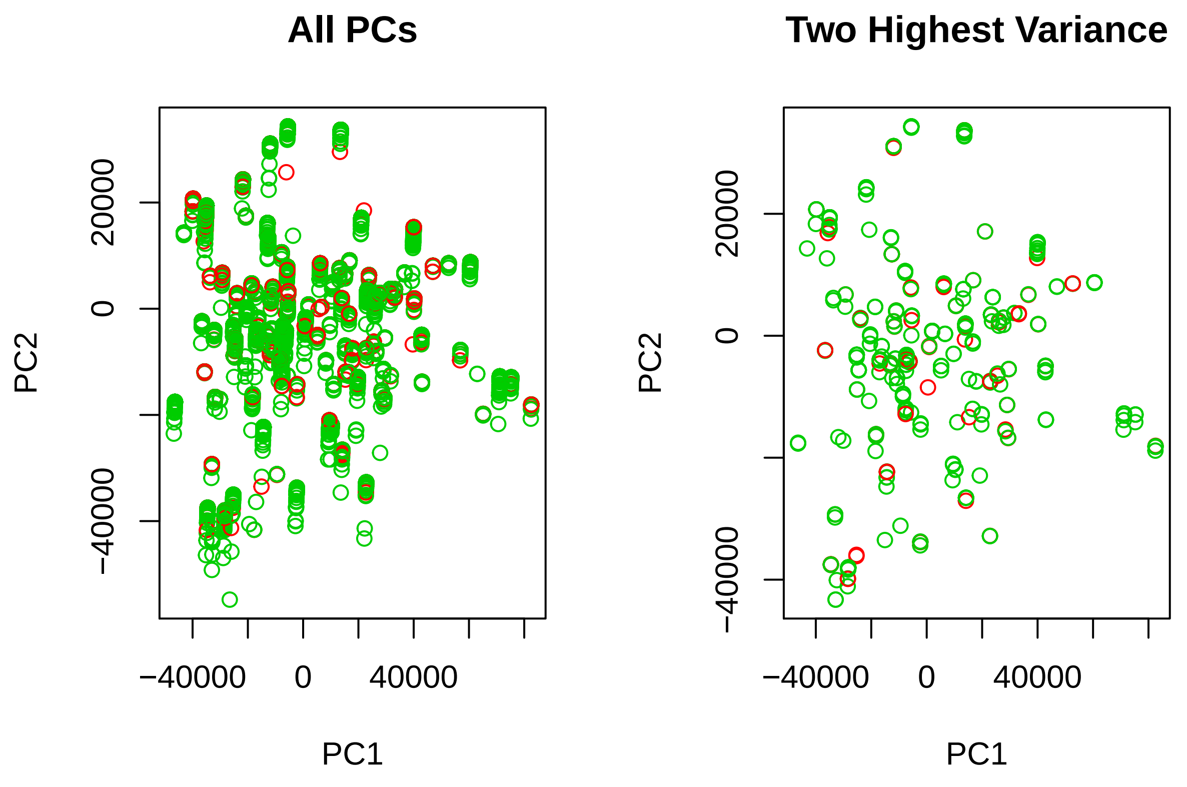
PCA:

```
### More feature selection necessary for PCA
par(mfrow = c(1,2))

pca <- prcomp(temp)

# Sort PCs by highest variance
bestPCs <- sort(pca$sdev, decreasing = T)[1:2]

plot(pca$x[,], col = temp$averageReviewBusiness, xlab = "PC1", ylab = "PC2", main = "All PCs")
plot(pca$x[pca$sdev == bestPCs, ], col = temp$averageReviewBusiness, xlab = "PC1", ylab = "PC2", main =
```



SVM:

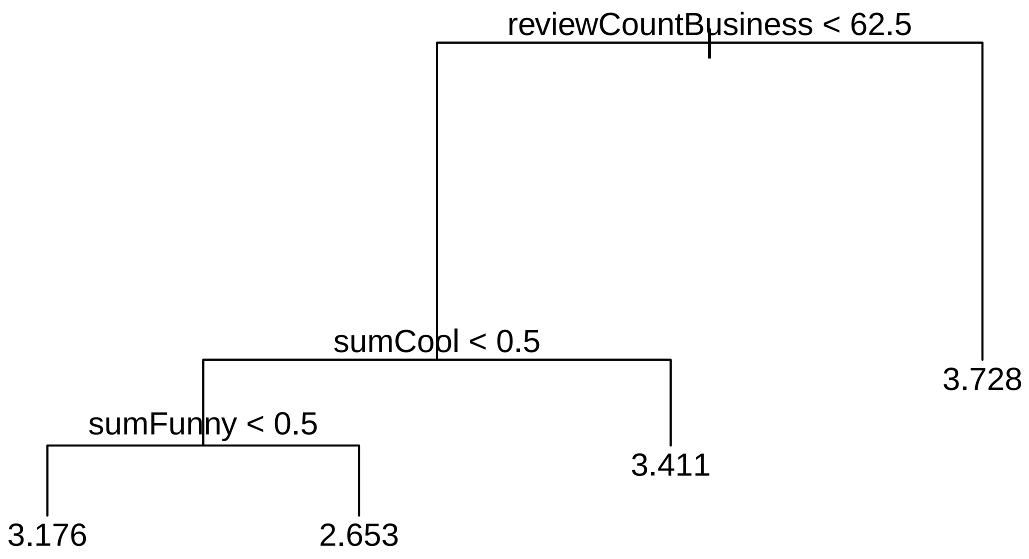
```r
#svm.mod <- svm(data = temp, avgReviewStars~., kernel = "linear", scale = F, cost = .1)
```

Decision Trees:

```r
tree.mod <- tree(data = temp %>%
                     dplyr::select(-c(averageReviewBusiness, avgUserPastReview)), avgReviewStars ~.)

#plot(tree(data = temp,
#     as.formula(paste("avgReviewStars", "~",
#                      paste(colnames(temp)[8:16], collapse = "+"),
#                      sep = ""))))

plot(tree.mod)
text(tree.mod, pretty = 0)
```



Kmeans:

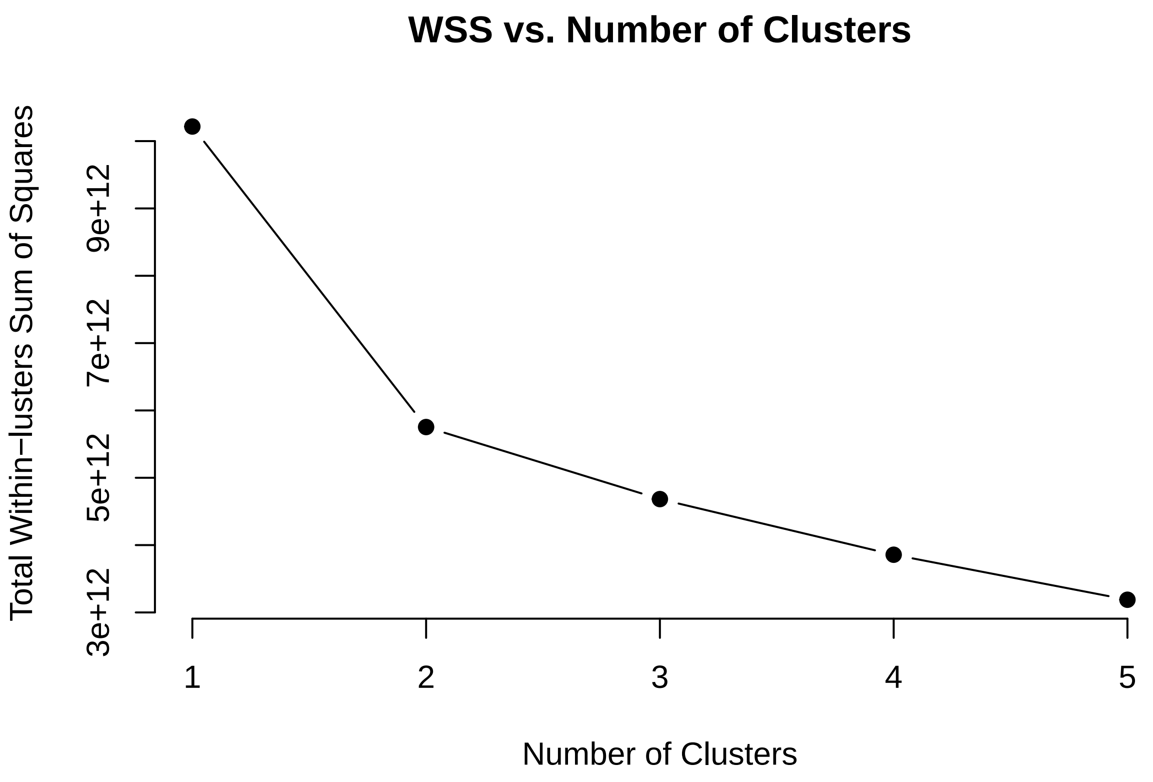
K means supports our justification for splitting reviews into two categories (<3 and >=3). The "elbow" point on the Error vs. K (# of clusters) plot provides evidence.

```r
#euclidianGene <- as.data.frame(cbind(euclidianMeans$cluster, Subtypes))

# Drop start labels and then perform kmeans

# See if k = 5 clusters best models the 5 different star levels
wss <- sapply(1:5,
              function(k){kmeans(temp %>%
                                    dplyr::select(-averageReviewBusiness), k, nstart = 25)$tot.withinss}

plot(1:5, wss,
     type="b", pch = 19, frame = FALSE,
     xlab="Number of Clusters",
     ylab="Total Within-lusters Sum of Squares",
     main="WSS vs. Number of Clusters")
```



```
# confusion matrix
```

Logistic Regression

Quadratic Discriminant Analysis

Quadratic Discriminant Analysis allows for non-linear (quadratic) decision boundaries, unlike Linear Discimrinant Analysis. QDA require the number of predictor variables (p) to be less then the sample size (n). We are assuming predictor variables X are drawn from a multivariate Gaussian (aka normal) distribution and that the covariance matrix can be different for each class so we must estimate the covariance matrix separately for each class. However, QDA is recommended if the training set is very large, so that the variance of the classifier is not a major concern, or if the assumption of a common covariance matrix is clearly untenable.

First, we will define a binary variable for averageReviewBusiness. Ratings greater than 3 are considered good ratings and ratings below 3 are bad ratings.

We will create the training and test sets(80%/20%).

Now, we will run QDA and look at the test MSE (mean squared error).

```r
az <- read.csv("data/phoenixAg.csv")
az$rate <- ifelse(az$averageReviewBusiness < 3, 0,1)

#Create Train & Test Sets
train <- sample(1:(0.80*nrow(az)),replace=FALSE)
az.train <- az[train,]
az.test <- az[-train,]
```

```r
qda.fit <- qda(rate~reviewCountBusiness, data = az.train)
qda.fit
```

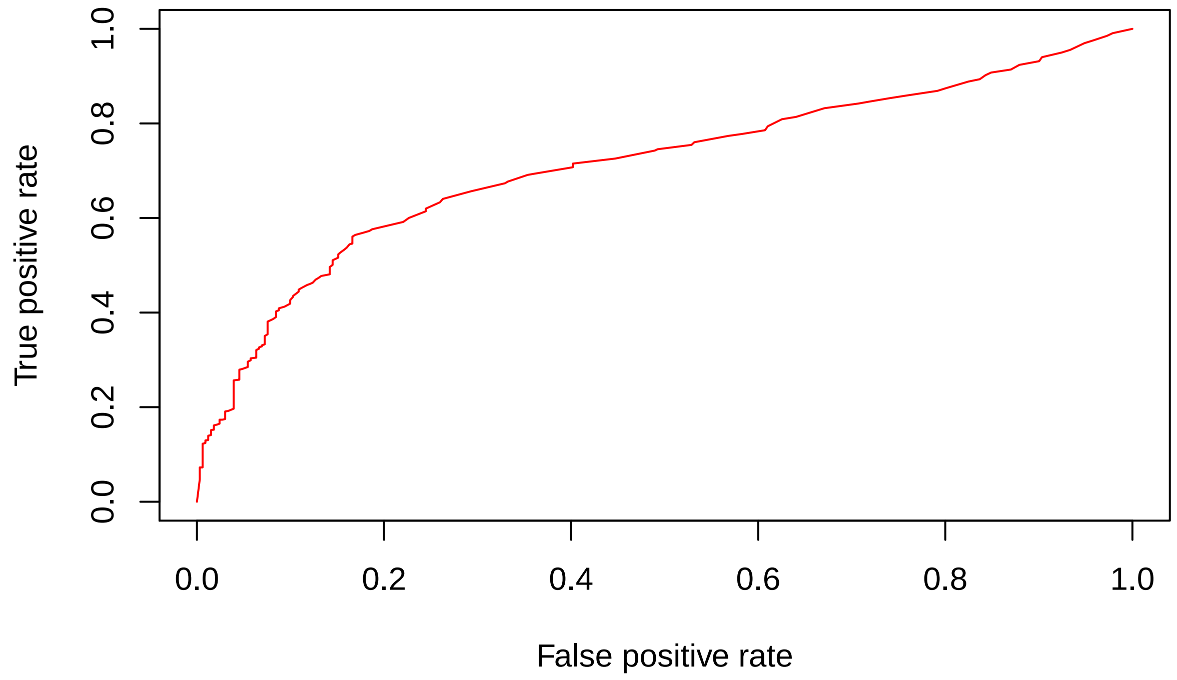
```
## Call:
## qda(rate ~ reviewCountBusiness, data = az.train)
##
## Prior probabilities of groups:
##         0         1
## 0.1756042 0.8243958
##
## Group means:
##   reviewCountBusiness
## 0            33.97557
## 1           117.85794
```

```r
#predict
qda.predict <- predict(qda.fit, newdata=az.test)
qda.class <- qda.predict$class
#Confusion matrix
table(qda.class,az.test$rate)
```

```
##
## qda.class    0    1
##         0    0    0
##         1  331 1418
```

```r
#Overall fraction of incorrect test predictions (MSE: mean squared error)
mean(qda.class != az.test$rate)
```

```
## [1] 0.189251
```



```
## [[1]]
## [1] 0.7128034
```