CS 498 HW7, UIUC

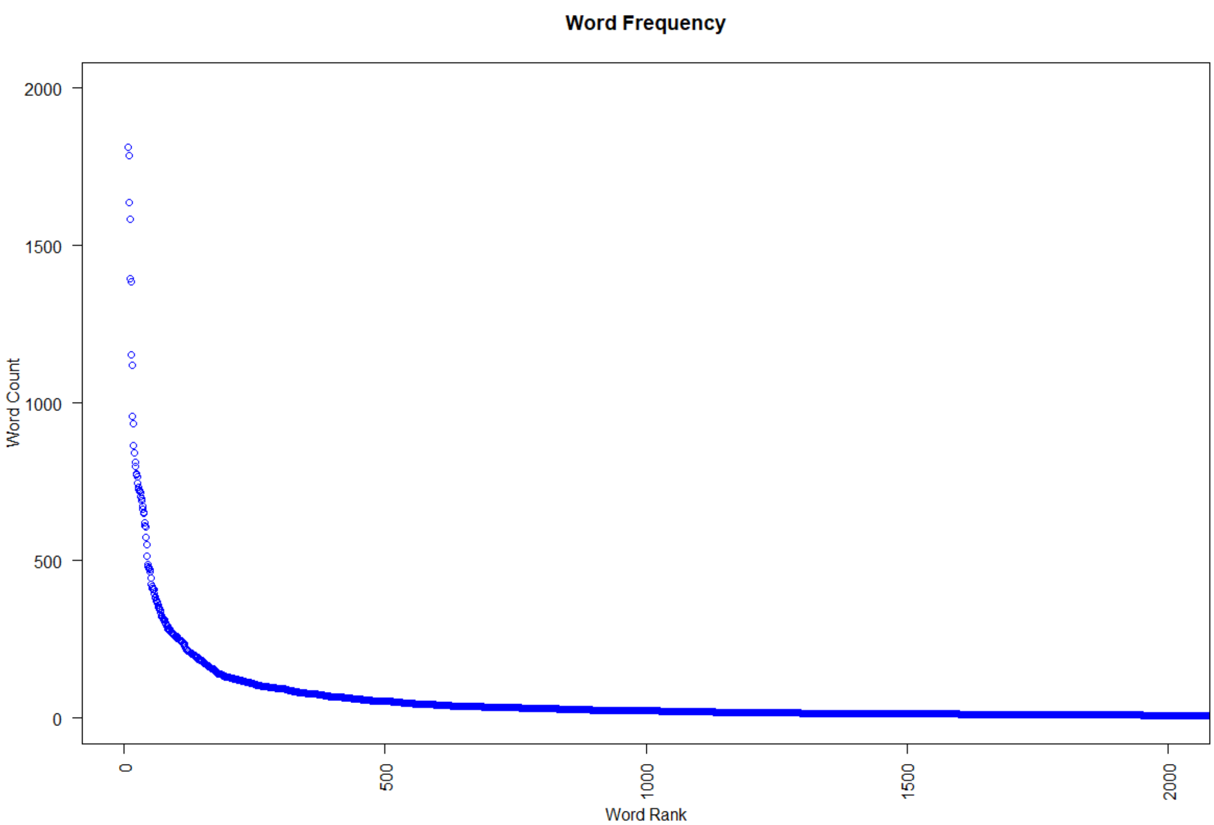
March 24th, 2019

Team:

Apoorva Srinivasa (apoorva6@illinois.edu) , Julia Tcholakova (idt2@illinois.edu)

P1

Word Frequency before removing stopwords



P2

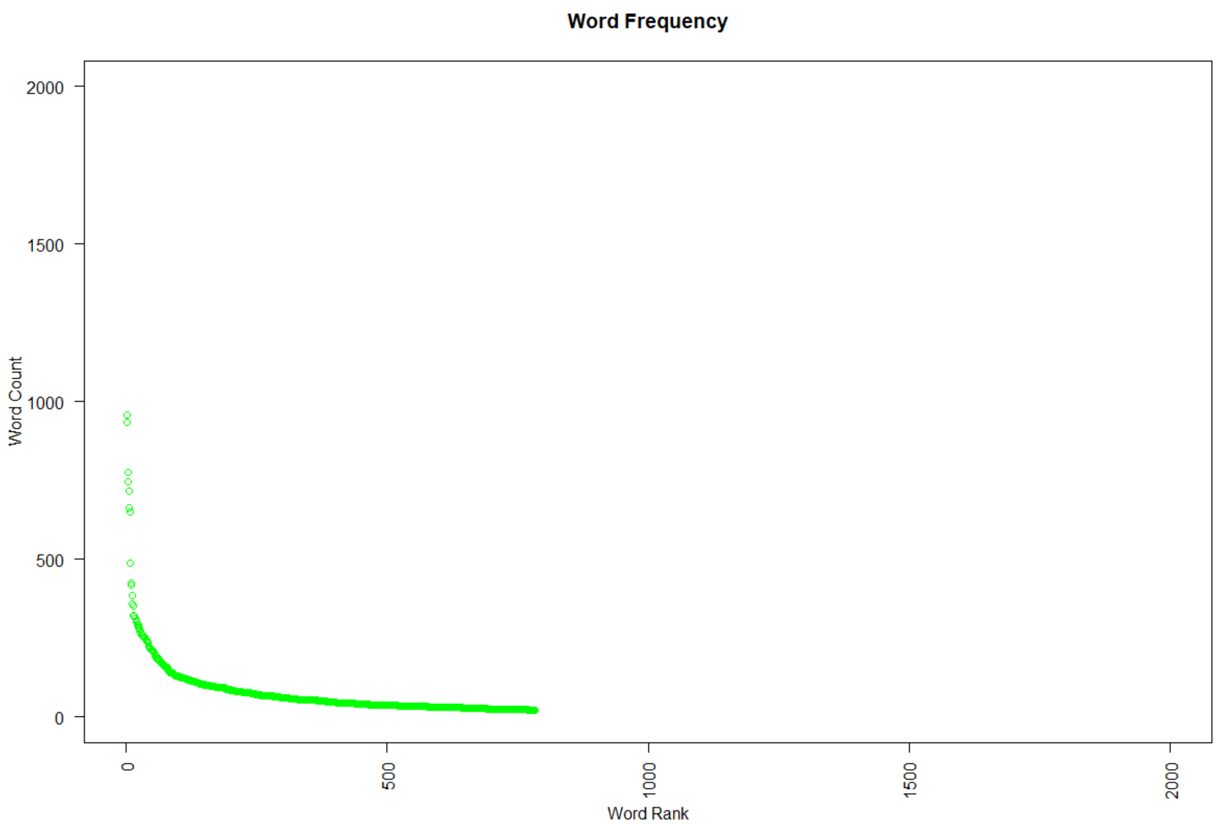
Frequency threshold at 0.99

Stopwords:

"a" "about" "above" "according" "across" "after" "afterwards" "again" "against" "all" "almost" "alone" "along" "already" "also" "although" "always" "am" "among" "an" "and" "another" "any" "anybody" "anyhow" "anyone" "anything" "anyway" "anywhere" "apart" "are" "around" "as" "at" "be" "became" "because" "become" "becomes" "becoming" "been" "before" "beforehand" "behind" "being" "below" "beside" "besides" "between" "beyond" "both" "but" "by" "can" "can't" "cannot" "certain" "choose" "could" "day" "do" "does" "doesn't" "doing" "down" "during" "each" "either" "else" "elsewhere" "enough" "etc" "even" "ever" "every" "everybody" "everyone" "everything" "everywhere" "except" "exception" "exclude" "excluding" "far" "farther" "farthest" "few" "first" "for" "formerly" "forth" "forward" "from" "front" "further" "furthermore" "furthest" "get" "go" "had" "halves" "hardly" "has" "hast" "hath" "have" "he" "hence" "henceforth" "her" "here" "hereabouts" "hereafter" "hereby" "herein" "hereto" "hereupon" "hers" "herself" "him" "himself" "his" "hither" "hitherto" "how" "however" "howsoever" "if" "in" "include" "included" "including" "indeed" "indoors" "inside" "insomuch" "instead" "into" "inward" "inwards" "is" "it" "its" "itself" "just" "kind" "kg" "km" "last" "latter" "latterly" "less" "lest" "let" "little" "ltd" "many" "may" "maybe" "me" "meantime" "meanwhile" "might" "moreover" "most" "mostly" "more" "mr" "mrs" "ms" "much" "must" "my" "myself" "namely" "need" "neither" "never" "nevertheless" "next" "no" "nobody" "none" "nonetheless" "noone" "nope" "nor" "not" "nothing" "notwithstanding" "now" "nowadays" "nowhere" "of" "off" "often" "ok" "on" "once" "one" "only" "onto" "or" "other" "others" "otherwise" "ought" "our" "ours" "ourselves" "out" "outside" "over" "own" "per" "perhaps" "plenty" "provide" "quite" "rather" "really" "round" "said" "sake" "same" "sang" "save" "saw" "see" "seeing" "seem" "seemed" "seeming" "seems" "seen" "seldom" "selves" "sent" "several" "shalt" "she" "should" "shown" "sideways" "since" "slept" "slew" "so" "some" "somebody" "somehow" "someone" "something" "sometime" "sometimes" "somewhat" "somewhere" "spake" "spat" "spoke" "spoken" "still" "such" "than" "that" "the" "thee" "their" "them" "themselves" "then" “thence" "thenceforth" "there" "thereabout" "thereafter" "thereby" "therefore" "therein" "thereof" "thereon" "thereto" "thereupon" "these" "they" "this" "those" "thou" "though" "thrice" "through" "throughout" "thus" "till" "to" "together" "too" "toward" "towards" "unable" "under" "underneath" "unless" "unlike" "until" "up" "upon" "upward" "upwards" "us" "use" "used" "using" "very" "via" "vs" "want" "was" "we" "week" "well" "were" "what" "whatever" "whatsoever" "when" "whence" "whenever" "whensoever" "where" "whereabouts" "whereafter" "whereas" "whereat" "whereby" "wherefore" "wherefrom" "wherein" "whereinto" "whereof" "whereon" "wheresoever" "whereto" "whereunto" "whereupon" "wherever" "wherewith" "whether" "whew" "which" "whichever" "whichsoever" "while" "whilst" "whither" "who" "whoa" "whoever" "whole" "whom" "whomever" "whomsoever" "whose" "whosoever" "why" "will" "wilt" "with" "within" "without" "worse" "worst" "would" "wow" "yet" "year" "you" "your" "yours" "yourself" "yourselves"

P3

Word Frequency after removing stopwords



P4 **Code Snippets**

```
#read the data
yelp_data = read.csv("yelp_2k.csv", header = TRUE, stringsAsFactors=FALSE)
yelp_df = yelp_data[, c("text", "stars")]
colnames(yelp_df) = c("X", "y")
#process text
library(tm)
library(SnowballC)
#reprocess with stopwords
stopword = read.table("stopword.txt", header = FALSE, quote  = "", fill = FALSE)
stopw_vec = stopword$V1
#get counts after removing stopwords
text_corpus = tm_map(text_corpus, removeWords, stopw_vec)
#bag of words vectors
freq_doc_mat = DocumentTermMatrix(text_corpus)
#remove sparse terms, choose max frequency
freq_doc_mat = removeSparseTerms(freq_doc_mat, 0.99)
freq_term_mat = TermDocumentMatrix(text_corpus)
freq_term_mat = removeSparseTerms(freq_term_mat, 0.99)
freq_term_mat = as.matrix(freq_term_mat)
freq_vec = sort(rowSums(freq_term_mat),decreasing = TRUE)
freq_df = data.frame(word = names(freq_vec),freq = freq_vec)
#plot after removing stopwords
plot(freq_df$freq, las = 2, col ="green", main ="Word Frequency", xlab = "Word Rank",
     ylab = "Word Count", xlim = c(0, 2000), ylim = c(0, 2000))
#convert the word vectors to dataframe for next part
reviews_df = as.data.frame(as.matrix(freq_doc_mat))
library(proxy)
query_df = as.data.frame(matrix(0,1, ncol(reviews_df)))
colnames(query_df) = colnames(reviews_df)
names(query_df)[names(query_df) == 'horrible'] = 'Horrible'
#fill in the query df
query_df[, c("Horrible", "customer", "service")] = 1
#calculate cosine distance, less is better
test_vec = rep(0, nrow(reviews_df))
for(i in 1:length(test_vec)){
  test_vec[i] = dist(reviews_df[i,], query_df[1,], method = "cosine")
}

#add the score to yelp_df
yelp_q = cbind(yelp_df, "q_score" = round(test_vec, 5))
yelp_q_ord = yelp_q[order(yelp_q$q_score),]
#1 get top 5 that match
top_5 = yelp_q_ord[1:5,]
#possible matches - if label is 1 and score is < cerain theshold
poss_match = subset(yelp_q, (yelp_q$y == 1 & yelp_q$q_score < 0.62))
```

P5

Screenshot of score results for the original reviews (cosine dist based on the query)

	X	y	q_score
1	This car wash sucks. Paid $40 for the Ultimate VIP. still had dirt all over...	1	0.87961
2	I was referred to Earnie by friends and since then I've referred him ma...	5	1.00000
3	The food is okay, but they have the worst service I've ever seen. If you ...	1	0.71132
4	Opting out from the noise and hustle of Flo's we called in a take-out o...	1	0.88622
5	Basically, unlimited steak. If you like steak, you will like this place. On...	5	1.00000
6	I visited Double Wide for the first time today. My first impression was ...	1	1.00000
7	Last week me and my wife had dinner. Sat for 10 minutes to get a serv...	1	1.00000
8	This place is the best!!! I was recommended by my co worker to get a ...	5	1.00000
9	I could not wait to check OUT! At arrival, the valet did not offer any as...	1	0.86577
10	Horrible customer service! Been with them over 2 years, and after stay...	1	0.53812
11	Terribly disorganized with zero knowledge in customer service. I place...	1	0.85858
12	Don't be fooled by the open seats. The pho was fantastic and authenti...	5	1.00000
13	*WARNING* this restaurant is under new management and it's shot to...	1	1.00000
14	So, it's great that this massive biergarden\/resto has activated Chester...	1	0.92323
15	The food is "ok," but I LOVE the service. It is a small mom and pop ty...	5	0.65184
16	Really disappointed with the sushi. Deep fried rolls were way over frie...	1	1.00000
17	This place a is the worst and the most of the checkers are so dumb. T...	1	1.00000
18	Dr Cho does a great job. Staff and hygienists are also excellent Eventh...	5	1.00000
19	This is a general overview of our experience with Tire Works. We had ...	1	0.96453
20	I have been a long time customer of Elaine's and am so happy for thei...	5	0.74180
21	I was super excited to try this place out. I've lived in Arizona my entire ...	1	1.00000
22	The service is super friendly, the food tastes amazing and the prices ar...	5	0.91393
23	This restaurant exceeds expectations! The food is amazing and the ser...	5	0.81743
24	BEWARE!! We just signed up and was told we could bring a friend for f...	1	1.00000
25	Arrival at 11 am. Now it is 4 pm. Still not seen by doctor. My sister is cr...	1	1.00000
26	The best Cirque show I've ever had the pleasure to see. Like most othe...	5	1.00000
27	Walked in to only rudely get turned away. Had called earlier in the day...	1	1.00000
28	Elvira did an excellent job with my facial. Great information, great pro...	5	1.00000
29	Emailed like 5 times - finally got some straight answers but alas the m...	1	1.00000
30	I came here during Phoenix Comic Con, I was planning on going here...	1	1.00000

Showing 1 to 30 of 2,000 entries

P6

Original reviews for top 5 as per query

The following shows the top 5 reviews based on the minimum cosine distance between the query “Horrible customer service” and the original reviews.
These reviews represent best 5 matches for the given query, yielding the min cosine distance.

Index	Review	Stars	Score
1809	Rogers ...1) is over priced 2) have horrible customer service 3) faulty and incorrect billing 4) poor customer service 5) not enough options 6) never arrive for an appointment	1	0.19936
91	Horrible service, horrible customer service, and horrible quality of service! Do not waste your time or money using this company for your pool needs. Dan (602)363-8267 broke my pool filtration system and left it in a nonworking condition. He will not repair the issue he caused, and told me to go somewhere else. Save yourself the hassle, there are plenty of other quality pool companies out there. Take care!	1	0.39073
1430	"They shut down. Makes sense, they had terrible service and subpar food.. should have listened to your customer base."	1	0.42265
1724	The service is horrible. It's not bad inside, but really one of the most annoying clubs in Vegas. I'm all for Vegas clubs, but service here sucks.	1	0.45228
730	"Service was horrible came with a major attitude. Payed 30 for lasagna and was no where worth it. Won't ever be going back and will NEVER recommend this place. was treated absolutely horrible. Horrible."	1	0.4836

P7

Code for classifier

```
#We have chosen to work with h2o logistic regression as this yielded the best accuracy results.
#separate into train and test data
reviews_df$y = yelp_df$y
train_idx = createDataPartition(y = reviews_df$y, p = 0.90, list = FALSE)
train_df = reviews_df[train_idx,]
test_df = reviews_df[-train_idx,]
#convert (1, 5) to (0, 1)
train_df$y = ifelse(train_df$y == 1, 0, 1)
test_df$y = ifelse(test_df$y == 1, 0, 1)
library(h2o)
h2o.init(ip = 'localhost', port = 54321, max_mem_size = '2650m')
#function to return accuracy and predictions
calcAccuracy = function(mod, test_data, y){
  res_ls = list()
  test_pred = h2o.predict(object = mod, newdata = test_data[, -y])
  acc_test = mean(test_pred[, 1] == test_data[, y])
  res_ls = c(test_pred, acc_test)
  return(res_ls)
}
#create dataframes for h2o input
train_hex = as.h2o(train_df, destination_frame = "t.hex")
test_hex = as.h2o(test_df, destination_frame = "t_test.hex")
num_col = dim(train_hex)[2]
#logistic regression model
mod_h2o = h2o.glm(x = 1:num_col-1,
                  y = num_col, training_frame = train_hex,
                  seed = 457124,
                  family = "binomial",
                  lambda_search = TRUE,
                  alpha = 0.5,
                  nfolds = 5)
train_res = calcAccuracy(mod_h2o, train_hex, num_col)
train_pred_df = as.data.frame(train_res[[1]])
train_acc = train_res[[2]]
test_res = calcAccuracy(mod_h2o, test_hex, num_col)
test_pred_df = as.data.frame(test_res[[1]])
test_acc = test_res[[2]]
```

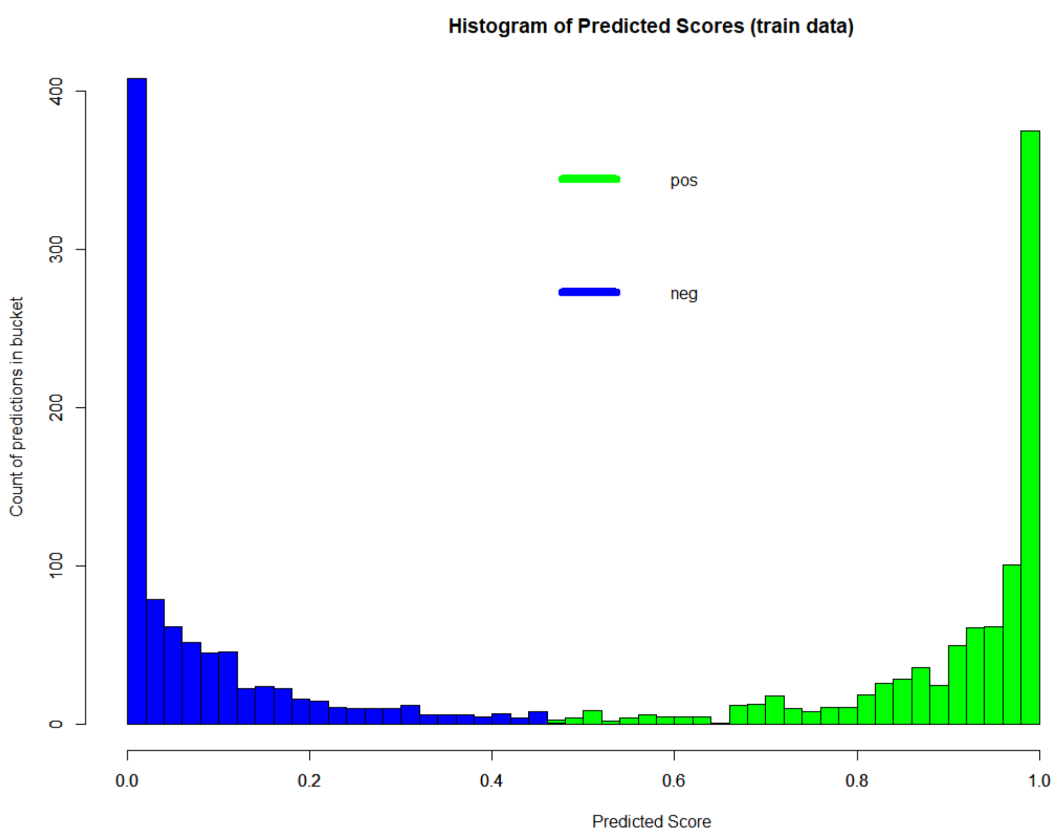
Accuracy for train and test data with threshold 0.5

train accuracy	0.986111
test accuracy	0.925

P8

Code for predicted scores:

```
#scores on training data
train_pred_0 = subset(train_pred_df, train_pred_df$predict == 0)
train_pred_1 = subset(train_pred_df, train_pred_df$predict == 1)
#histogram train data
hist(train_pred_1$p1, col = "green", xlim = c(0, 1.15), ylim = c(0, 400), breaks = 30,
    main = "Histogram of Predicted Scores (train data)", xlab = "Predicted Score", ylab = "Count
of predictions in bucket")
par(new = TRUE)
hist(train_pred_0$p1,  col = "blue", xlim = c(0, 1.15), ylim = c(0, 400), breaks = 30,
    main = "", xlab = "", ylab = "")
legend("top", legend = c("pos", "neg"), col = c("green", "blue"), lty=c(1, 1), bty = "n",
      lwd = 8, seg.len = 1)
```



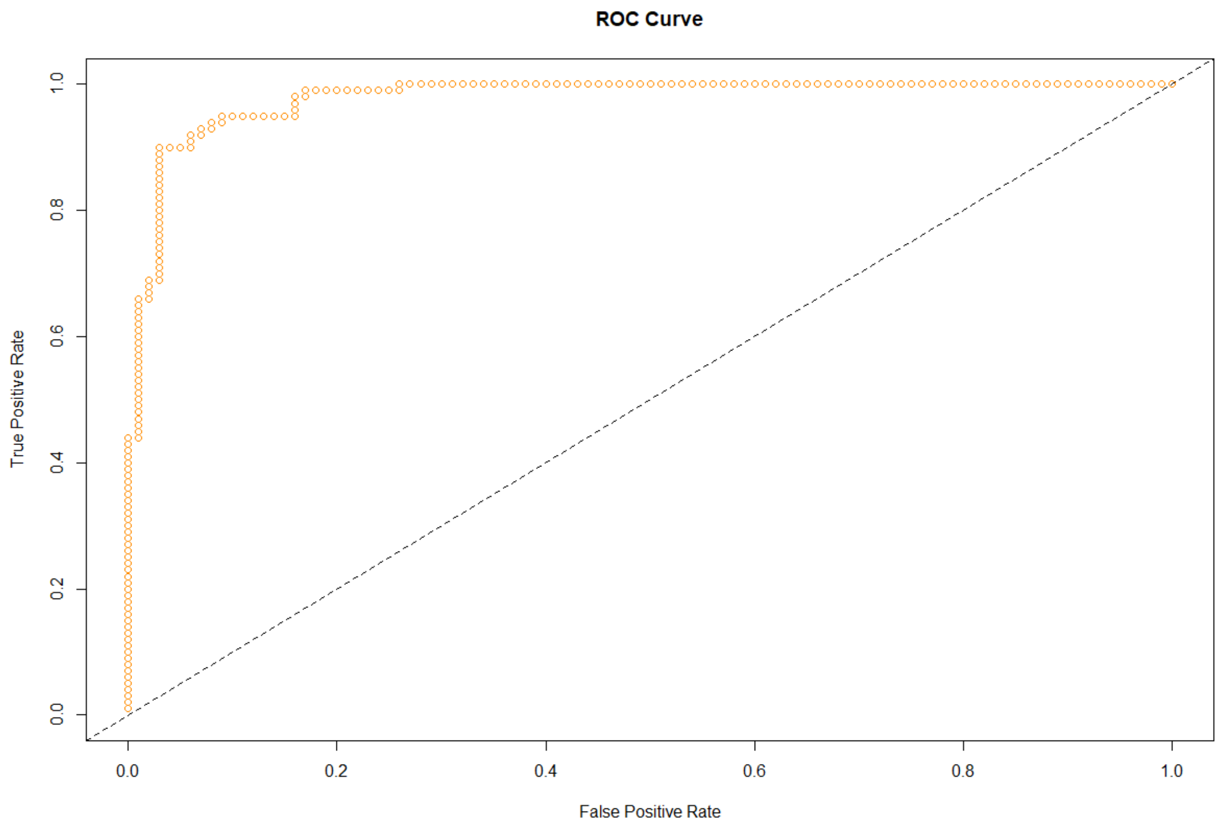
P9

Change threshold to **0.6**

Accuracy for train and test data with threshold 0.6, that threshold yielded better train accuracy, but the test accuracy was lower, thus for the ROC a threshold of 0.5 was chosen

train accuracy	0.99035
test accuracy	0.9124

ROC, based on the better model, which is with threshold **0.5**



P10

Looking at the ROC curve, we could choose FP rate at around 0.25 where we would have the max TP rate.
We would suggest that a more balanced approach in the case of this particular dataset would be to choose FP rate somewhere between 0.17-0.18, where we would still have a very high TP rate, but smaller FP rate.