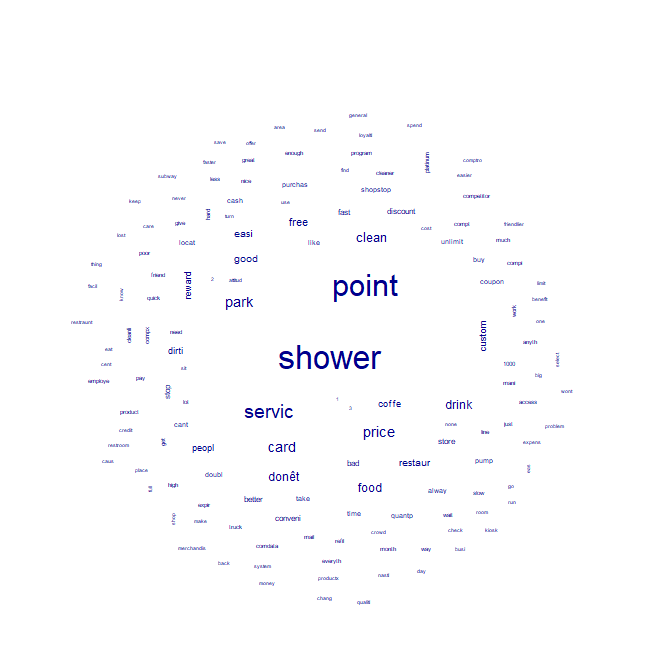
NLP

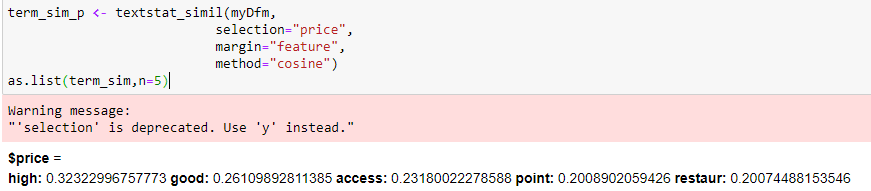
1. Provide the word cloud after all necessary pre-processing.

* For preprocessing, we removed the stopwords.
* We observed Tokens like ‘can’,’use’,’get’,’productx’ are not useful and removed the tokens.
* After removing the stopwords, we considered trimming myDfm with minimum of 4 words per document and with term frequency to be minimum of 2.



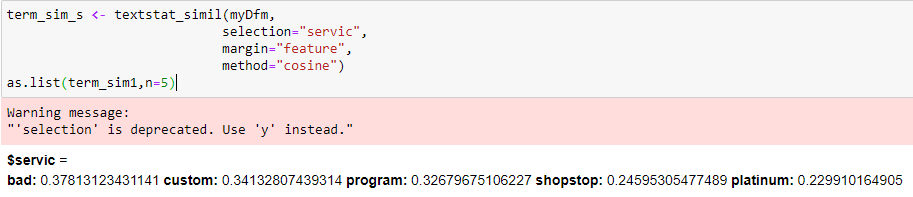
1. What are the top 5 terms that are most related to “price”? Please specify your similarity measurement method and detailed results.

* I used Cosine as my Similarity measurement method.
* The top 5 terms that are most related to price: high, good, access, point, restaur



1. What are the top 5 terms that are most related to “service”? Please specify your similarity measurement method and detailed results.

* I used Cosine as my Similarity measurement method.
* The top 5 terms that are most related to service: bad, custom, program, shopstop, platinum.

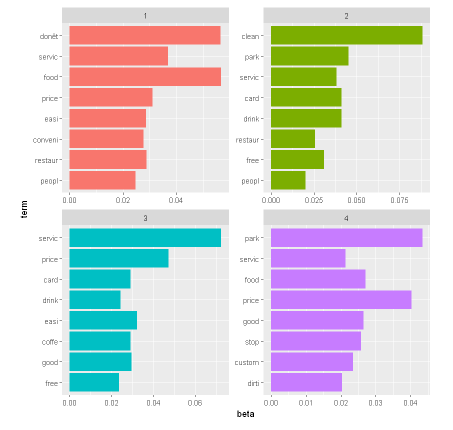


1. Perform topic modeling with 4 topics

* Removed the common words: ‘shower’ and ‘point’.
* Performed topic modelling with number of terms=8.

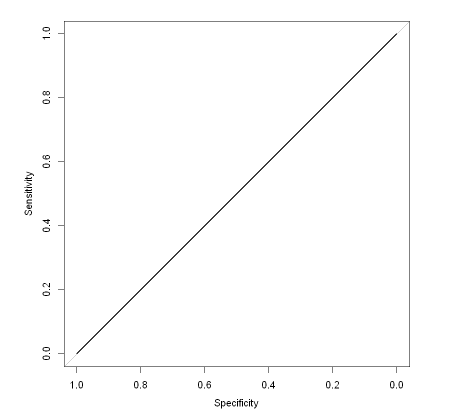
Summary of Four topics:

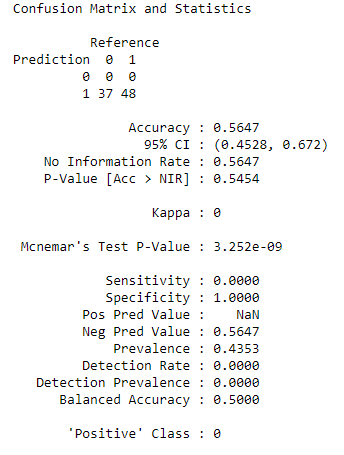
1. For Topic 1: We observe that it has convenience price and don’t have food restaurant.
2. For Topic 2: We observe that it is a clean restaurant, free parking, accept cards, offers drinks.
3. For topic 3: known for good service, offers free coffee, accepts cards and offers drink.
4. For Topic 4: we observe it has good food, dirty service, pricy parking.



1. Please run two decision tree models
   * Model 1 only uses non-text information (i.e., all other columns ***except the Comment column***)

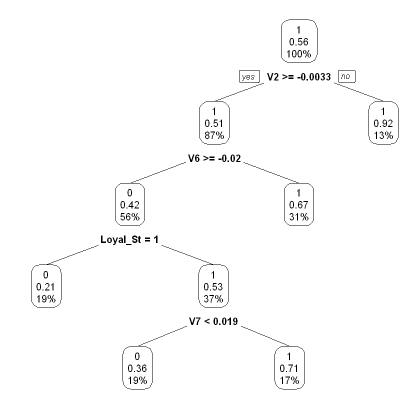


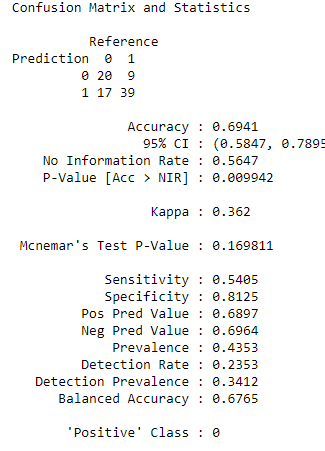


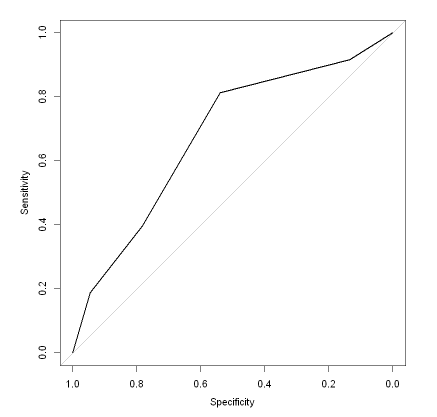




* + Model 2 combines both non-text and text information
    - Text mine the ***Comment*** column
    - Apply SVD to extract text information from the ***Comment*** column
    - Keep the number of SVD as 8
    - Combine 8 SVD with all other columns ***except the Comment column***









From the Above two models, from the confusion matrix, we see model1 has an accuracy of 56% and model2 has an accuracy of 69%, Hence we conclude model2 performed better compared to model1.

**R code:**

library(quanteda)

library(ggplot2)

library(stopwords)

library(topicmodels)

library(tidytext)

library(ggplot2)

library(dplyr)

library(pROC)

library(rpart)

library(caret)

library(rpart.plot)

library(quanteda.textmodels)

df <-read.csv('gastext.csv',stringsAsFactors = F)

df

myCorpus <- corpus(df$Comment)

summary(myCorpus)

myDfm <- dfm(myCorpus)

dim(myDfm)

View(myDfm)

tstat\_freq <- textstat\_frequency(myDfm)

head(tstat\_freq, 20)

myDfm %>% textstat\_frequency(n = 20) %>%

ggplot(aes(x = reorder(feature, frequency), y = frequency))+ geom\_point()+

labs(x =NULL, y = "Frequency")+ theme\_minimal()

myDfm <- dfm(myCorpus,

remove\_punc = T,

remove = c(stopwords("english")),

stem = T)

dim(myDfm)

tstat\_freq <- textstat\_frequency(myDfm)

head(tstat\_freq, 20)

stopwords<-c('can','use','get','productx')

myDfm <- dfm(myCorpus,

remove\_punc = T,

remove=c(stopwords('english'),stopwords),

stem = T)

tstat\_freq <- textstat\_frequency(myDfm)

head(tstat\_freq, 20)

myDfm<- dfm\_trim(myDfm,min\_termfreq=4, min\_docfreq=2)

dim(myDfm)

textplot\_wordcloud(myDfm,max\_words=200,)

topfeatures(myDfm,30)

term\_sim\_p <- textstat\_simil(myDfm,

selection="price",

margin="feature",

method="cosine")

as.list(term\_sim\_p,n=5)

term\_sim\_s <- textstat\_simil(myDfm,

selection="servic",

margin="feature",

method="cosine")

as.list(term\_sim\_s,n=5)

myDfm <- dfm\_remove(myDfm, c('shower','point'))

myDfm <- as.matrix(myDfm)

myDfm

myDfm <-myDfm[which(rowSums(myDfm)>0),]

myDfm <- as.dfm(myDfm)

myLda <- LDA(myDfm,k=4,control=list(seed=101))

myLda

myLda\_td <- tidy(myLda)

myLda\_td

top\_terms <- myLda\_td %>%

group\_by(topic)%>%

top\_n(8, beta) %>%

ungroup() %>%

arrange(topic, -beta)

top\_terms %>%

mutate(term = reorder(term, beta)) %>%

ggplot(aes(term, beta, fill = factor(topic)))+

geom\_bar(stat = "identity", show.legend =FALSE)+

facet\_wrap(~ topic, scales = "free")+

coord\_flip()

df[,3:15]<-lapply(df[,3:15],factor)

str(df)

dfSA <- df[,3:15]

str(dfSA)

set.seed(101)

trainIndex <- createDataPartition(dfSA$Target, p=0.7, list=FALSE, times=1)

dfSA.train <- dfSA[trainIndex,]

dfSA.valid <-dfSA[-trainIndex,]

treeSA <- train(Target~., data=dfSA.train, method="rpart", na.action=na.pass)

treeSA

prp(treeSA$finalModel,type=2,extra=106)

prediction <- predict(treeSA,newdata=dfSA.valid,na.action = na.pass)

confusionMatrix(prediction,dfSA.valid$Target)

tree.probabilities <- predict(treeSA,newdata=dfSA.valid,type='prob',na.action=na.pass)

treeSA.ROC <- roc(predictor=tree.probabilities$`1`,

response=dfSA.valid$Target,

levels=levels(dfSA.valid$Target))

plot(treeSA.ROC)

treeSA.ROC$auc

modelDfm <- dfm(myCorpus,

remove\_punc = T,

remove=c(stopwords('english'),stopwords),

stem = T)

dim(modelDfm)

modelDfm<- dfm\_trim(modelDfm,min\_termfreq=4, min\_docfreq=2)

dim(modelDfm)

modelDfm\_tfidf <- dfm\_tfidf(modelDfm)

modelSvd <- textmodel\_lsa(modelDfm\_tfidf, nd=8)

head(modelSvd$docs)

dim(modelSvd$docs)

dfSA\_two<- cbind(dfSA,as.data.frame(modelSvd$docs))

str(dfSA\_two)

summary(dfSA\_two)

trainIndex <- createDataPartition(dfSA\_two$Target,

p=0.7,

list=FALSE, times=1)

dfSA\_two.train <- dfSA\_two[trainIndex,]

dfSA\_two.valid <-dfSA\_two[-trainIndex,]

tree2SA.model <- train(Target~.,

data=dfSA\_two.train,

method="rpart",

na.action=na.pass)

tree2SA.model

prp(tree2SA.model$finalModel,type=2,extra=106)

prediction2 <- predict(tree2SA.model,

newdata=dfSA\_two.valid,

na.action = na.pass)

confusionMatrix(prediction2,dfSA\_two.valid$Target)

tree2SA.probabilities <- predict(tree2SA.model,

newdata=dfSA\_two.valid,

type='prob',

na.action=na.pass)

tree2SA.ROC <- roc(predictor=tree2SA.probabilities$`1`,

response=dfSA\_two.valid$Target,

levels=levels(dfSA\_two.valid$Target))

plot(tree2SA.ROC)

tree2SA.ROC$auc