```
#install.packages('tibble')
#library(tibble)
#library(doBy)
#library(aod)
#library(ggplot2)
#library(Rcpp)
#library(mclust)
#require(colorspace)
#require(sampling)
#require(MASS)
library(plyr)
library(dplyr)
require(nnet)
require(caret)
require(gridExtra)
require(xtable)
require(corrplot)
install.packages('corrplot')
require(car)

# this is basically from this video
#   https://www.youtube.com/watch?v=fDjKa7yWk1U

###Note that I added a few extra columns to booksDF.  The only extra one currently used is the AuthorID and rndselect used to pick the 2nd book for training data set.

# Read input file
dirname <- c('C:/Users/anobs/Documents/GitHub/MSDS_6372_Project_3_Adverbs/data/')
inputFile <- c('booksDF2.csv')
adverbs = read.csv(paste(dirname,inputFile, sep=''),header=TRUE)
str(adverbs)

adverbs[is.na(adverbs)] <- 0                                #Set any missing values to 0
adverbs$recnum <- as.numeric(rownames(adverbs))             #Add a rownumber field
adverbs$AuthorID_Factor <- factor(adverbs$AuthorID)         #Create a factor variable for AuthorID
adverbs$out <- relevel(adverbs$AuthorID_Factor, ref ='1')   #Create a referance variable using authorID=1

# Create a partition by Author for training dataset,  This function seems to make sure we get one of each author in the training file
# even with a very low percent, it will still pick at least one record per Author
(TrainIndex <- createDataPartition(adverbs$Author,p=0.6, list = F))

# Create test and training datasets
(train <- adverbs[TrainIndex,])
(test <- adverbs[-TrainIndex,])

# Write an output file wtih the Authors and Books in the test dataset
(BooksInTestset <- as.data.frame(adverbs.test[,c('recnum','Author', 'AuthorID','Title')]))
(outfile <- paste(dirname,'BooksInTestSet.csv', sep=''))
write.csv(BooksInTestset,file = outfile , row.names = TRUE)

str(train)
str(test)


# This section is different models, run one then skip to summary()
#mymodel <- multinom(out~ Per_Small+Per_Medium+Per_Large + little+ without+  other+ nothing+ again +before + these + least+  about+  those +though + after+ through+
together + where+  under + never+  right, data = train)
mymodel <- multinom(out~ little+ without+  other+ nothing+ again +before + these + least+  about+  those +though + after+ through+ together + where+  under + never+
right, data = train)
```

```
mymodel <- multinom(out~ little+ without+  other+ nothing+ again +before + these + least+  about+  those , data = train)
mymodel <- multinom(out~ about + little + these +  again +  other + right +those , data = train)
mymodel <- multinom(out~ about + little + these +  again +  other + right  , data = train)
mymodel <- multinom(out~ about + little + these +  again +  other  , data = train)
mymodel <- multinom(out~ about + little + these +  again , data = train)
mymodel <- multinom(out~ little+ without+ other  , data = train)

class(mymodel)
summary(mymodel, Wald = TRUE)
exp(coef(mymodel))

pred = predict(mymodel, newdata=test)
accuracy <- table(pred, test[,"Author"])
# Calculate prediction accuracy
sum(diag(accuracy))/sum(accuracy)

#anova(mymodel, mymodel2, test = 'Chisq')


mostImportantVariables <- varImp(mymodel,value = "rss")
mostImportantVariables <- varImp(mymodel)
mostImportantVariables$Variables <- row.names(mostImportantVariables)
(mostImportantVariables <- mostImportantVariables[order(-mostImportantVariables$Overall),])

z <- summary(mymodel)$coefficients/summary(mymodel)$standard.errors
p <- (1-pnorm(abs(z),0,1)) *2
p

mymodel2 <- multinom(out~ about + little + these + other + right + again + those + never  , data = train)


(ConfidenceMatrix <- table(predict(mymodel2),train$AuthorID))
(CheckPredictions <- predict(mymodel2, type = "class", newdata =test))
summary(mymodel2)

mostImportantVariables2 <- varImp(mymodel2,value = "rss")
mostImportantVariables2 <- varImp(mymodel2)
mostImportantVariables2$Variables <- row.names(mostImportantVariables2)
(mostImportantVariables2 <- mostImportantVariables2[order(-mostImportantVariables2$Overall),])

z <- summary(mymodel2)$coefficients/summary(mymodel2)$standard.errors
p <- (1-pnorm(abs(z),0,1)) *2
p

# extract the coefficients and update external file
as.data.frame(coef(mymodel))
(outfile <- paste(dirname,'SummaryModel.csv', sep=''))
write.csv(as.data.frame(coef(mymodel)),file = outfile , row.names = TRUE)

M <- cor(train[,6:32])
corrplot(M, method='circle')

# This produces a table of p values showing significance for each adverb predicting a given authorID.  Need to figure out how to find minimal number of best predicting
adverbs
z <- summary(mymodel)$coefficients/summary(mymodel)$standard.errors
p <- (1-pnorm(abs(z),0,1)) *2
p
exp(coef(mymodel))
```

```
str(summary(mymodel))
row.names(coef(mymodel))

names(coef(mymodel))
predictors(mymodel)
class(mymodel)
summary(mymodel)  # getting NaNs here, some sites say don't worry about them, not sure about this

#predict(mymodel,train)  #this is a list of predictions, hard to read so skip it
#predict(mymodel,train,type ="prob")  #This gives probabilities, kind of hard to read
summary(mymodel)
predict(mymodel)
options(digits=4)

(CheckPredictions <- predict(mymodel, type = "class", newdata =test))


(ConfidenceMatrix <- table(predict(mymodel),train$AuthorID))
ConfidenceMatrix <- table(predict(mymodel),train$AuthorID_Factor)
# This creates a matrix that shows the predicted author vs the actual author
# Perfect match is when the number of books by the author is in the intersection of predicted vs actual authorID
#                           row variable     column variable

(misclassificationpcterror <- 1-sum(diag(ConfidenceMatrix))/sum(ConfidenceMatrix))



varImp(mymodel,value = "rss")
varImp(mymodel,value = "pls")
mostImportantVariables <- varImp(mymodel,value = "rss")
mostImportantVariables <- varImp(mymodel)
mostImportantVariables$Variables <- row.names(mostImportantVariables)
(mostImportantVariables <- mostImportantVariables[order(-mostImportantVariables$Overall),])
print(head(mostImportantVariables))

# having trouble getting a plot to work
#
plot(mostImportantVariables)
#
g <-plot(Y=mostImportantVariables$Overall,main = 'Variable Importance Plot', xlab = 'x', ylab = 'y')
g + axis(side = 2,1:length(mostImportantVariables$Variables),labels =mostImportantVariables$Variables)
g
#par(las=2)
barplot(mostImportantVariables$Overall,horiz=TRUE,names.arg=mostImportantVariables$Variables)




# https://www.youtube.com/watch?v=qkivJzjyHoA&t=6s
# from ordinal logistic regression video, doesn't seem to work here the same way
(ctable <- coef(summary(mymodel)))
p <- pnorm(abs(ctable[, 't value']), lower.tail = FALSE) *2
(ctable <- cbind(ctable, 'p value' = p))



postResample(vehiclesTest$cylinders,preds2)
```

```
#########################################################################################################
#########################################################################################################
#########################################################################################################
#########################################################################################################
#adverbs = read.csv("C:/Users/anobs/Documents/GitHub/MSDS_6372_Project_3_Adverbs/data/booksDF2.csv",header=TRUE)
#adverbs = read.csv("C:/Users/anobs/Documents/GitHub/MSDS_6372_Project_3_Adverbs/data/booksDF2normalized.csv",header=TRUE)

#adverbs$Author_Factor <- factor(adverbs$Author)         #Create a factor variable for AuthorID  Shouldnt need this

(ConfidenceMatrix <- table(predict(mymodel),train$AuthorID))
(CheckPredictions <- predict(mymodel, type = "class", newdata =test))
(ConfidenceMatrix <- table(predict(mymodel),train$AuthorID))
ConfidenceMatrix <- table(predict(mymodel),train$AuthorID_Factor)


correlationmatrix <- cor(adverbs[6:32])
(highlycorrelated <- findCorrelation(correlationmatrix,cutoff=0.5, names = TRUE))
class(highlycorrelated)

step(mymodel)

step(mymodel, scope =  ~.^2)

# scope = ~.^2does all interactions
# step(mymodel, scope =  ~.^2)
#
# Call:
#   multinom(formula = out ~ Per_Large + other + though + Per_Small,
#            data = train)
#
# Coefficients:
#   (Intercept) Per_Large   other  though Per_Small
# 2       -828.4    -466.8  4820.9  1460.4    -468.1
# 3       -691.8     447.6  1215.1  1270.9     687.8
# 4       1143.2   -4535.9  1001.1  2890.4   -1199.3
# 5       1122.0    -732.7 -1114.5 -1983.3   -1613.2
# 6        709.8     -34.2   523.6   910.1   -1804.8
# 7       -471.1     348.3   743.0  -444.2     690.4
# 8      -1128.2    1962.4   939.9   221.0    1060.2
# 9        395.7    2843.0   214.9  -440.1   -2616.2
# 10       -67.0    2955.6 -2708.0  2155.9   -1234.4
# 11       -96.4    2671.1  1765.8 -2441.5   -1819.9
# 12     -1519.5    1783.6  5700.7 -1354.0    -535.9
# 13     -1410.1   -2297.5  6205.8 -1778.5     801.1
# 14     -1147.4     383.4  1289.5   960.6    1744.3
# 15      -278.8    -334.0  1037.3 -3888.3     798.0
# 16       377.9    2168.3 -2981.2  -817.0   -1217.4
#
# Residual Deviance: 9.824
# AIC: 159.8
#
#
#
# test <- adverbs[adverbs$RndSelect > 3,]     #I couldn't figure out how to select 1 book from each author as a training data set so I added a column and pick the 2nd
book as training set

#train <- adverbs[adverbs$RndSelect <= 3,]      #The training data set
```

```
#train <- adverbs        #For now using all the books in the adverb file, comment this out to use the actual test array of books not in the training set
#test <- train
# Setup a random list of books to pull out for the test dataset
#TestSize <- 4
#BookIds <- seq(from=1, to = max(adverbs$recnum), by=1)
#(BookIdsInTestset <- sample(BookIds, size = TestSize, replace = FALSE))
# Print the books in the test dataset and write a .csv for use in Tableau
#(BooksInTestset <- as.data.frame(c(adverbs[BookIdsInTestset,c('recnum','Author', 'AuthorID','Title')])))
#BooksInTestset <- BooksInTestset[order(BooksInTestset$AuthorID),]
#(outfile <- paste(dirname,'BooksInTestSet.csv', sep=''))
#write.csv(BooksInTestset,file = outfile , row.names = TRUE)

# Build the test and training datasets
#(test <- adverbs[adverbs$recnum %in% BookIdsInTestset,])     #How do we select just certain rows?
#(train <- adverbs[!adverbs$recnum %in% BookIdsInTestset,])


# Hess option may produce something useful, couldn't figure it out.
mymodel <- multinom(out~ little+ without+  other+ nothing+ again  , data = test, hess=TRUE, model=TRUE)
mymodel <- multinom(out~ little+ without+  other+ nothing+ again  , data = test, hess=TRUE)
?multinom
?predict.multinom
?findCorrelation
?predict()
?table()
?str()
?summary()
?pnorm()
?print
?step

?barplot
```