#### Ronak Logistic Regression #####

library(ggplot2)

install.packages("cowplot", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")

library(cowplot)

## NOTE: The data used in this demo comes from the UCI machine learning

## repository.

## http://archive.ics.uci.edu/ml/index.php

## Specifically, this is the heart disease data set.

## http://archive.ics.uci.edu/ml/datasets/Heart+Dise

url <- "http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data"

data <- read.csv(url, header=FALSE)

data

#####################################

##

## Reformat the data so that it is

## 1) Easy to use (add nice column names)

## 2) Interpreted correctly by glm()..

##

#####################################

head(data) # you see data, but no column names

colnames(data) <- c("age","sex",# 0 = female, 1 = male

"cp", # chest pain # 1 = typical angina, # 2 = atypical angina,# 3 = non-anginal pain,# 4 = asymptomatic

"trestbps", # resting blood pressure (in mm Hg)

"chol", # serum cholestoral in mg/dl

"fbs", # fasting blood sugar if less than 120 mg/dl, 1 = TRUE, 0 = FALSE

"restecg", # resting electrocardiographic results # 1=normal#2=ST-T wave abnormality# 3=probable or definite left ventricular hypertrophy

"thalach", # maximum heart rate achieved

"exang", # exercise induced angina, 1 = yes, 0 = no

"oldpeak", # ST depression induced by exercise relative to rest

"slope", # the slope of the peak exercise ST segment# 1 = upsloping# 2 = flat# 3 = downsloping

"ca", # number of major vessels (0-3) colored by fluoroscopy

"thal", #thalium heart scan#3=norma(no cold spots)# 6=fixed defect(coldSpotsDuringRestAndExercise)#7=reversibleDefect(coldSpotsAappearDuringExercise)

"hd" # (the predicted attribute)-diagnosis of heart disease#0 if less than or equal to 50% diameter narrowing#1if greater than 50% diameter narrowing)

head(data)

str(data)

# this shows that we need to tell R which columns contain factors it also shows us that there are some missing values. There are "?"s

## in the dataset. These are in the "ca" and "thal" columns. First, convert "?"s to NAs...

data[data == "?"] <- NA

## Now add factors for variables that are factors and clean up the factors that had missing data...

data[data$sex == 0,]$sex <- "F"

data[data$sex == 1,]$sex <- "M"

data$sex <- as.factor(data$sex)

data$cp <- as.factor(data$cp)

data$fbs <- as.factor(data$fbs)

data$restecg <- as.factor(data$restecg)

data$exang <- as.factor(data$exang)

data$slope <- as.factor(data$slope)

str(data)

# since this column had "?"s in it

# R thinks that the levels for the factor are strings, but

# we know they are integers, so first convert the strings to integers and then into strings

data$ca <- as.integer(data$ca)

data$ca <- as.factor(data$ca)

str(data)

data$thal <- as.integer(data$thal)

data$thal <- as.factor(data$thal)

data$hd <- ifelse(test=data$hd == 0, yes="Healthy", no="Unhealthy") #Another way of adding factors

str(data)

data$hd <- as.factor(data$hd)

str(data)

## Now determine how many rows have "NA" (aka "Missing data"). If it's just

## a few, we can remove them from the dataset, otherwise we should consider

## imputing the values with a Random Forest or some other imputation method.

nrow(data[is.na(data$ca) | is.na(data$thal),])

data[is.na(data$ca) | is.na(data$thal),]

nrow(data)

## NOTE: We also want to exclude variables that only have 1 or 2 samples in

## a category since +/- one or two samples can have a large effect on the

## odds/log(odds)

##

data <- data[!(is.na(data$ca) | is.na(data$thal)),]

nrow(data)

xtabs(~ hd + sex, data=data)

xtabs(~ hd + cp, data=data)

xtabs(~ hd + fbs, data=data)

xtabs(~ hd + restecg, data=data)

xtabs(~ hd + exang, data=data)

xtabs(~ hd + slope, data=data)

xtabs(~ hd + ca, data=data)

xtabs(~ hd + thal, data=data)

## Now we are ready for some logistic regression. First we'll create a very

## simple model that uses sex to predict heart disease

##

xtabs(~ hd + sex, data=data)

## Most of the females are healthy and most of the males are unhealthy.

## Being female is likely to decrease the odds in being unhealthy.

## In other words, if a sample is female, the odds are against it that it

## will be unhealthy

## Being male is likely to increase the odds in being unhealthy...

## In other words, if a sample is male, the odds are for it being unhealthy

logistic\_simple <- glm(hd ~ sex, data=data, family="binomial")

summary(logistic\_simple)

## The intercept is the log(odds) a female will be unhealthy. This is because

## female is the first factor in "sex" (the factors are ordered,

## alphabetically by default,"female", "male")

## Now let's look at the second coefficient...

## sexM 1.2737 0.2725 4.674 2.95e-06 \*\*\*

##

## sexM is the log(odds ratio) that tells us that if a sample has sex=M, the

## odds of being unhealthy are, on a log scale, 1.27 times greater than if

## a sample has sex=F.

female.log.odds <- log(25 / 71)

female.log.odds

# Now you know how these are calculated

male.log.odds.ratio <- log((112 / 89) / (25/71))

male.log.odds.ratio

## Now calculate the overall "Pseudo R-squared" and its p-value

## NOTE: Since we are doing logistic regression...

## Null devaiance = 2\*(0 - LogLikelihood(null model))

## = -2\*LogLikihood(null model)

## Residual deviance = 2\*(0 - LogLikelihood(proposed model))

## = -2\*LogLikelihood(proposed model)

ll.null <- logistic\_simple$null.deviance/-2

ll.proposed <- logistic\_simple$deviance/-2

ll.null

ll.proposed

## McFadden's Pseudo R^2 = [ LL(Null) - LL(Proposed) ] / LL(Null)

(ll.null - ll.proposed) / ll.null

## chi-square value = 2\*(LL(Proposed) - LL(Null))

## p-value = 1 - pchisq(chi-square value, df = 2-1)

1 - pchisq(2\*(ll.proposed - ll.null), df=1)

1 - pchisq((logistic\_simple$null.deviance - logistic$deviance), df=1)

## Lastly, let's see what this logistic regression predicts, given

## that a patient is either female or male (and no other data about them).

predicted.data <- data.frame(probability.of.hd=logistic\_simple$fitted.values,sex=data$sex)

predicated.data

## We can plot the data...

ggplot(data=predicted.data, aes(x=sex, y=probability.of.hd)) +

geom\_point(aes(color=sex), size=5) +

xlab("Sex") +

ylab("Predicted probability of getting heart disease")

## Since there are only two probabilities (one for females and one for males),

## we can use a table to summarize the predicted probabilities.

xtabs(~ probability.of.hd + sex, data=predicted.data)

#####################################

##

## Now we will use all of the data available to predict heart disease. This is not the best way to do this

##

#####################################

logistic <- glm(hd ~ ., data=data, family="binomial")

summary(logistic)

## Now calculate the overall "Pseudo R-squared" and its p-value

ll.null <- logistic$null.deviance/-2

ll.proposed <- logistic$deviance/-2

## McFadden's Pseudo R^2 = [ LL(Null) - LL(Proposed) ] / LL(Null)

(ll.null - ll.proposed) / ll.null

## The p-value for the R^2

1 - pchisq(2\*(ll.proposed - ll.null), df=(length(logistic$coefficients)-1))

## now we can plot the data

predicted.data <- data.frame(probability.of.hd=logistic$fitted.values,hd=data$hd)

predicted.data <- predicted.data[order(predicted.data$probability.of.hd, decreasing=FALSE),]

predicted.data$rank <- 1:nrow(predicted.data)

## Lastly, we can plot the predicted probabilities for each sample having

## heart disease and color by whether or not they actually had heart disease

ggplot(data=predicted.data, aes(x=rank, y=probability.of.hd)) +

geom\_point(aes(color=hd), alpha=1, shape=4, stroke=2) +

xlab("Index") +

ylab("Predicted probability of getting heart disease")

# Few packages for confusion matrix. Lets look at them one by one

install.packages("regclass", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")

library(regclass)

confusion\_matrix(logistic)

install.packages("caret", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")

library(caret)

pdata <- predict(logistic,newdata=data,type="response" )

pdata

data$hd

pdataF <- as.factor(ifelse(test=as.numeric(pdata>0.5) == 0, yes="Healthy", no="Unhealthy"))

install.packages("e1071", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")

library(e1071)

confusionMatrix(pdataF, data$hd)

install.packages("pROC", lib="/Library/Frameworks/R.framework/Versions/3.5/Resources/library")

library(pROC)

roc(data$hd,logistic$fitted.values,plot=TRUE)

par(pty = "s")

roc(data$hd,logistic$fitted.values,plot=TRUE)

## NOTE: By default, roc() uses specificity on the x-axis and the values range

## from 1 to 0. This makes the graph look like what we would expect, but the

## x-axis itself might induce a headache. To use 1-specificity (i.e. the

## False Positive Rate) on the x-axis, set "legacy.axes" to TRUE.

roc(obese, glm.fit$fitted.values, plot=TRUE, legacy.axes=TRUE)

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE)

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage")

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4)

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4)

## If we want to find out the optimal threshold we can store the

## data used to make the ROC graph in a variable...

roc.info <- roc(data$hd, logistic$fitted.values, legacy.axes=TRUE)

str(roc.info)

roc.df <- data.frame(tpp=roc.info$sensitivities\*100, ## tpp = true positive percentage

fpp=(1 - roc.info$specificities)\*100, ## fpp = false positive precentage

thresholds=roc.info$thresholds)

roc.df

head(roc.df) ## head() will show us the values for the upper right-hand corner of the ROC graph, when the threshold is so low

## (negative infinity) that every single sample is called "obese".

## Thus TPP = 100% and FPP = 100%

tail(roc.df) ## tail() will show us the values for the lower left-hand corner

## of the ROC graph, when the threshold is so high (infinity)

## that every single sample is called "not obese".

## Thus, TPP = 0% and FPP = 0%

## now let's look at the thresholds between TPP 60% and 80%

roc.df[roc.df$tpp > 60 & roc.df$tpp < 80,]

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, percent=TRUE)

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, percent=TRUE, print.auc=TRUE)

roc(data$hd,logistic$fitted.values,plot=TRUE, legacy.axes=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, percent=TRUE, print.auc=TRUE, partial.auc=c(100, 90), auc.polygon = TRUE, auc.polygon.col = "#377eb822", print.auc.x=45)

# Lets do two roc plots to understand which model is better

roc(data$hd, logistic\_simple$fitted.values, plot=TRUE, legacy.axes=TRUE, percent=TRUE, xlab="False Positive Percentage", ylab="True Postive Percentage", col="#377eb8", lwd=4, print.auc=TRUE)

# Lets add the other graph

plot.roc(data$hd, logistic$fitted.values, percent=TRUE, col="#4daf4a", lwd=4, print.auc=TRUE, add=TRUE, print.auc.y=40)

legend("bottomright", legend=c("Simple", "Non Simple"), col=c("#377eb8", "#4daf4a"), lwd=4) # Make it user friendly

# reset the par area back to the default setting

par(pty - "m")