Predicting CVD Risk

Ted Laderas, Mark Klick and Shannon McWeeney

8/7/2017

Predicting CVD Risk

We will attempt to predict cardiovascular risk using a patient dataset called `cvd_patient`, which we will load from a Dropbox folder. This dataset is completely synthetic, so don't worry about patient confidentiality.

Before we do anything, let's do a summary on the data.

```
library(tidyverse)
library(broom)
library(caret)
set.seed(111)

cvd_patient <- read.csv("https://www.dropbox.com/s/ve6a3at7h2r9zkg/patient_cvd.csv?raw=1")

summary(cvd_patient)
```

```
##   cvd            age                  race        gender          bmi
##   N:28566   Min.   :41.00   AmInd     :  170   F:21034   Min.   :15.00
##   Y: 7034   1st Qu.:48.00   Asian/PI  : 6389   M:14566   1st Qu.:19.00
##             Median :57.00   Black/AfAm: 2046             Median :21.00
##             Mean   :58.64   White     :26995             Mean   :21.91
##             3rd Qu.:67.00                                3rd Qu.:24.00
##             Max.   :90.00                                Max.   :36.00
##        sbp         htn           tchol       smoking
##   Min.   : 81.0   N:21965   Min.   :155.0   N:31273
##   1st Qu.:117.0   Y:13635   1st Qu.:160.0   Y: 4327
##   Median :128.0             Median :181.0
##   Mean   :142.1             Mean   :188.1
##   3rd Qu.:175.0             3rd Qu.:207.0
##   Max.   :217.0             Max.   :245.0
```

A number of things to note here: Here we can see that of our dataset, 7034 of the total 35600 cases have been diagnosed with cardiovascular disease.

There are a number of covariates we might use to predict whether a patient has cardiovascular disease or not. Please refer to the Data Dictionary for more information about these covariates (note that we are only looking at a limited set of covariates in this dataset, so the number of covariates in the data dictionary is different).

- `age`
- `gender`
- `bmi`
- `sbp`
- `htn`
- `smoking`

Separating Our Data

One of the things we might like to check is the predictive power of the model. For this reason, we want to save a little bit of our data that the model doesn't "see" for testing the predicitve power of the model.

We hold out 20 percent of the data by using the `createPartitionData()` function in `caret`. `createPartitionData()` returns a number of rows that we can use to subset the data into two sets: 1) our `test` dataset (20% of our data), which we'll use to test our model's predictive value, and 2) our `training` dataset (80% of our data), which we'll use to actually build (or train) our model.

```
#grab indices of the dataset that represent 80% of the data
trainingIndices <- createDataPartition(y = cvd_patient$cvd, p=.80,
                                       list=FALSE)

#show the first few training indices
trainingIndices[1:10]
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
#select the rows
trainData <- cvd_patient[trainingIndices,]
#confirm the number of rows (should be 80)
nrow(trainData)
```

```
## [1] 28481
```

```
#build our test set using the R-indexing
#using the "-" operator
testData <- cvd_patient[-trainingIndices,]

#confirm the number of rows
nrow(testData)
```

```
## [1] 7119
```

The Formula Interface for R

One of the most confusing things about R is the formula interface. The thing to remember is that formulas have a certain form. If `Y` is our dependent variable and `X1`, `X2` are independent variables, then the formula to predict `Y` has the format `Y ~ X1 + X2`. Usually these variables come from a `data.frame`, which is supplied by the `data` argument to the function. Note that we don't need quotes to refer to the variables in the data.frame.

Logistic Regression

Here we perform a logistic regression using `cvd` as our dependent variable and our `age` and `gender` as our independent variables. A logistic regression is a type of regression where the *outcome*, or *dependent variable* is related to the probability of categorical variable being true (in our case, whether a patient is considered a cvd risk or not). The output is a model that predicts, for our example, cvd risk.

```
#show variable names in analytic data.frame
colnames(trainData)
```

```
## [1] "cvd"     "age"     "race"    "gender"  "bmi"     "sbp"     "htn"
## [8] "tchol"   "smoking"
```

```
#run a simple logistic regression model just using age and gender
#we can cast gender as categorical data using factor()

ageGenderModel <- glm(cvd ~ age + gender, data= trainData, family="binomial")
summary(ageGenderModel)
```

```
##
## Call:
## glm(formula = cvd ~ age + gender, family = "binomial", data = trainData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5218  -0.6556  -0.4954  -0.3261   2.4785
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.407512   0.083904  -64.45   <2e-16 ***
## age          0.058132   0.001238   46.94   <2e-16 ***
## genderM      0.956444   0.032048   29.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 28314  on 28480  degrees of freedom
## Residual deviance: 25145  on 28478  degrees of freedom
## AIC: 25151
##
## Number of Fisher Scoring iterations: 4
```

```
tidy(ageGenderModel)
```

```
##          term    estimate   std.error statistic       p.value
## 1 (Intercept) -5.40751158 0.083903834 -64.44892  0.000000e+00
## 2         age  0.05813198 0.001238476  46.93834  0.000000e+00
## 3     genderM  0.95644367 0.032047820  29.84427 1.041925e-195
```

Interpreting Logistic Regression Models

Let's look at the output of our model. This gives us the coefficients on the logit scale.

```
#Summarize the model
tidy(ageGenderModel)
```

```
##          term    estimate   std.error statistic       p.value
## 1 (Intercept) -5.40751158 0.083903834 -64.44892  0.000000e+00
## 2         age  0.05813198 0.001238476  46.93834  0.000000e+00
## 3     genderM  0.95644367 0.032047820  29.84427 1.041925e-195
```

```
#grab coefficients themselves
coef(ageGenderModel)
```

```
## (Intercept)         age     genderM
## -5.40751158  0.05813198  0.95644367
```

We note that both of our predictors (`age` and `gender`) are significant terms in our model, which indicates that they are useful predictors in our model. For example, our age p-value is less than 2 x 10^-16, which is highly significant.

How can we use these coefficients? You cannot interpret the Logistic model coefficients strictly in terms of probabilities, because the logistic model is non-linear in terms of probabilities.

However, the coefficients in the model can be interpreted in terms of Odds Ratio. What is the Odds Ratio? Remember that odds can be expressed as `numberCases:numberNonCases`. For example, an odds of 5:1 (win:lose) means that 5 times out of 6, we expect to win, and 1 times out of 6 we expect not to win. The odds ratio (OR) is just `numberCases/numberNonCases`. In the case of 5:1 odds, the OR is 5/1 = 5. The probability of winning in this case would be `numberCases / (numberCases + numberNonCases)` or 5/(1+5) = 0.833333.

So mathematically, the Odds Ratio for our model using our `age` as an independent variable can be calculated as:

$$OddsRatio = \frac{prob(cvd = TRUE)}{prob(cvd = FALSE)} = \frac{numberWithCVD}{numberNoCVD}$$

Note that we use `0` and `1` as shorthand for `TRUE` and `FALSE`.

$$oddsRatio(cvd = 1) = \frac{prob(cvd = 1)}{prob(cvd = 0)} = \frac{prob(cvd = 1)}{1 - prob(cvd = 1)}$$

because

$$prob(cvd = 0) = 1 - prob(cvd = 1)$$

by definition. So, we can define our logistic model as:

$$log(OddRatio(cvd = TRUE)) = Constant + CoefAge * age + CoefGender * gender$$

We call `log(OddsRatio(cvd=TRUE))` the logit. Notice that the logit has a linear relation to our `age` and our `gender`. Our model parameters are a `Constant`, `CoefL` is the fitted model coefficient for our `age`, and `CoefA` is the fitted model coefficient for our `gender`.

if we exponentiate both sides, remembering that `exp(A+B) = exp(A) * exp(B)`:

$$OddsRatio(cvd = TRUE) = exp(Constant + CoefAge * age + CoefGender * gender)$$

Moving things around, we get:

$$OddsRatio(cvd = TRUE) = exp(\frac{prob(cvd = TRUE)}{1 - prob(cvd = TRUE)}) = exp(Constant)*exp(CoefAge*age)*exp(CoefGender*gender$$

So we find that the `OddsRatio(cvd = TRUE)` is calculated by multiplying `exp(Constant)`, `exp(CoefAge * age)` and `exp(CoefGEnder * gender)`, which is a nice result. This means that in order to interpret the coefficients in terms of odds, we need to exponentiate them. We can then interpret these transformed coefficients in terms of an associated increase in the Odds Ratio. So let's first transform our coefficients by exponentiate them.

```r
coefs <- coef(ageGenderModel)
expCoefs  <- exp(coefs)
expCoefs
```

```
## (Intercept)         age     genderM
## 0.004482781 1.059854870 2.602424926
```

Looking at the exponentiated coefficient for `age`, this means that for a 1 unit, or year increase in `age`, the `OddsRatio(cvd score)` is increased by 5.985487 percent. This means that if you want to interpret the coefficients in the model in terms of increases in units, you need to first multiply the unit increase by the coefficient and then exponentiate. For example, going from 1 to 6 in our age (a 5 unit increase), our odds ratio increases by `exp(5 * CoefAge)` or NA.

The interpretation of the `gender` is different because we're treating it as a categorical variable. If the patient is Male, there is a 160.2424926 percent increase in our predicted Odds Ratio, which is very large.

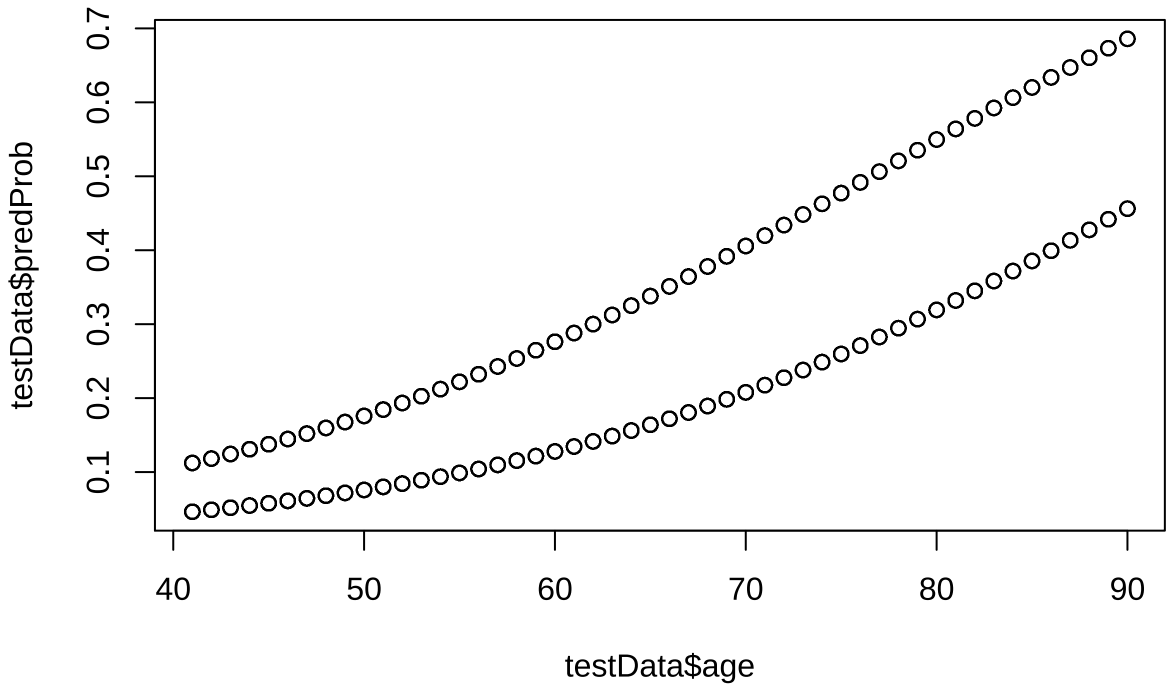
Using models for prediction on our test set

Now you have built a model. What next? If you want to see the values that the model predicts for the dependent variable, you can use `predict()`. This command will return two types of values, based on the arguments we pass it. Either 1) *predicted probabilities* or 2) the *Log(Odds Ratio)*.

If you look at the help entry for `predict.glm` it mentions that by setting the option of `type` to be `response`, you can directly get the predicted probabilities (that is, `prob(cvd=TRUE)`) from the model. We'll use these later when we compare the misclassification rates in our model.

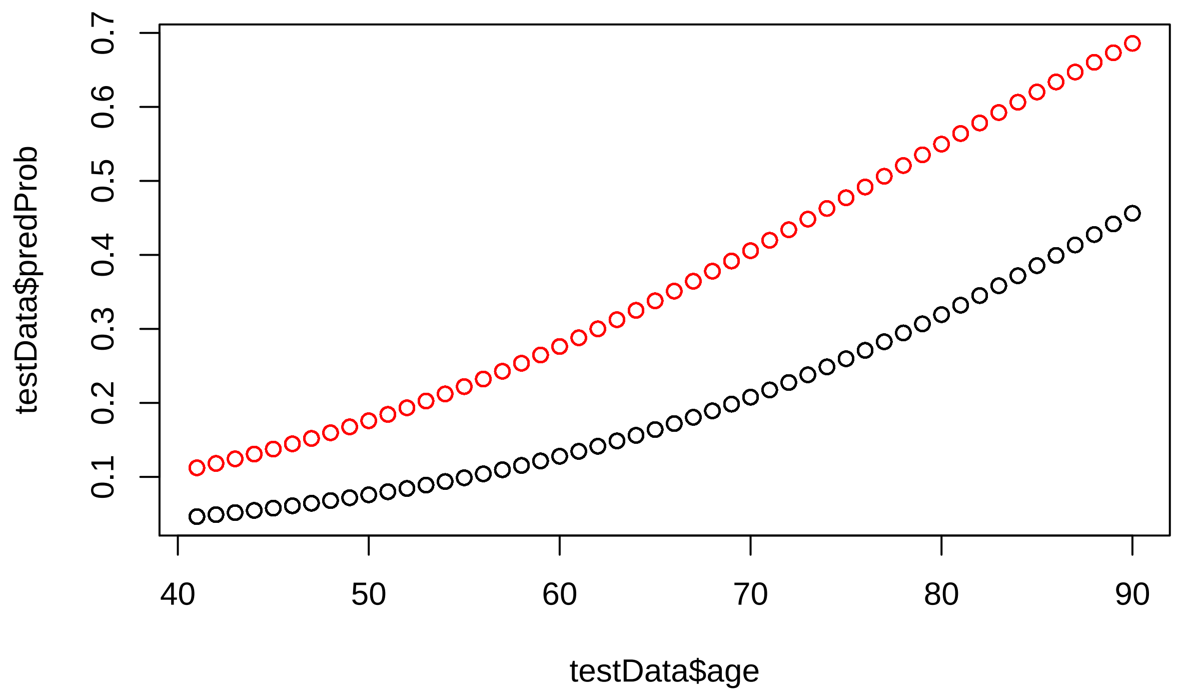
The interpretation of the `gender` is different because we're treating it as categorical data. If the patient is male, there is a 160.2424926 percent increase in our predicted Odds Ratio.

```r
modelPredictedProbabilities <- predict(ageGenderModel, newdata=testData, type = "response")

##add the modelPredictedProbabilities as a column in testData

testData <- data.frame(testData, predProb=modelPredictedProbabilities)

plot(testData$age, testData$predProb)
```



Looking at this plot, we can see two things: our predicted probabilities are in two groups of points. The bottom set of points are the ones for which our `gender` is 0, and the top set of points are where `gender` is 3. This becomes more obvious if we color the points according by `gender`.

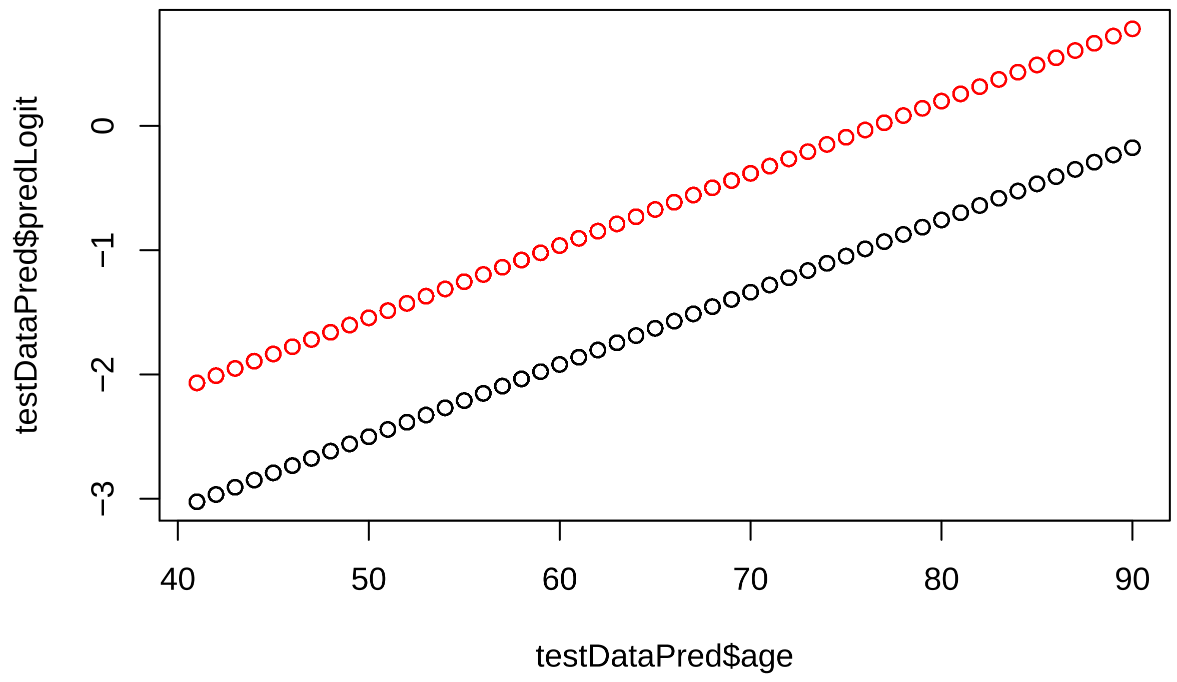
```
plot(testData$age, testData$predProb, col=testData$gender)
```



There are two other things to notice: our predicted probabilities are not that high (the maximum is 0.3) and that the relation between the predicted probabilities and `age` isn't linear.

So, let's visualize the logit instead. If you do not specify the `type` parameter, `predict()` returns the logit, or `log(OddsRatio)`. That means that in order to get the predicted Odds Ratio, you will need to exponentiate the output using `exp()`.

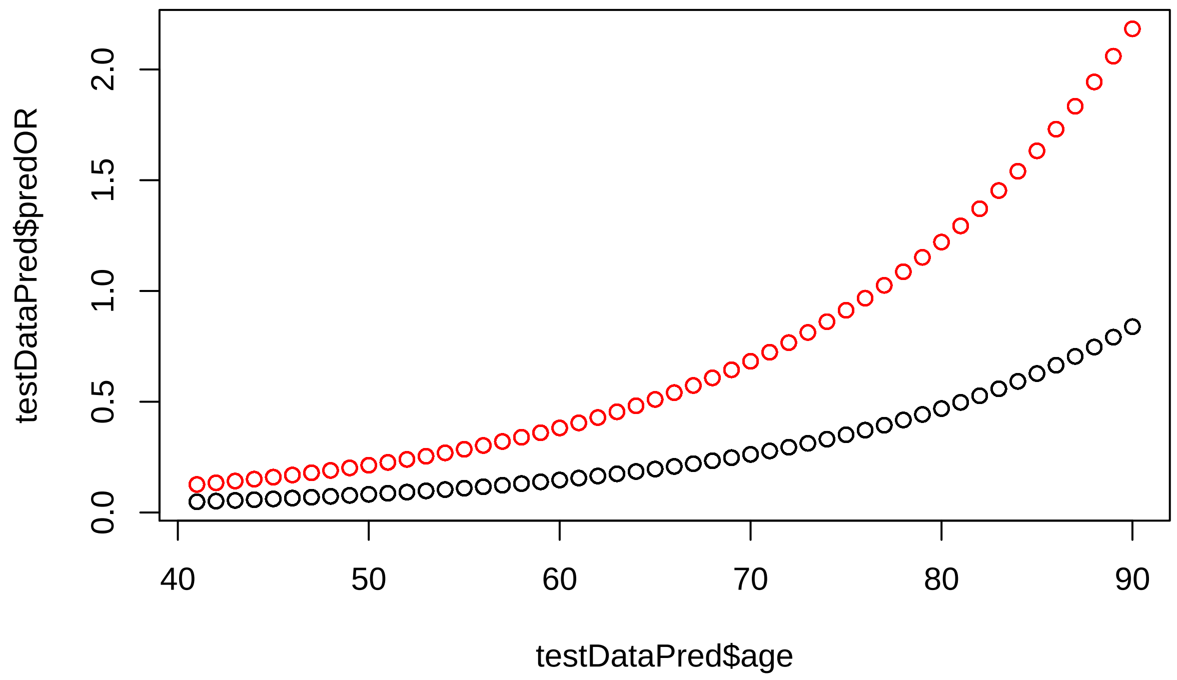
```
#predict the logit instead for our testData
modelPredictedLogOddsRatio <- predict(ageGenderModel,newdata = testData)

#add as another column in our table
testDataPred <- data.frame(testData, predLogit = modelPredictedLogOddsRatio)

#plot the age versus logit (coloring by gender)
plot(testDataPred$age, testDataPred$predLogit, col=testDataPred$gender)
```



```
#transform the logit to the predictedOdds ratio
modelPredictedOddsRatio <- exp(modelPredictedLogOddsRatio)
modelPredictedOddsRatio[1:10]
```

```
##        13        23        30        31        33        38        43
## 0.6076884 0.2397389 0.6826117 0.1340524 0.4816098 0.1595930 0.1264818
##        53        55        58
## 0.1420761 0.5409887 0.1420761
```

```
#add as column in our table
testDataPred <- data.frame(testDataPred, predOR = modelPredictedOddsRatio)

#plot Odds ratio versus age
plot(testDataPred$age, testDataPred$predOR, col=testDataPred$gender)
```



```
#
exp(coef(ageGenderModel))
```

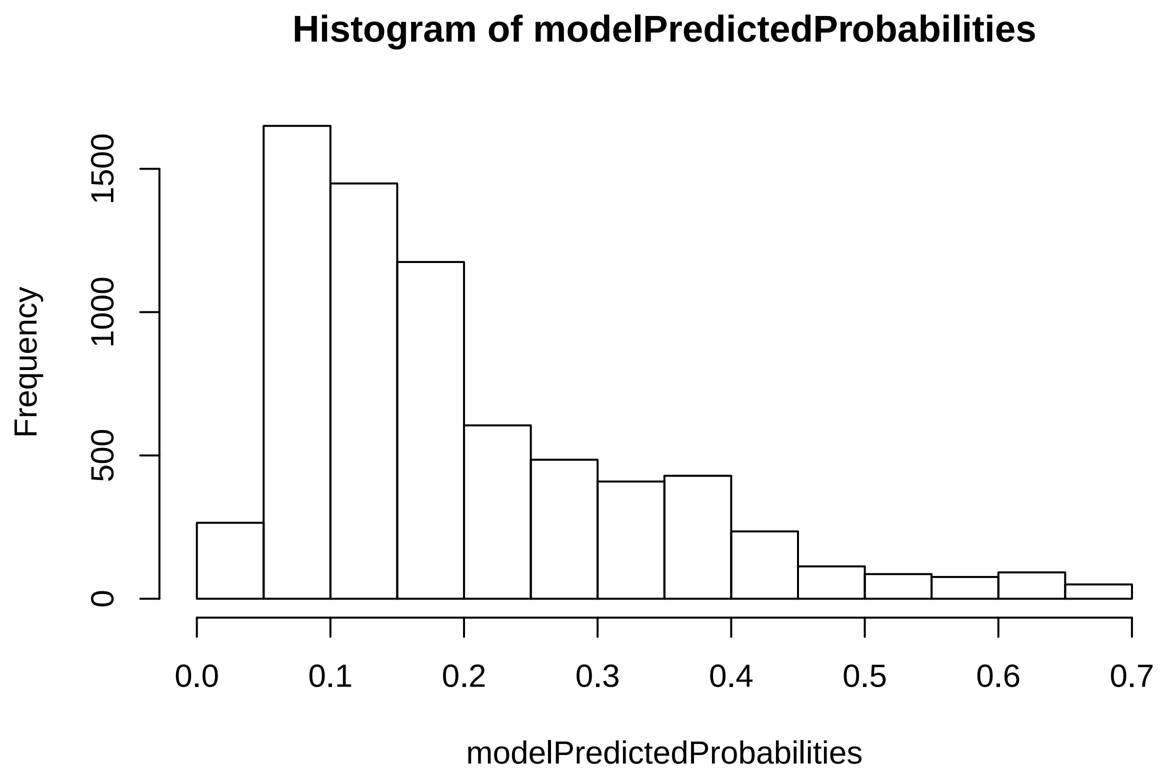
```
## (Intercept)         age     genderM
## 0.004482781 1.059854870 2.602424926
```

Selecting a Probability Threshold

So you might notice that we have predicted probabilities, but we haven't actually predicted any values (whether a patient is at risk for CVD or not). We can do this by choosing a *threshold probability*, that is, a cutoff value for our predicted probabilities that separates who we call as a cvd risk and who isn't.

How do we decide the threshold? One simple way to decide is to do a histogram of the *predicted probabilities*. We note that there is a drop in the predicted probabilities between 0.1 and 0.3.

```
hist(modelPredictedProbabilities)
```



What happens when we set our probability threshold at 0.225? We can use `ifelse()` to recode the probabilities using this threshold.

```
modelPredictions <- ifelse(modelPredictedProbabilities < 0.225, 0, 1)
modelPredictions[1:10]
```

```
## 13 23 30 31 33 38 43 53 55 58
##  1  0  1  0  1  0  0  0  1  0
```

We can do a crosstab between our predictions from our `ageGenderModel` model and the truth (those we have identified as cvd risks) in our `testData`.

```
truthPredict <- table(testData$cvd, modelPredictions)
truthPredict
```

```
##    modelPredictions
##         0    1
##    N 4275 1438
##    Y  613  793
```

Looking at this 2x2 table, you might notice that we do kind of badly in terms of predicting cvd risk. Our accuracy can be calculated by calculating the total number of misclassifications (where predict does not equal truth). The misclassifications are where we predict 1, but the truth is 0 (false positives), and where we predict 0, but the truth is 1 (false negatives).

```
totalCases <- sum(truthPredict)
misclassified <- truthPredict[1,2] + truthPredict[2,1]
misclassified
```

```
## [1] 2051
```

```r
accuracy <- (totalCases - misclassified) / totalCases
accuracy
```

```
## [1] 0.7118977
```

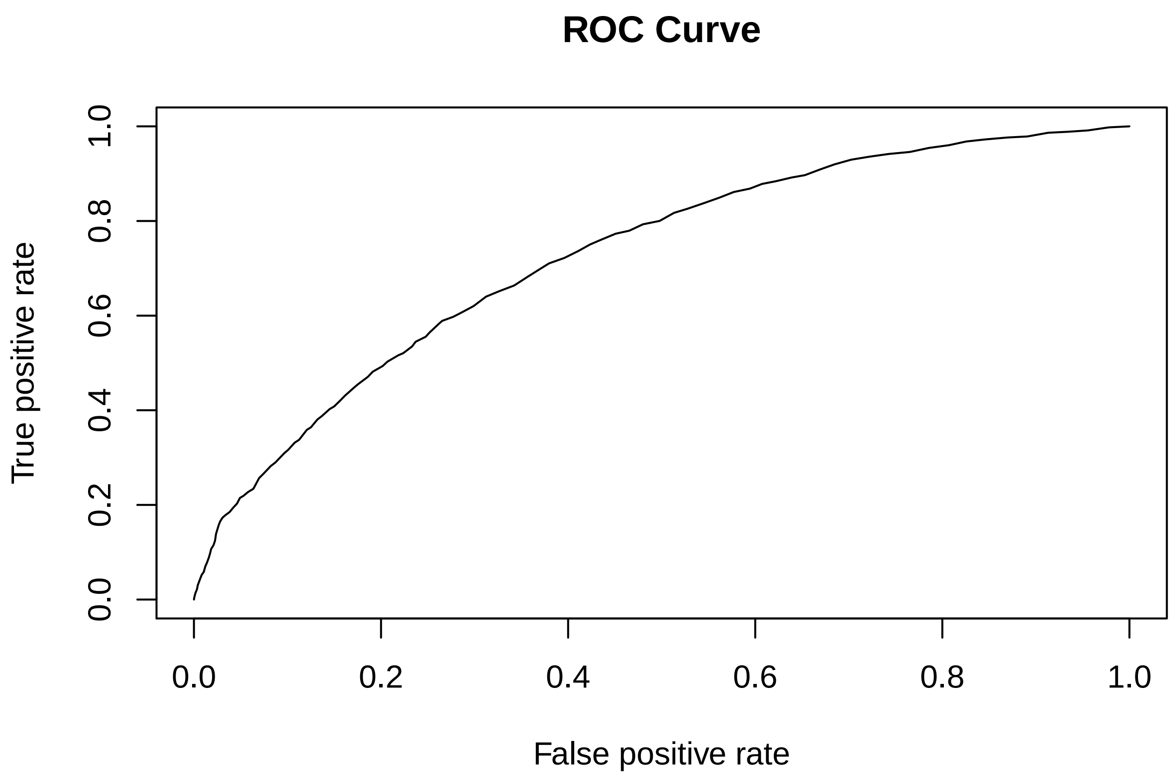
For our `age + gender` model, our prediction threshold of 0.225 has an accuracy of 71.1897738 percent.

ROC Curves

We can examine the impact of setting our probability threshold using the `ROCR` package (be sure to install it using `install.packages("ROCR")`).

An ROC curve (Receiver-Operator-Characteristic) is a way of assessing how our probability threshold affects our Sensitivity (our ability to detect true positives) and Specificity (our ability to detect true negatives). Any test has a sensitivity/specificity tradeoff. We can actually use an ROC curve to select a threshold based on whether we value Sensitivity or Specificity.

```r
library(ROCR)

pr <- prediction(modelPredictedProbabilities, testData$cvd)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf, main="ROC Curve")
```



The area under the ROC curve (AUC) is one way we can summarize our model performance. A model with perfect predictive ability has an AUC of 1. A random test (that is, a coin flip) has an AUC of 0.5.

```r
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.7257228
```

Our `age` + `gender` model has an AUC of 0.7257228, which is not super great. Perhaps you can make it better?

More Info on Logistic Regression

Please note: this is a brief, non-comprehensive introduction to utilizing logistic regression for this class example. In addition to the R links at the end of this section, we highly recommend texts such as *Applied Logistic Regression* (Hosmer, Lemeshow and Sturidvant) for more detailed treatment of logistic regression, including topics such as model building strategies, diagnostics and assessment of fit as well as more complex designs which are beyond the scope of this assignment.

This page https://www.r-bloggers.com/evaluating-logistic-regression-models/ does a nice job explaining how to run logistic regressions and various ways to evaluate logistic regression models.

https://www.r-bloggers.com/how-to-perform-a-logistic-regression-in-r/ is also a good resource for understanding logistic regression models. This page http://www.ats.ucla.edu/stat/mult_pkg/faq/general/odds_ratio.htm is a good page for understanding odds ratios and predicted probabilities.

Homework Week 7

Problem 1 (2 pts)

Modify `ageGenderModel` by changing the covariates. Justify why you think the covariates should be included in the model. Assess the predicted accuracy of your new model compared to the old one. Interpret the effect of one of your numerical covariates (`age`, `bmi`, `sbp` and `tchol`). Is the effect a large one? Which model is a better predictor of CVD? By how much? (use either accuracy or AUC when reporting your result.)

Problem 2 (1 pt)

Should we include all of these covariates: `sbp`, `htn`, and `smoking` in our model? Why or why not? (Think about whether you are providing the same information by including all of these covariates.)

Problem 3 (Optional, 2 pts extra credit)

If you are interested in this, check out the machine learning document to learn more about machine learning apporaches. Compare your model in problem 1 with models built using `lda` (linear discriminant analysis) and `rpart` (classification and regression trees) in terms of accuracy. Use the same covariates you used in Problem 1 to build your models. Do either of these methods do any better than logistic regression on this dataset?