Lec 09: Assessing Model Performance

MATH 456 - Spring 2016

Navbar: [Home] [Schedule] [Data] [Week 12 Overview] [HW Info] [Google Group]

Assigned Reading & additional references

- Afifi Chapter 12.8
- The Wikipedia article on a Confusion Matrix. https://en.wikipedia.org/wiki/Confusion_matrix
  - Terminology that we will be looking at include: sensitivity, specificity, positive predictive value, negative predictive value, false positive rate, false negative rate and accuracy.
  - Notice the connection between Type I and II error and false negatives and false positives.
- The Wikipedia article on Binary Classification https://en.wikipedia.org/wiki/Binary_classification
- An Article discussing accuracy of a blood test for Down syndrome http://www.downsyndromeprenataltesting.com/how-accurate-is-the-new-blood-test-for-down-syndrome/
- An article in Nature regarding the effect of false-positives on newborn screening results on families http://www.nature.com/gim/journal/v14/n1/full/gim20115a.html

Case example: Predicting Depression

```
depress <- read.delim("~/GitHub/MATH456/data/depress_030816.txt")
names(depress) <- tolower(names(depress))
```

Fitting the model

Let's consider a model to predict depression using gender, income and age as predictors.

```
depress$sex <- depress$sex -1
model <- glm(cases ~ sex + income + age, data=depress, family="binomial")
print(xtable(summary(model), digits=3), type='html')
```

Estimate

Std. Error

z value

Pr(>|z|)

(Intercept)

-0.676

0.579

-1.169

0.243

sex

0.929

0.386

2.409

0.016

income

-0.037

0.014

-2.595

0.009

age

-0.021

0.009

-2.318

0.020

Note: The constant displayed here differs from the book

Creating predictions

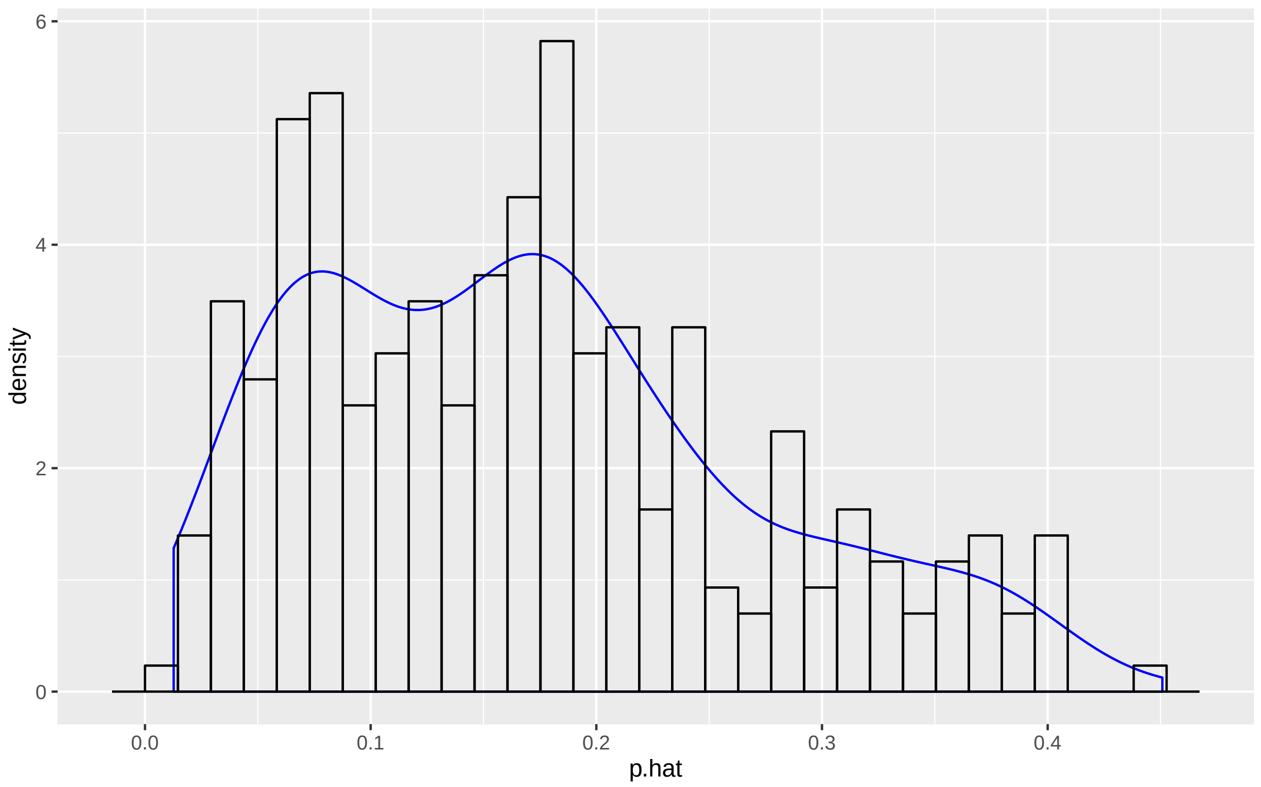
The generic `predict()` function will calculate the predicted probabilities for each record *used in the model* based on the linear predictors for that record $p_i = exp(X\beta)/(1 + exp(X\beta))$. I highlight the *used in the model* because if there is missing data, not all records in the data set will be used to build the model.

So let's make a new data frame that contains the true outcome y, and the predicted probabilities based on the logistic model.

```r
pred_data <- data.frame(y.obs = model$y,
                        p.hat = predict(model, type="response"))
```

What is the distribution of predicted probability of depression?

```r
ggplot(pred_data, aes(x=p.hat)) + geom_density(col="blue") +
  geom_histogram(aes(y=..density..), colour="black", fill=NA)
```



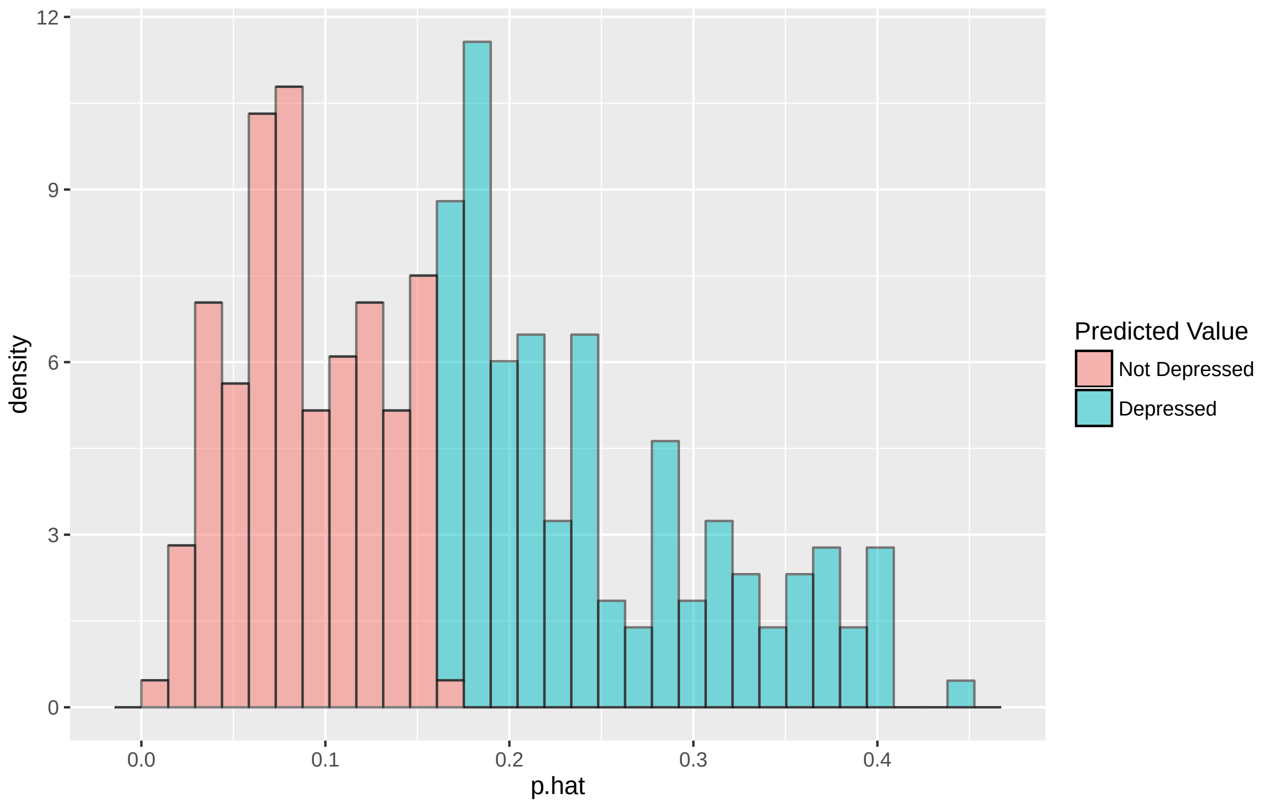
The predicted probability of depression in this sample has an approximately normal distribution that ranges from 0.009 to 0.457, with a mean of 0.170 and median 0.157.

If the investigator wishes to classify cases, a cutoff point on the probability of being depressed, for example, must be found. This cutoff point is denoted (in the book) as P_c. We would classify a person as depressed if the probability of depression is greater than or equal to P_c.

As an example let's choose the median predicted probability as P_c. Records with `p.hat` > 0.157 are classified as being depressed, otherwise they are classified as not depressed.

```r
pred_data$y.pred <- ifelse(pred_data$p.hat >= median(pred_data$p.hat), 1, 0)

ggplot(pred_data, aes(p.hat, fill=factor(y.pred))) +
 # geom_density(color="black") +
  geom_histogram(aes(y=..density..), alpha=.5, color="black") +
  scale_fill_discrete(name="Predicted Value", breaks=c(0,1), labels=c("Not Depressed", "Depressed") )
```



How much trust can we put into such predictions? In other words, can this equation predict correctly a high proportion of the time?

To understand the predictive ability of a model we turn to the **confusion matrix** and measures of model accuracy.

Confusion Matrix.

When the real value of the outcome is known, a confusion matrix can be created by looking at a 2x2 table of the predicted outcome against the true outcome. Be sure to specify what the *positive* outcome is: in this case we are modeling $P(cases = 1)$, so '1' is the event of interest.

```
library(caret)
confusionMatrix(pred_data$y.pred, pred_data$y.obs, positive='1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 132  14
##          1 112  36
##
##                Accuracy : 0.5714
##                  95% CI : (0.5127, 0.6287)
##     No Information Rate : 0.8299
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1467
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7200
##             Specificity : 0.5410
##          Pos Pred Value : 0.2432
##          Neg Pred Value : 0.9041
##              Prevalence : 0.1701
##          Detection Rate : 0.1224
##    Detection Prevalence : 0.5034
##       Balanced Accuracy : 0.6305
##
##        'Positive' Class : 1
##
```

This function is most useful because it provides direct calculations of many of the pieces of information you would need to make a decision on the performance of a model. Some of which are:

- Sensitivity: Probability of correctly predicting the outcome (true positive rate)
- Specificity: Probability of correctly predicting the non-outcome (true negative rate)
- False positive rate: Probability of incorrectly classifying someone as depressed when they really are not.
- Accuracy: Probability of correctly predicting the class (both positive and negative outcomes)

In high dimensional studies (such as genomics), the false discovery rate (probability of a false positive) is a more informative metric than a p-value due to the issues behind multiple comparisons.

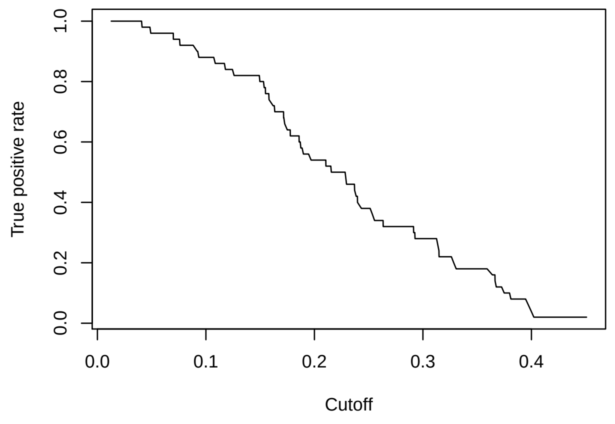
Visualizing model performance.

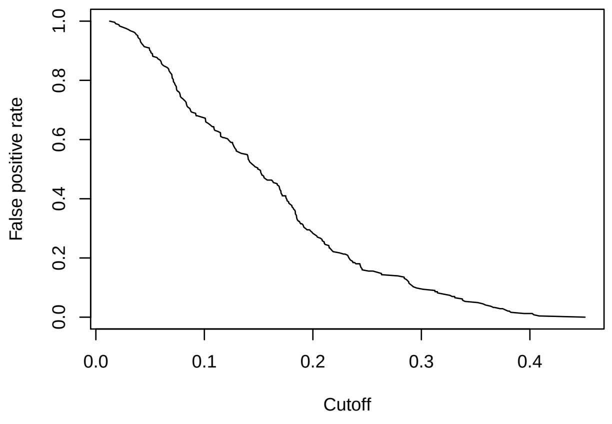
The `ROCR` package provides two useful functions: `prediction` and `performance`. After you load the library you create a `prediction` object by specifying the predicted probability and the true category labels.

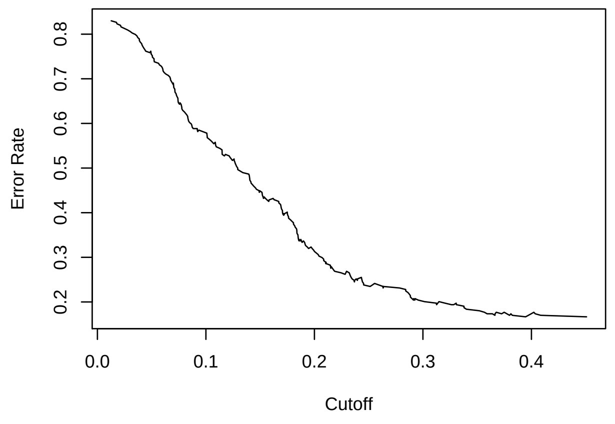
```
library(ROCR)
rocr.pred <- prediction(pred_data$p.hat, pred_data$y.obs)
```

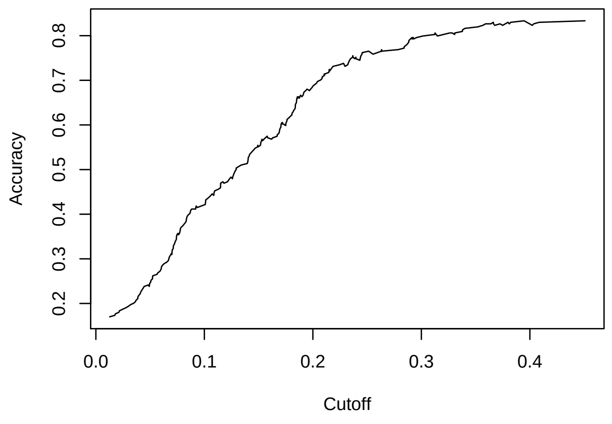
Then we can evaluate the `performance` of the model by creating plots of the measure of interest as a function of the cutoff value. The bottom right plot (accuracy) is similar to the "total" line in Figure 12.5 in Afifi.

```
par(mfrow=c(2,2))
plot(performance(rocr.pred, "tpr"));plot(performance(rocr.pred, "fpr"))
plot(performance(rocr.pred, "err"));plot(performance(rocr.pred, "acc"))
```



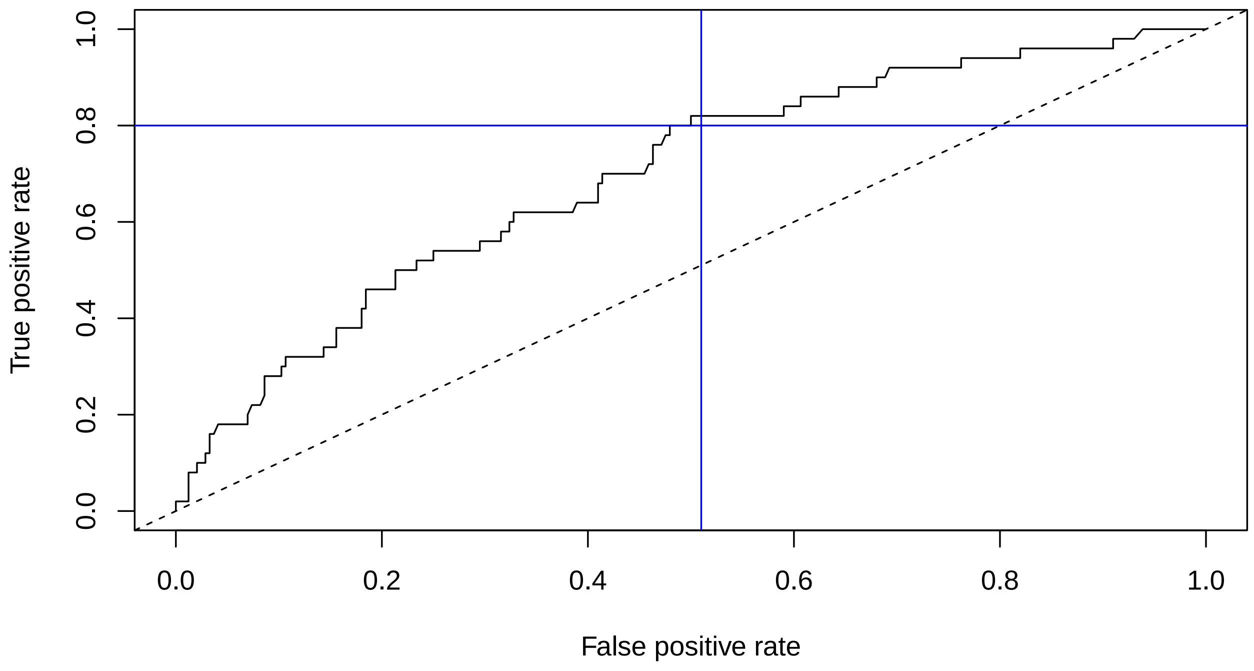






By plotting the true positive rate against the false positive rate you can create the ROC curve. Recall that the greater the area under the ROC curve, i.e. the further away the curved line is from the dashed diagonal line, the better predictive ability a model has (Afifi Figure 12.6).

```r
plot(performance(rocr.pred, "tpr", "fpr"))
lines(abline(h=.8, col=4))
lines(abline(v=.51, col=4))
lines(abline(a=0, b=1, lty="dashed"))
```



The blue lines on the above plot demonstrate that to obtain a high proportion (say 0.80), of depressed persons correctly classified as depressed (true positive rate) we would end up with a false positive rate (classifying non-depressed persons as depressed) of over .5. For most situations this would be considered an unacceptable level of misclassification.

Choosing a cutoff values

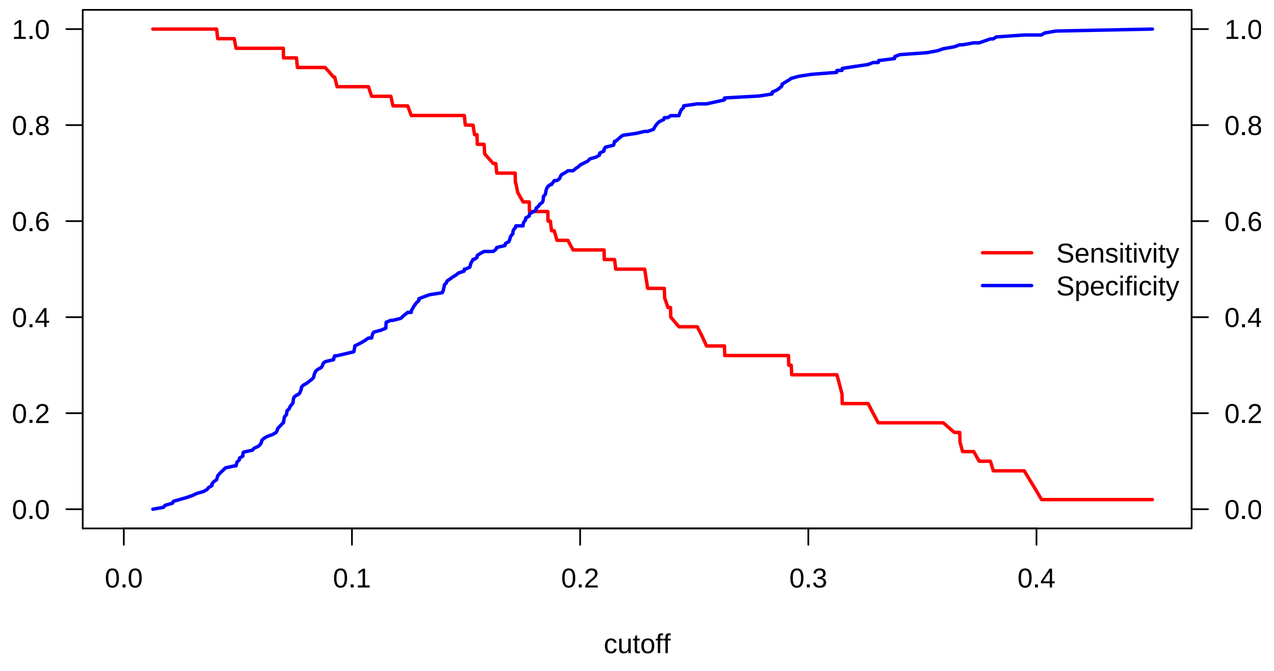
The cutoff value P_c can be thought of as a *tuning* parameter. Tuning parameters are parameters that you can “tweak” to improve your model’s fitness.

One way to choose a cutoff value is to find a balance between sensitivity and specificity.

```
sens <- performance(rocr.pred, "sens")
spec <- performance(rocr.pred, "spec")

x  <- unlist(sens@x.values)
sn <- unlist(sens@y.values)
sp <- unlist(spec@y.values)

plot(c(0, .45), c(0, 1), type="n", axes=FALSE, ylab="", xlab="cutoff")
lines(x, sn, col="red", lwd=2)
lines(x, sp, col="blue", lwd=2)
axis(1); axis(2, las=2)
axis(4, las=2)
box()
legend("right", lwd=2, bty='n', col=c(2,4), legend=c("Sensitivity", "Specificity"))
```



Hosmer Lemeshow goodness of fit test (Afifi pg 298)

From Wikipedia: The Hosmer-Lemeshow test is a statistical test for goodness of fit for logistic regression models. It is used frequently in risk prediction models. The test assesses whether or not the observed event rates match expected event rates in subgroups of the model population. The Hosmer-Lemeshow test specifically identifies subgroups as the deciles of fitted risk values. Models for which expected and observed event rates in subgroups are similar are called well calibrated.

Let's walk through this method once in detail.

1. Calculate p_i for $i = 1, \ldots, n$.

```
# We did this earlier by creating p.hat
head(pred_data)
```

```
##   y.obs      p.hat y.pred
## 1     0 0.21108906      1
## 2     0 0.08014012      0
## 3     0 0.15266203      0
## 4     0 0.24527840      1
## 5     0 0.15208679      0
## 6     0 0.17056409      1
```

2. Divide the probability values into subgroups (usually deciles).

```
# Find the cut points for the groups
deciles <- quantile(pred_data$p.hat, probs=seq(0,1,.1))
deciles
```

```
##         0%        10%        20%        30%        40%        50%
## 0.01271159 0.05219290 0.07429955 0.10111872 0.13025190 0.16303143
##        60%        70%        80%        90%       100%
## 0.18372230 0.21043034 0.24468481 0.31871797 0.45081713
```

```r
# Create categories based on these decile groups
pred_data$group.p <- cut(pred_data$p.hat, deciles)
head(pred_data)
```

```
##   y.obs      p.hat y.pred        group.p
## 1     0 0.21108906      1   (0.21,0.245]
## 2     0 0.08014012      0 (0.0743,0.101]
## 3     0 0.15266203      0   (0.13,0.163]
## 4     0 0.24527840      1  (0.245,0.319]
## 5     0 0.15208679      0   (0.13,0.163]
## 6     0 0.17056409      1  (0.163,0.184]
```

3. For each subgroup calculate the observed and expected number of depressed patients in each decile.

```r
HL <- pred_data %>% group_by(group.p) %>% summarize(O = sum(y.obs), E = sum(p.hat))
```

4. Calculate the Pearson χ^2, and the associated p-value (2 df)

```r
HL.chisq <- sum((HL$O - HL$E)^2 / HL$E)
HL.chisq
```

```
## [1] 2.899538
```

```r
1-pchisq(HL.chisq, 2)
```

```
## [1] 0.2346245
```

Large values of the test statistic indicate poor fit of the model. Equivalently, small p values are an indication of a poor fit. Unfortunately, it does not tell you what is causing the poor fit or how to modify the variables to improve the fit.

If you don't want to do this by hand you can explore the function `hoslem.test` that is found in the `ResourceSelection` package.

- http://thestatsgeek.com/2014/02/16/the-hosmer-lemeshow-goodness-of-fit-test-for-logistic-regression/

False Discovery Rate in Genomics

False discovery rate and Genome wide testing is very important concept in genomic studies. Here are a few articles that Remember: Reading journal articles is hard. It takes time to get good at it, and even then you will struggle.

- An article about false discovery rates in gene wide association studies. http://genomesunzipped.org/2014/11/incorporating-false-discovery-rates-into-genetic-association-in-autism.php * Statistical significance in genetic association studies: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3270946/

- A fully reproducible (yay) article on statistical significance for genomewide studies. This article took another article's data on the genes related to breast cancer and analyzed the effect of p-values and family-wise error rates (Bonferroni, Tukey etc) on high dimensional data. http://www.pnas.org/content/100/16/9440.full
  - All the R code and data files are available here http://genomine.org/qvalue/results.html

[top]

On Your Own

On Your Own

1. Discuss the relationship between sensitivity, specificity, and Type I and II errors.
2. Afifi 12.23
3. For your model chosen in 12.23 create predicted classes based off a median split of the predicted probabilities. Calculate and interpret in context of the problem the following measures: Sensitivity, specificity, accuracy,
   false positive rate, false discovery rate.
4. Consider a different cutoff value (instead of the median). Try to improve the *accuracy* of your model.