IDS 702: Module 2.4

Model assessment and validation - binned residuals and ROC curves

Dr. Olanrewaju Michael Akande

Model assessment and validation

There are various types of residuals when working with generalized linear models (GLMs). For logistic regression in particular, we have

- Response residuals

$$e_i = y_i - \hat{\pi}_i.$$

- Pearson residuals

$$e_i^P = \frac{y_i - \hat{\pi}_i}{\sqrt{\hat{\pi}_i(1-\hat{\pi}_i)}},$$

which are obtained by "normalizing" the response residuals by the estimated Bernoulli standard deviation.

- Deviance residuals

$$e_i^D = \text{sign}(y_i - \hat{\pi}_i) \times 2\left(y_i \log\frac{1}{\hat{\pi}_i} + (1-y_i)\log\frac{1}{1-\hat{\pi}_i}\right),$$

which are the default in R when using the residuals() function. We will talk a bit more about deviance later, but deviance residuals represent the contributions of individual samples to the deviance.

Model assessment and validation

- Deviance residuals are usually the most appropriate for residual plots, when working with GLMs.
- However, unlike what we had for linear regression, just looking at the residuals does not work well here.
  - They are always positive when $Y = 1$ and always negative when $Y = 0$.
  - Also, constant variance is not an assumption of logistic regression.

    Why is that the case?

    Think about the properties of the Bernoulli distribution when we write $y_i|x_i \sim \text{Bernoulli}(\pi_i)$
- We also do not have normality of residuals to work with either.

Model assessment and validation

- What we can do is check to see if the function of predictors is well specified using binned residuals.
- We can assess the overall fit of our model using deviance and change in deviance.
- We can also see how well our model predicts (model validation) using
  - Confusion matrix
  - ROC curves

Binned residuals

- Compute raw (response) residuals for fitted logistic regression.
- Order observations by values of predicted probabilities (or predictor values) from the fitted regression.
- Using ordered data, form g bins of (approximately) equal size. Default: $g = \sqrt{n}$.
- Compute average residual in each bin.
- Plot average residual versus average predicted probability (or average predictor value) for each bin.
- Use the arm package in R.

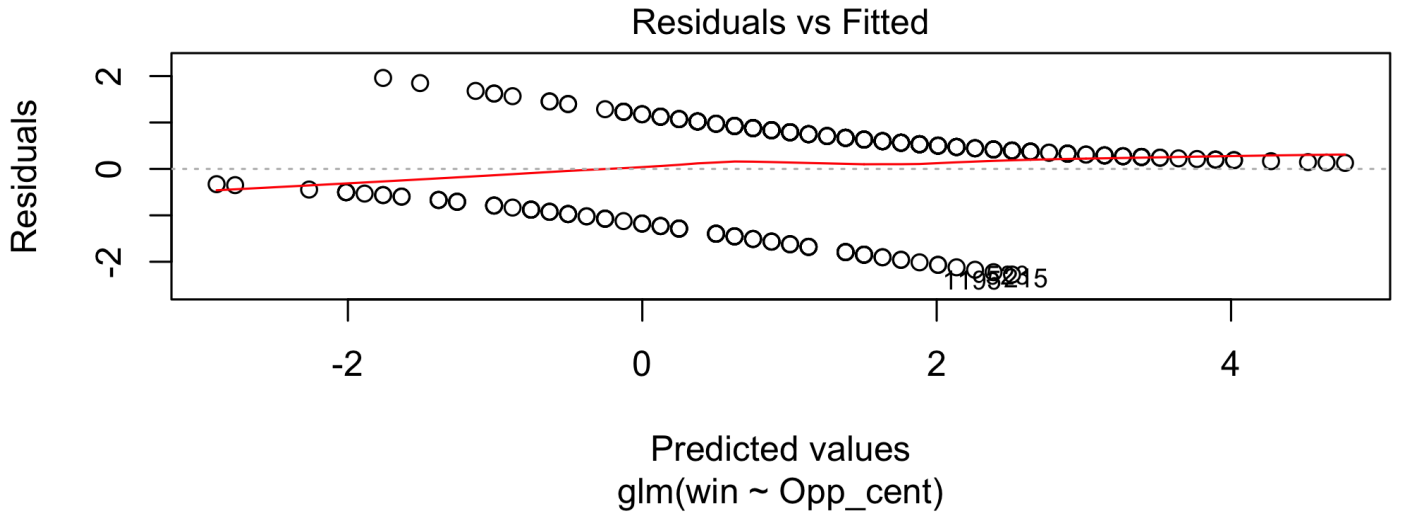
NBA ANALYSIS

Recall the NBA data

```
nba <- read.csv("data/nba_games_stats_reduced.csv",header=T)
nba <- nba[nba$Team=="SAS",]
colnames(nba)[3] <- "Opp"
nba$win <- rep(0,nrow(nba))
nba$win[nba$WINorLOSS=="W"] <- 1
nba$win <- as.factor(nba$win)
nba$Opp_cent <- nba$Opp - mean(nba$Opp)
nbareg <- glm(win~Opp_cent,family=binomial(link=logit),data=nba)
```

NBA ANALYSIS

```
plot(nbareg,which=1)
```



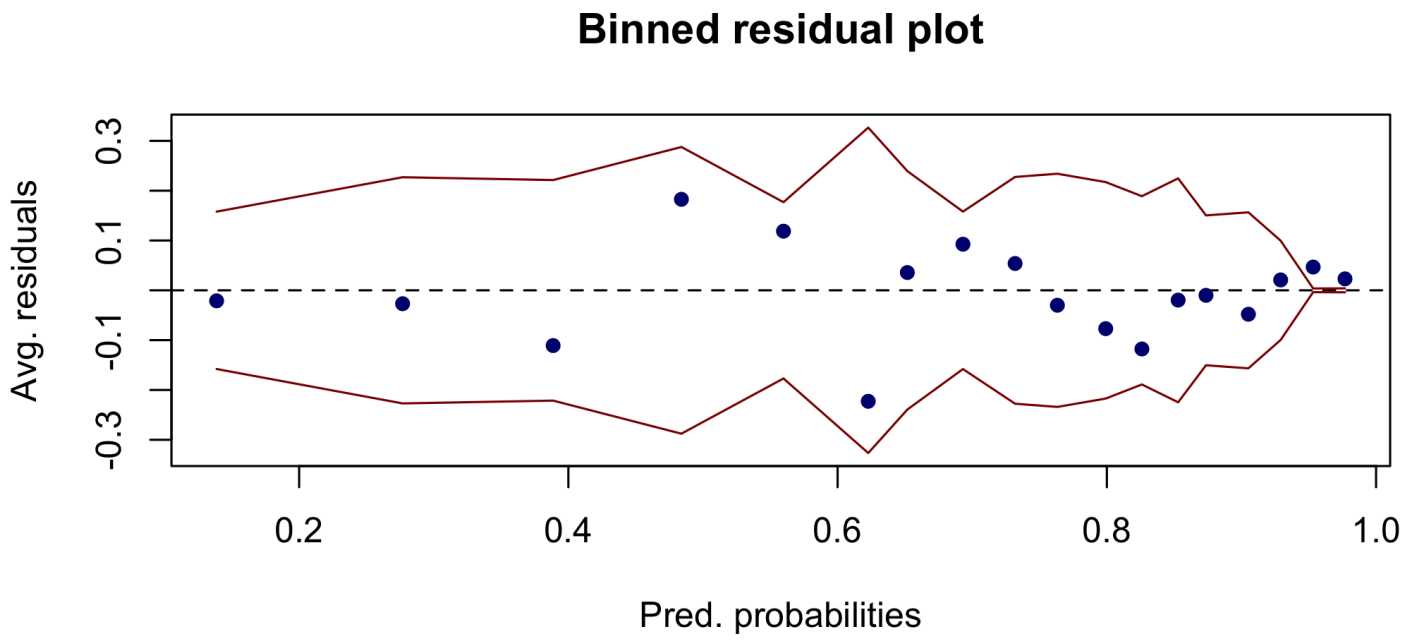
The residuals are the deviance residuals, while the predicted values are on the linear (logit) scale, that is, $\beta_0 + \beta_1 x_i$.

Look to see which cases have large absolute values for cases that don't fit well, but not too useful otherwise.

NBA ANALYSIS

Plot binned raw residuals versus predicted probabilities (arm package).

```
binnedplot(fitted(nbareg),residuals(nbareg,"resp"),xlab="Pred. probabilities",col.int="red
           ylab="Avg. residuals",main="Binned residual plot",col.pts="navy")
```



Look for "randomness" with almost all points within the red lines.

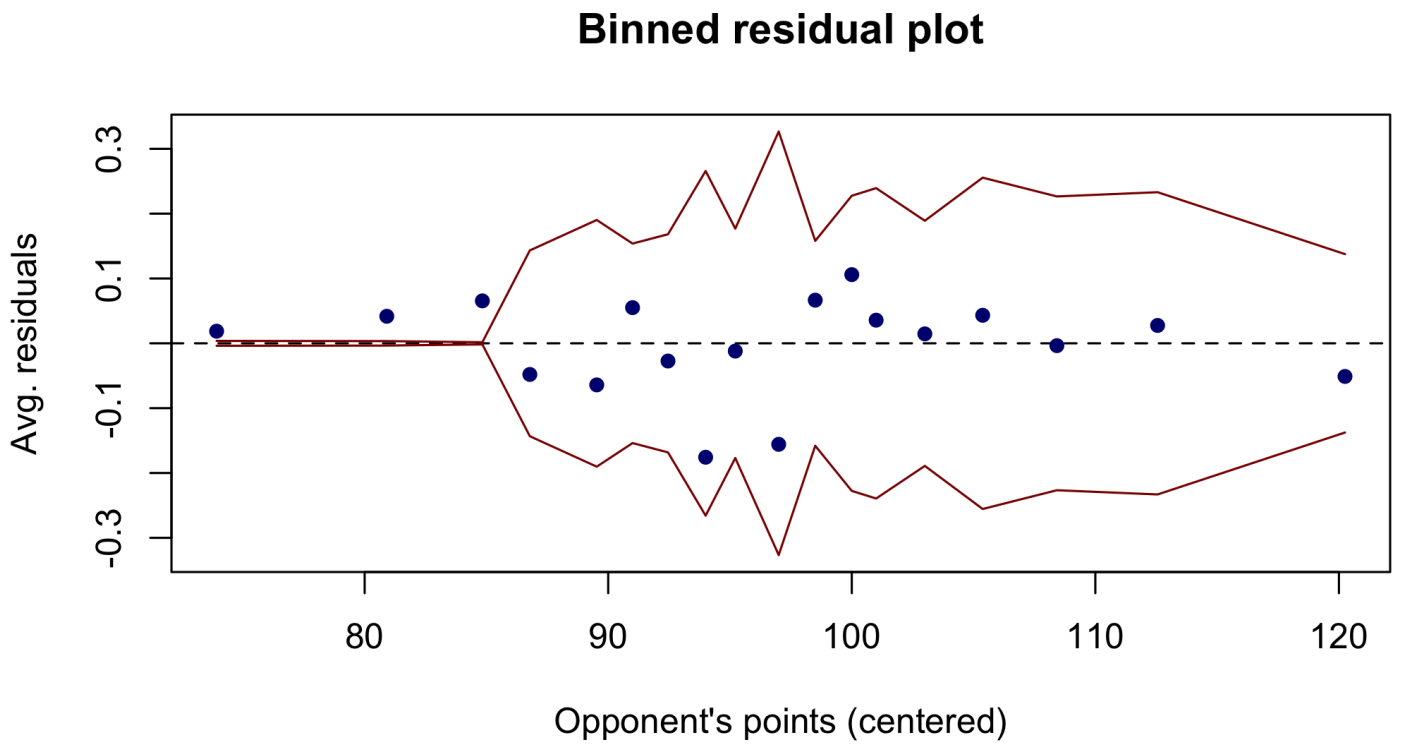
NBA ANALYSIS

- Useful as a "one-stop shopping" plot; especially with many predictors and you want an initial look at model adequacy.
- What we have is mostly good, although model seems to struggle for fitted values over 0.95 or so.
- The red lines represent ± 2 SE bands, which we would expect to contain about 95% of the observations.
- Too few points here to draw any conclusions!
- You usually want many more data points before these plots start being useful.

NBA ANALYSIS

Plot binned raw residuals versus individual predictors.

```
binnedplot(nba$Opp,residuals(nbareg,"resp"),xlab="Opponent's points (centered)",
           col.int="red4",ylab="Avg. residuals",main="Binned residual plot",col.pts="navy'
```



NBA ANALYSIS

- Mostly good, although model seems to struggle for low values of opponent's points.
- Also, too many points (16.7%) outside the bands.
- However, still too few points here for any conclusive takeaways.
- We also know some important predictors are missing by construction...

Deviance

- To assess overall model fit, we can also look at deviance.
- Deviance measures how well the model fits the data, when compared to the saturated model, that is, an abstract model that fits the sample perfectly.
- Precisely, deviance is defined as the difference of likelihoods between the fitted model and the saturated model:

$$D = -2\left[\text{ Log Likelihood(Fitted Model)} - \text{ Log Likelihood(Saturated Model)}\right].$$

- However, this "abstract saturated model" will have likelihood equal to one, so that deviance is simply

$$D = -2\,\text{Log Likelihood(Fitted Model)} = -2\sum_{i=1}^{n}\left[y_i \log(\hat{\pi}_{1i}) + (1-y_i)\log(1-\hat{\pi}_{1i})\right].$$

- Note that deviance is always larger or equal than zero, and will only be zero if the fit is "perfect".
- Overall, deviance is a measure of error, so that, lower values of deviance means better fit to the data.

Deviance

- Like the metrics used under MLR, it is also often useful to use deviance for a model in relation to another model. We will revisit this soon.
- For now, a model we can use for this comparison is the null model, that is, the model with only the intercept.
- Intuitively, this gives us a sense of how much the model improves from the "worst model", by the addition of the predictors.
- The deviance of the null model, denoted D_0, is thus referred to as the null deviance.
- To get a general sense of how much better the fitted model is to the null model, compare D to D_0, usually through the difference $D_0 - D$.
- The "larger" this change in deviance $D_0 - D$ is, the more confident we are that the predictors we have included improve model fit.
- In large samples, $D_0 - D$ has approximately a chi-squared distribution with degrees of freedom equal to the difference in the number of predictors between the two models.

NBA ANALYSIS

For the NBA data for example, we see what looks like a meaningful difference in the two deviance scores.

```
summary(nbareg)
```

```
##
## Call:
## glm(formula = win ~ Opp_cent, family = binomial(link = logit),
##     data = nba)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2760  -0.7073   0.4454   0.7902   1.9593
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.13387    0.15145   7.487 7.06e-14
## Opp_cent    -0.12567    0.01655  -7.594 3.11e-14
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 400.05  on 327  degrees of freedom
## Residual deviance: 313.42  on 326  degrees of freedom
## AIC: 317.42
##
## Number of Fisher Scoring iterations: 5
```

NBA ANALYSIS

- We can formalize this by doing a chi-squared test on the null model vs our fitted model. That is,

```
nbareg_null <- glm(win~1,family=binomial(link=logit),data=nba)
anova(nbareg_null,nbareg,test= "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: win ~ 1
## Model 2: win ~ Opp_cent
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       327     400.05
## 2       326     313.42  1    86.63 < 2.2e-16
```

- The low p-value then confirms our previous statement.
- We will revisit this again when we look at logistic regression with multiple predictors.
- We will be able to use deviance for model comparison and selection by looking at the change in deviance $D_{M_1} - D_{M_2}$, for two models M_1 and M_2, where M_1 is nested within M_2.

Confusion matrix

- We can use the estimated probabilities from our fitted model to predict outcomes, and then compare those to the observed values.
- For example, we could decide to predict $Y = 1$ when the predicted probability exceeds 0.5 and predict $Y = 0$ otherwise.
- We then can determine how many cases we classify correctly and incorrectly.
- Resulting 2×2 table is called the confusion matrix.
- When mis-classification rates are high, model may not be an especially good fit to the data.

Confusion matrix

		Observed	
		Y=1	Y=0
Predicted	Y=1	TP (True Positives)	FP (False Positives)
	Y=0	FN (False Negatives)	TN (True Negatives)

- True positive rate (TPR) = $\frac{TP}{TP + FN}$ (also known as sensitivity)
- False negative rate (FNR) = $\frac{FN}{TP + FN}$
- True negative rate (TNR) = $\frac{TN}{FP + TN}$ (also known as specificity)
- False positive rate (FPR) = $\frac{FP}{FP + TN}$ (1 - specificity)

ROC Curves

- We want high values of sensitivity and low values of (1 - specificity)!
- The receiver operating characteristic (ROC) curve plots
  - Sensitivity on Y axis
  - 1 - specificity on X axis
- Evaluated at lots of different values (beyond 0.5) for the threshold.
- Good fitting logistic regression curves toward the upper left corner, with area under the curve (AUC) near one.
- Make ROC curves in R using the pROC package.
- By the way, we also often define accuracy as $\frac{TP + TN}{TP + FN + FP + TN}$.
  This estimates how well the model predicts correctly overall.

NBA ANALYSIS

Let's look at the confusion matrix for the NBA data. Load the arm, e1071, caret, and pROC packages.

```
Conf_mat <- confusionMatrix(as.factor(ifelse(fitted(nbareg) >= 0.5, "W","L")),
                            nba$WINorLOSS,positive = "W")
Conf_mat$table
```

```
##           Reference
## Prediction   L   W
##          L  44  19
##          W  54 211
```

```
Conf_mat$overall["Accuracy"];
```

```
## Accuracy
## 0.777439
```

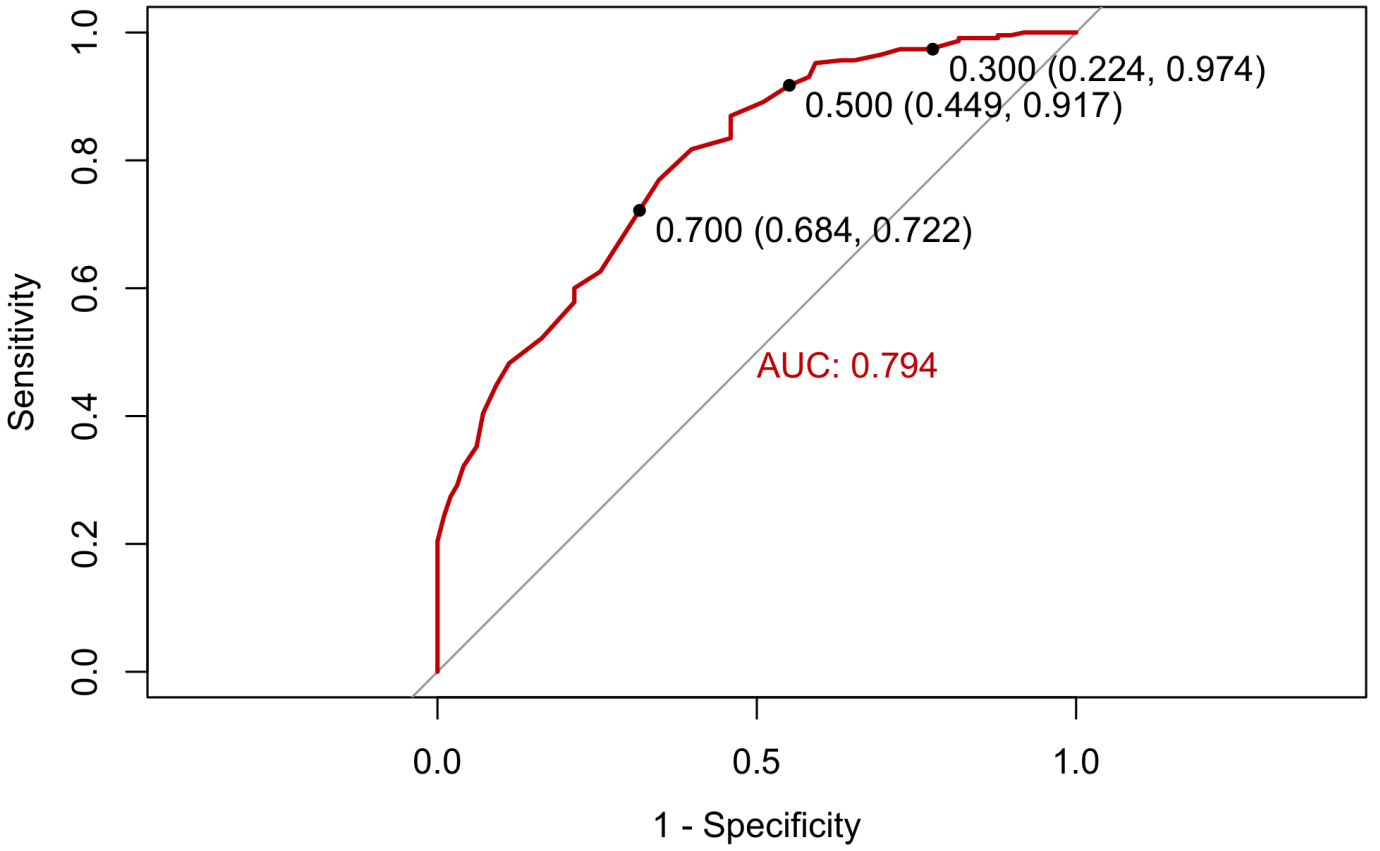
```
Conf_mat$byClass[c("Sensitivity","Specificity")]
```

```
## Sensitivity Specificity
##   0.9173913   0.4489796
```

confusionMatrix produces a lot of output. Print the Conf_mat object to see all of them.

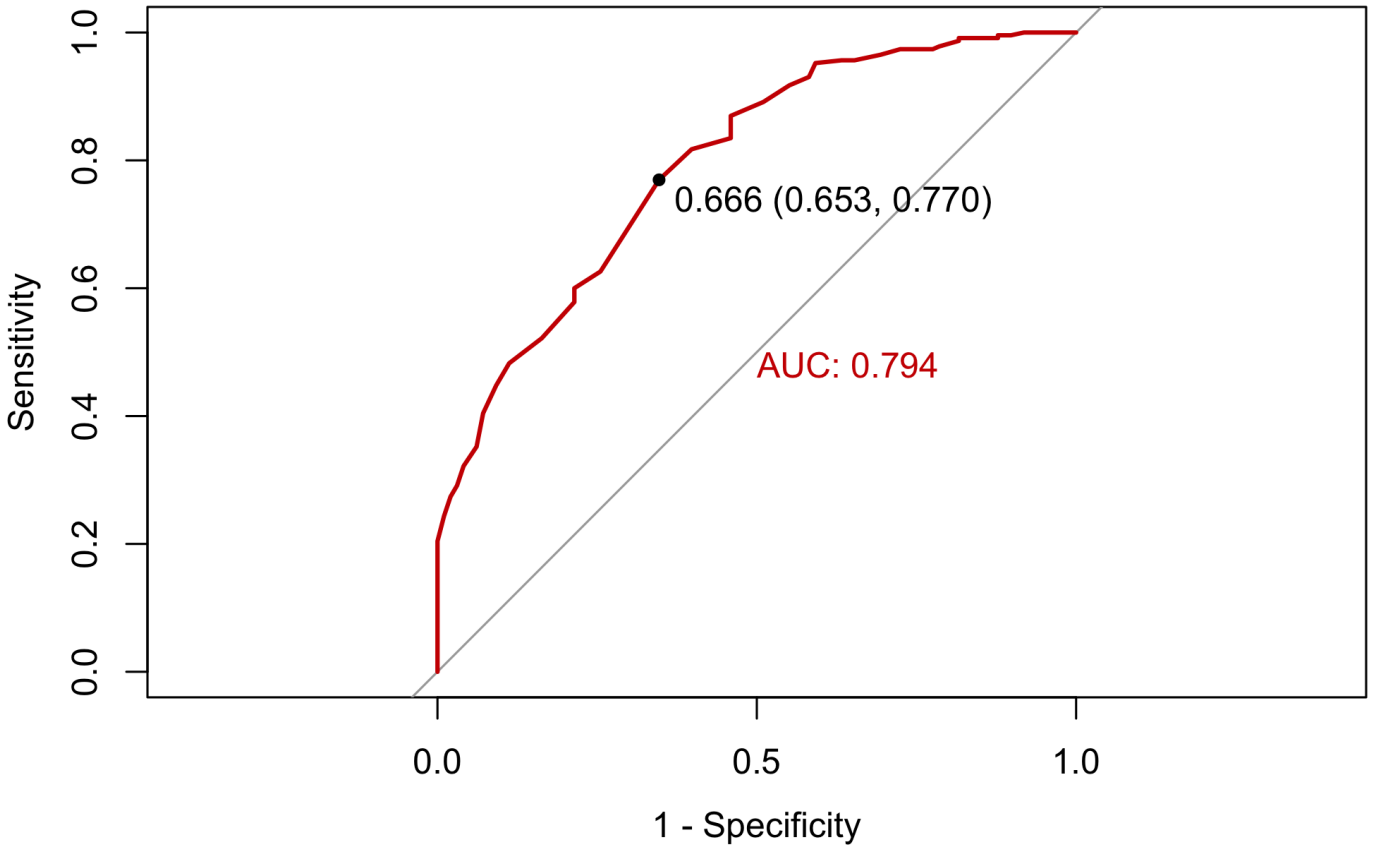
NBA ANALYSIS

```
invisible(roc(nba$win,fitted(nbareg),plot=T,print.thres=c(0.3,0.5,0.7),legacy.axes=T,
              print.auc =T,col="red3"))
```



NBA ANALYSIS

```
invisible(roc(nba$win,fitted(nbareg),plot=T,print.thres="best",legacy.axes=T,
              print.auc =T,col="red3"))
```



What's next?

Move on to the readings for the next module!