

Interpreting logistic models

Frank Edwards

2/22/2019

Review Homework,

Logistic regression part 2

The logit and logistic functions

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \alpha$$

$$\text{logit}^{-1}(\alpha) = \text{logistic}(\alpha) = \frac{\exp(\alpha)}{\exp(\alpha) + 1}$$

Let's define a function in R for logit

```
logit <- function(p) {  
  alpha <- log(p/(1 - p))  
  return(alpha)  
}
```

- What should we expect to see if we run `logit(0.5)`

Let's define a function in R for logit

```
logit <- function(p) {  
  alpha <- log(p/(1 - p))  
  return(alpha)  
}
```

- What should we expect to see if we run `logit(0.5)`
- What about `logit(0.6)`
- `logit(0.9)`?

What does it do?

```
logit(0.5)
```

```
## [1] 0
```

```
logit(0.6)
```

```
## [1] 0.4054651
```

```
logit(0.9)
```

```
## [1] 2.197225
```

Functions in R can run over vectors!

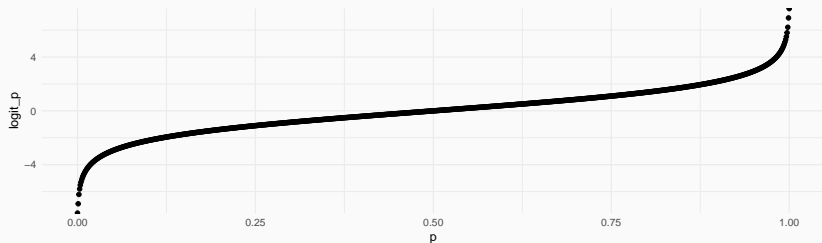
```
p <- c(0.5, 0.6, 0.9)
```

```
logit(p)
```

```
## [1] 0.0000000 0.4054651 2.1972246
```


What does a logit look like with probabilities on [0,1]

```
p <- seq(from = 0, to = 1, by = 0.001)
p_dat <- data.frame(p = p, logit_p = logit(p))
ggplot(p_dat, aes(x = p, y = logit_p)) + geom_point()
```



Defining the inverse logit (logistic) function

Remember:

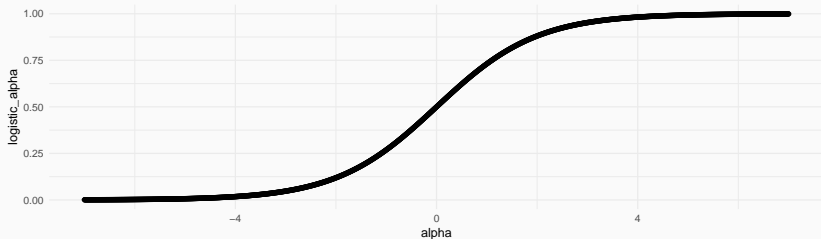
$$\text{logit}^{-1}(\alpha) = \text{logistic}(\alpha) = \frac{\exp(\alpha)}{\exp(\alpha) + 1}$$

```
inv.logit <- function(alpha) {  
  p <- exp(alpha)/(exp(alpha) + 1)  
  return(p)  
}
```

- What does `inv.logit(0)` return?
- What does `inv.logit(10)` return?
- What does `inv.logit(-10)` return?

The shape of the logistic function

```
alpha <- seq(from = -7, to = 7, by = 0.01)
p_dat <- data.frame(alpha = alpha, logistic_alpha = inv.logit(alpha))
ggplot(p_dat, aes(x = alpha, y = logistic_alpha)) + geom_point()
```



Logit and logistic are inverses

```
p <- c(0.1, 0.3, 0.5, 0.7, 0.9)
alpha <- logit(p)
alpha
```

```
## [1] -2.1972246 -0.8472979  0.0000000  0.8472979  2.1972246
```

```
inv.logit(alpha)
```

```
## [1] 0.1 0.3 0.5 0.7 0.9
```

- Remember that logistic regression is a GLM with a logit link function

OK - so why do I need this?

- Remember that logistic regression is a GLM with a logit link function
- A GLM takes the form: $g(y) = \mathbf{X}\beta$

OK - so why do I need this?

- Remember that logistic regression is a GLM with a logit link function
- A GLM takes the form: $g(y) = \mathbf{X}\beta$
- Logistic regression is the special case where g is the logit function:
 $\text{logit}(y) = \mathbf{X}\beta$

OK - so why do I need this?

- Remember that logistic regression is a GLM with a logit link function
- A GLM takes the form: $g(y) = \mathbf{X}\beta$
- Logistic regression is the special case where g is the logit function:
 $\text{logit}(y) = \mathbf{X}\beta$
- A logistic regression model returns $\mathbf{X}\beta$ on the logit scale

OK - so why do I need this?

- Remember that logistic regression is a GLM with a logit link function
- A GLM takes the form: $g(y) = \mathbf{X}\beta$
- Logistic regression is the special case where g is the logit function:
 $\text{logit}(y) = \mathbf{X}\beta$
- A logistic regression model returns $\mathbf{X}\beta$ on the logit scale
- How can we convert $\mathbf{x}\beta$ to something useful?

Let's return to the grad school admission example

```
admits <- read_csv("./data/binary.csv")  
summary(admits)
```

##	admit	gre	gpa	rank
##	Min. :0.0000	Min. :220.0	Min. :2.260	Min. :1.0
##	1st Qu.:0.0000	1st Qu.:520.0	1st Qu.:3.130	1st Qu.:2.0
##	Median :0.0000	Median :580.0	Median :3.395	Median :2.0
##	Mean :0.3175	Mean :587.7	Mean :3.390	Mean :2.4
##	3rd Qu.:1.0000	3rd Qu.:660.0	3rd Qu.:3.670	3rd Qu.:3.0
##	Max. :1.0000	Max. :800.0	Max. :4.000	Max. :4.0

Is rank a numeric?

Is the distance between 1 and 2 symmetric to the distance between 2 and 3?

Is rank a numeric?

Is the distance between 1 and 2 symmetric to the distance between 2 and 3?

Not really. Let's make it a factor.

Is rank a numeric?

Is the distance between 1 and 2 symmetric to the distance between 2 and 3?

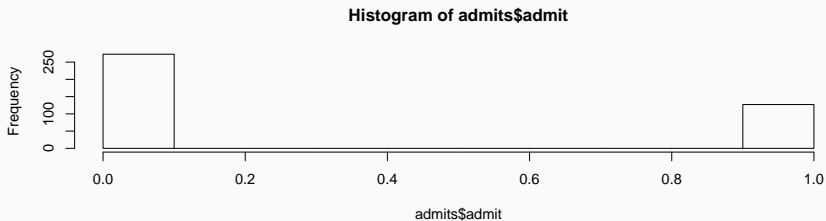
Not really. Let's make it a factor.

```
admits <- admits %>% mutate(rank = factor(rank))  
## as.character() would be fine too
```

Let's explore our outcome

Huh, all this tells us is $\text{mean}(\text{admits}) = 0.3175$

```
hist(admits$admit)
```



Let's look at this as the distribution of the probability of admissions across the data

- First, fit an intercept-only logistic regression model

```
m0 <- glm(admit ~ 1, data = admits, family = "binomial")  
m0_est <- tidy(m0)
```

- What does this model tell us?

What does this model tell us?

```
m0_est$estimate ## log odds
```

```
## [1] -0.7652847
```

```
exp(m0_est$estimate) ## odds
```

```
## [1] 0.4652015
```

```
inv.logit(m0_est$estimate) ## probability
```

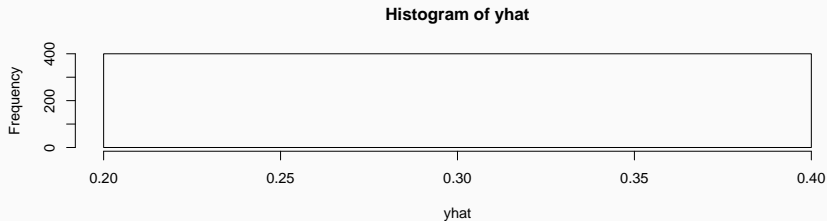
```
## [1] 0.3175
```

```
mean(admits$admit) ## mean admission probability
```

```
## [1] 0.3175
```


Or visually - fascinating!

```
yhat <- predict(m0, type = "response")  
hist(yhat)
```



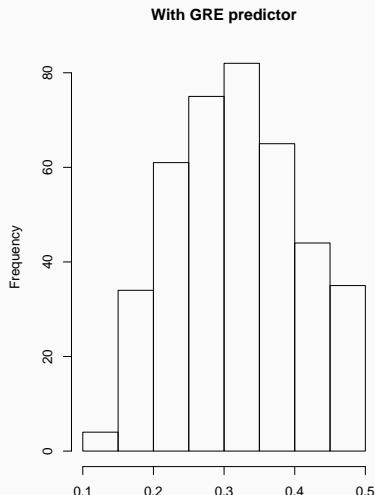
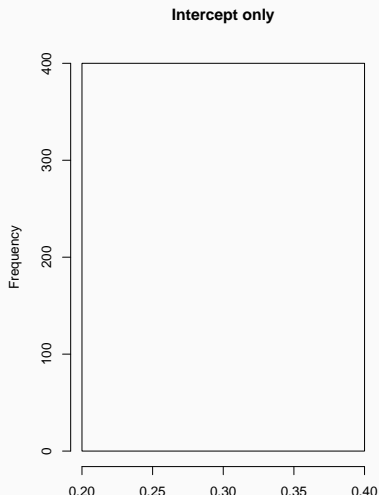
Let's add a predictor

```
m1 <- glm(admit ~ 1 + gre, data = admits, family = "binomial")  
m1_est <- tidy(m1)  
m1_est
```

```
## # A tibble: 2 x 5  
##   term          estimate std.error statistic    p.value  
##   <chr>          <dbl>     <dbl>    <dbl>    <dbl>  
## 1 (Intercept) -2.90      0.606     -4.79 0.00000169  
## 2 gre          0.00358   0.000986     3.63 0.000280
```

Before and after - what's going on?

```
par(mfrow = c(1, 2))  
hist(predict(m0, type = "response"), main = "Intercept only")  
hist(predict(m1, type = "response"), main = "With GRE predictor")
```



To ease interpretation, let's scale gre

Scale mean-centers and SD scales variables: $\text{scale}(x_i) = \frac{x_i - \bar{x}}{sd(x)}$

Re-estimate the model: much nicer to look at

```
admits <- admits %>% mutate(gre = as.numeric(scale(gre)))
m1 <- glm(admit ~ 1 + gre, data = admits, family = "binomial")
m1_est <- tidy(m1)
m1_est
```

```
## # A tibble: 2 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   -0.796     0.111     -7.20 6.01e-13
## 2 gre           0.414     0.114      3.63 2.80e- 4
```

Interpret the model

```
m1_est
```

```
## # A tibble: 2 x 5
```

##	term	estimate	std.error	statistic	p.value
##	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
## 1	(Intercept)	-0.796	0.111	-7.20	6.01e-13
## 2	gre	0.414	0.114	3.63	2.80e- 4

Remember: $\text{logit}(y) = X\beta = \log\left(\frac{y}{1-y}\right)$

So: $y = \text{logit}^{-1}(X\beta) = \frac{\exp(X\beta)}{\exp(X\beta)+1}$

Interpret the model

```
m1_est
```

```
## # A tibble: 2 x 5
```

```
##   term          estimate std.error statistic  p.value  
##   <chr>         <dbl>     <dbl>     <dbl>    <dbl>  
## 1 (Intercept)  -0.796      0.111     -7.20 6.01e-13  
## 2 gre          0.414      0.114      3.63 2.80e- 4
```

Remember: $\text{logit}(y) = X\beta = \log\left(\frac{y}{1-y}\right)$

So: $y = \text{logit}^{-1}(X\beta) = \frac{\exp(X\beta)}{\exp(X\beta)+1}$

- What is β_0 ?

Interpret the model

```
m1_est
```

```
## # A tibble: 2 x 5
```

##	term	estimate	std.error	statistic	p.value
##	<chr>	<dbl>	<dbl>	<dbl>	<dbl>
## 1	(Intercept)	-0.796	0.111	-7.20	6.01e-13
## 2	gre	0.414	0.114	3.63	2.80e- 4

Remember: $\text{logit}(y) = X\beta = \log\left(\frac{y}{1-y}\right)$

So: $y = \text{logit}^{-1}(X\beta) = \frac{\exp(X\beta)}{\exp(X\beta)+1}$

- What is β_0 ?
- What is β_1 ?

$$e^{y_1+y_2} = e^{y_1}e^{y_2}$$

$$e^{y_1+y_2} = e^{y_1}e^{y_2}$$

and

$$e^{y_1-y_2} = \frac{e^{y_1}}{e^{y_2}}$$

Refresher on exponentials

$$e^{y_1+y_2} = e^{y_1}e^{y_2}$$

and

$$e^{y_1-y_2} = \frac{e^{y_1}}{e^{y_2}}$$

so how can we rewrite:

$$\exp(\text{logit}(y)) = \frac{y}{1-y} = e^{\beta_0 + \beta_1 x_1}$$

On the log scale, β_0 and β_1 are related to y multiplicatively because

$$e^{\beta_0 + \beta_1 x_1} = e^{\beta_0} e^{\beta_1 x_1}$$

Odds are defined as the probability of the event occurring divided by the probability of probability of the event not occurring. To obtain odds in a logistic regression, we exponentiate both sides:

$$\frac{y}{1-y} = e^{\beta_0 + \beta_1 x_1}$$

Odds are defined as the probability of the event occurring divided by the probability of probability of the event not occurring. To obtain odds in a logistic regression, we exponentiate both sides:

$$\frac{y}{1-y} = e^{\beta_0 + \beta_1 x_1}$$

The odds of $y = 1$ are simply $e^{x\beta}$

The odds ratio is the ratio of two odds - or the proportional change in odds. We can obtain an isolated estimate for the relationship between $\beta_1 x_{1i}$ and y this way:

$$\frac{\text{Odds}(y|x_1 = 1)}{\text{Odds}(y|x_1 = 0)} = \frac{e^{x\beta + \beta_1}}{e^{x\beta}} = \frac{e^{x\beta} \times e^{\beta_1}}{e^{x\beta}} = e^{\beta_1}$$

The odds ratio can be interpreted as the change in odds of $y == 1$ for a one-unit change in x_1 .

- Odds ratios appear convenient - e^{β_1} is a percent change in y for a one-unit change in x_1

- Odds ratios appear convenient - e^{β_1} is a percent change in y for a one-unit change in x_1

How do they work?

In our example: what do these figures mean?

```
new_dat <- c(1, 0) # for scale(gre) == 0, mean score
odds_0 <- exp(new_dat %*% m1_est$estimate)
odds_0
```

```
##           [,1]
## [1,] 0.4510945
```

```
new_dat1 <- c(1, 1)
odds_1 <- exp(new_dat1 %*% m1_est$estimate)
odds_1
```

```
##           [,1]
## [1,] 0.6823082
```

```
odds_1/odds_0 # odds ratio
```

```
##           [,1]
## [1,] 1.512562
```

```
exp(m1_est$estimate[2]) # exp(beta_1)
```

```
## [1] 1.512562
```

The odds of admission are `exp(m1_est$estimate[2])` times higher for a student with a GRE score one standard deviation above the mean than they are for a student with a mean GRE score.

The odds of admission are `exp(m1_est$estimate[2])` times higher for a student with a GRE score one standard deviation above the mean than they are for a student with a mean GRE score.

Any trouble you can anticipate here?

The odds of admission are `exp(m1_est$estimate[2])` times higher for a student with a GRE score one standard deviation above the mean than they are for a student with a mean GRE score.

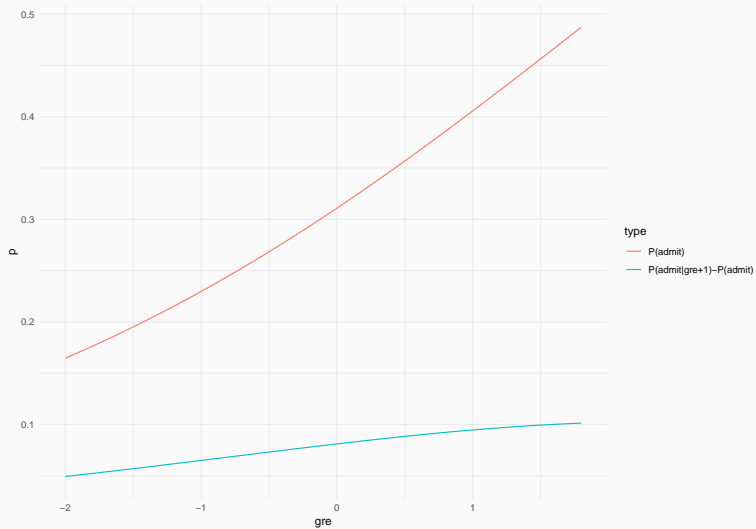
Any trouble you can anticipate here?

Problems with the odds ratio

```
p <- c(0.1, 0.2, 0.5, 0.8, 0.9)
odds <- function(p) {
  p/(1 - p)
}
inv_odds <- function(o) {
  o/(1 + o)
}
odds_p <- odds(p)
OR <- exp(m1_est$estimate[2])
inv_odds(OR + (OR * odds_p)) - p

## [1] 0.52695254 0.45406328 0.25156041 0.08321588 0.03798687
```

A visual example: the “effect” of 1 SD increase in GRE scores on $\Pr(\text{admit}=1)$



It is easy enough to work on the probability scale

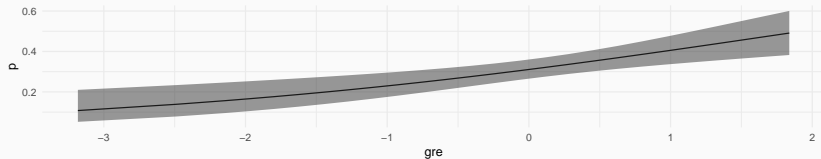
To obtain predicted probabilities of the observed:

- `p_hat<-inv.logit(predict(m1))`
- `p_hat<-predict(m1, type = "response")`

To obtain intervals

```
preds <- predict(m1, se.fit = TRUE)
p_hat <- data.frame(gre = admits$gre, p = inv.logit(preds$fit), upper = inv.logit(
  2 * preds$se.fit), lower = inv.logit(preds$fit - 2 * preds$se.fit))

ggplot(p_hat, aes(x = gre, y = p, ymin = lower, ymax = upper)) + geom_line() +
  geom_ribbon(alpha = 0.5)
```



We can use logistic regression to predict values of a binary variable. We can then assess how well our model performs relative to classifying cases relative to the observations.

We can use logistic regression to predict values of a binary variable. We can then assess how well our model performs relative to classifying cases relative to the observations.

What is the meaning of a:

- a true positive rate (sensitivity)?

We can use logistic regression to predict values of a binary variable. We can then assess how well our model performs relative to classifying cases relative to the observations.

What is the meaning of a:

- a true positive rate (sensitivity)?
- a true negative rate (specificity)?

$$Pr(admit = 1) = \text{logit}^{-1}(\beta_0 + \beta_1 GRE)$$

Returning to our model

$$Pr(admit = 1) = \text{logit}^{-1}(\beta_0 + \beta_1 GRE)$$

What does our model predict for each applicant?

```
phat <- predict(m1, type = "response")  
head(cbind(phat, admits))
```

##	phat	admit	gre	gpa	rank
## 1	0.1765202	0	-1.7980110	3.61	3
## 2	0.3688660	1	0.6258844	3.67	3
## 3	0.4911072	1	1.8378321	4.00	1
## 4	0.3523492	1	0.4527490	3.19	4
## 5	0.2614213	0	-0.5860633	2.93	4
## 6	0.4554030	1	1.4915613	3.00	2

Evaluating our predictions and its performance

```
phat <- predict(m1, type = "response")
summary(phat)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.1078  0.2614  0.3050  0.3175  0.3689  0.4911
```

```
threshold <- quantile(phat, 0.75)
preds <- data.frame(obs = admits$admit == 1, pred = phat > threshold)
table(preds$obs, preds$pred)
```

```
##
##      FALSE TRUE
## FALSE   214   59
## TRUE    87   40
```

```
sum((preds$obs == 1) & (preds$pred == 1))/sum(preds$obs == 1) # True positive rate
```

```
## [1] 0.3149606
```

```
sum((preds$obs == 0) & (preds$pred == 0))/sum(preds$obs == 0) # True negative rate
```

```
## [1] 0.7838828
```

What happens to predictive performance when we add a predictor to the model?

```
m2 <- glm(admit ~ gre + gpa, data = admits, family = "binomial")
phat <- predict(m2, type = "response")
threshold <- quantile(phat, 0.75)
preds <- data.frame(obs = admits$admit == 1, pred = phat > threshold)
sum((preds$obs == 1) & (preds$pred == 1))/sum(preds$obs == 1) # True positive r

## [1] 0.3307087

sum((preds$obs == 0) & (preds$pred == 0))/sum(preds$obs == 0) # True negative r

## [1] 0.7875458
```


What happens to predictive performance when we add a predictor to the model?

```
m3 <- glm(admit ~ gre + gpa + rank, data = admits, family = "binomial")
phat <- predict(m3, type = "response")
threshold <- quantile(phat, 0.75)
preds <- data.frame(obs = admits$admit == 1, pred = phat > threshold)
sum((preds$obs == 1) & (preds$pred == 1))/sum(preds$obs == 1) # True positive r

## [1] 0.4173228

sum((preds$obs == 0) & (preds$pred == 0))/sum(preds$obs == 0) # True negative r

## [1] 0.8278388
```

How can we use this to make decisions? ROC Curves

```
roc_dat <- data.frame(obs = admits$admit, phat = phat)

simple_roc <- function(labels, scores) {
  labels <- labels[order(scores, decreasing = TRUE)]
  data.frame(TPR = cumsum(labels)/sum(labels), FPR = cumsum(!labels)/sum(!labels),
             labels, probs = scores[order(scores, decreasing = TRUE)])
}

roc_out <- simple_roc(roc_dat$obs, roc_dat$phat)
1/sum(roc_dat$obs)

## [1] 0.007874016

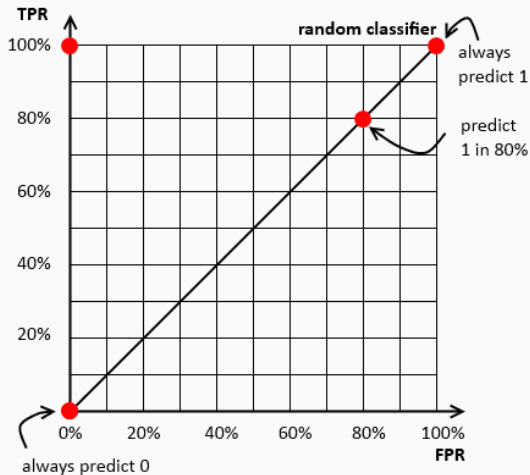
1/sum(!(roc_dat$obs))

## [1] 0.003663004

slope <- sum(!(admits$admit))/sum(admits$admit)
head(roc_out)

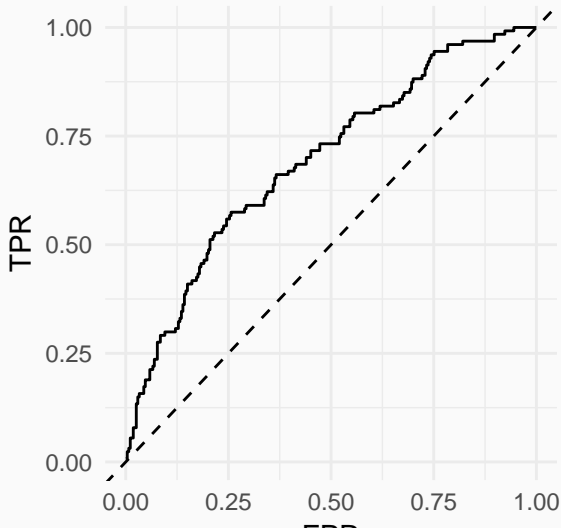
##           TPR           FPR labels      probs
## 1 0.007874016 0.000000000      1 0.7384082
## 2 0.007874016 0.003663004      0 0.7337223
## 3 0.015748031 0.003663004      1 0.7205386
## 4 0.023622047 0.003663004      1 0.6960719
## 5 0.023622047 0.007326007      0 0.6943683
## 6 0.031496063 0.007326007      1 0.6923799
```

Plotting in ROC space



Plotting the ROC curve

```
ggplot(roc_out, aes(x = FPR, y = TPR)) + geom_line() + geom_abline(
  lty = 2)
```



Comparing models

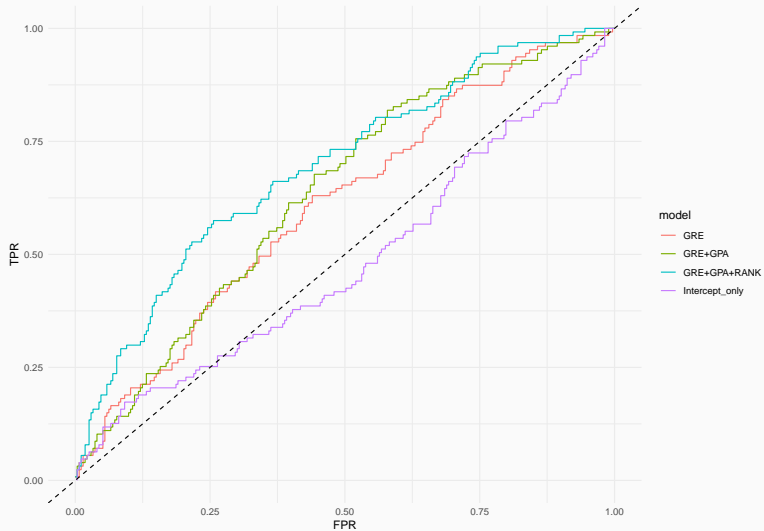
```
models <- list(m0, m1, m2, m3)
preds <- lapply(models, function(x) {
  predict(x, type = "response")
})
str(preds)
```

```
## List of 4
## $ : Named num [1:400] 0.318 0.318 0.318 0.318 0.318 ...
## ..- attr(*, "names")= chr [1:400] "1" "2" "3" "4" ...
## $ : Named num [1:400] 0.177 0.369 0.491 0.352 0.261 ...
## ..- attr(*, "names")= chr [1:400] "1" "2" "3" "4" ...
## $ : Named num [1:400] 0.231 0.4 0.555 0.306 0.208 ...
## ..- attr(*, "names")= chr [1:400] "1" "2" "3" "4" ...
## $ : Named num [1:400] 0.173 0.292 0.738 0.178 0.118 ...
## ..- attr(*, "names")= chr [1:400] "1" "2" "3" "4" ...
```

Comparing models (cont.)

```
roc_temp <- list()
for (i in 1:length(preds)) {
  roc_temp[[i]] <- simple_roc(labels = admits$admit, scores =
}
roc_dat <- bind_rows(roc_temp)
roc_dat <- roc_dat %>% mutate(model = rep(c("Intercept_only", "G
  "GRE+GPA+RANK"), each = nrow(admits)))
```

Plotting this monstrosity

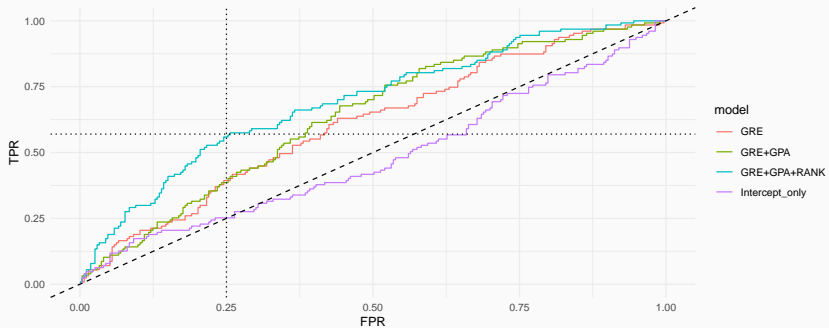


Selecting a threshold

We can use many thresholds:

- We can optimize overall accuracy - this will be the inflection point in the curve

Selecting a threshold



```
##   threshold      FPR      TPR
## 1 0.3589025 0.2490842 0.5590551
```

Selecting a threshold

We can use many thresholds:

- We can optimize overall accuracy - this will be the inflection point in the curve
- We can set arbitrary thresholds (i.e. 10 percent false positive)

```
roc_dat %>% filter(model == "GRE+GPA+RANK") %>% filter(FPR < 0.1)
      FPR = max(FPR), TPR = max(TPR))
```

```
##   threshold      FPR      TPR
## 1 0.4675591 0.0989011 0.2992126
```

- This model was fit to our observed data

- This model was fit to our observed data
- Then predicted itself...

- This model was fit to our observed data
- Then predicted itself...
- Can it predict new cases?

- We could use this model to predict next year's admits to target high and low likelihood cases

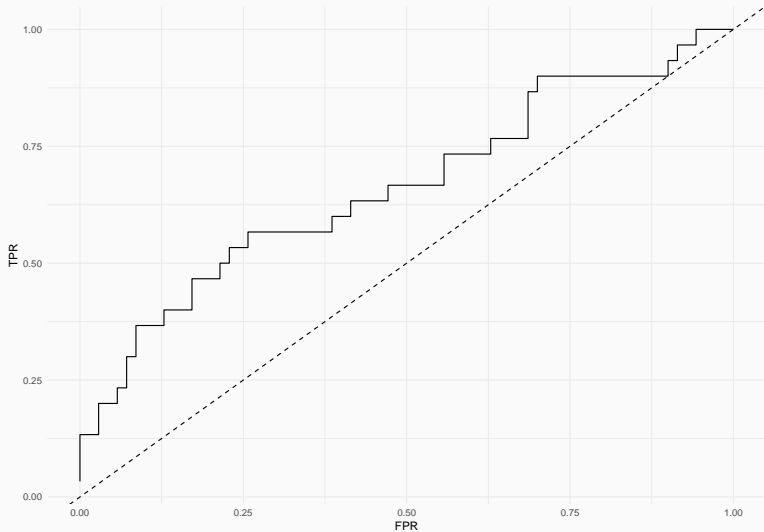
- We could use this model to predict next year's admits to target high and low likelihood cases
- But we only have one year of data. We can split our data into two sets: one for fitting the model (training) and one for evaluating the model fit (test data)

Example with the admissions data

Let's fit the data to a 75 percent subset, then test it against the remaining 25 percent

```
sample_size <- trunc(nrow(admits) * 0.25)
test_rows <- sample(1:nrow(admits), sample_size, replace = FALSE)
training_dat <- admits[-test_rows, ]
test_dat <- admits[test_rows, ]
model_validation <- glm(admit ~ gre + gpa + rank, data = admits, family = "binomial")
yhat <- predict(model_validation, newdata = test_dat, type = "response")
```


Now check out the ROC curve



Out of sample validation is crucial for prediction

- Using the same data to train and test a model leads to overfitting
- New data is necessary to effectively check the performance of your model
- Especially in real world settings where mistakes can be dangerous/costly