Old Assignment 1 Answer Sketch

Thomas E. Love

2018-01-06

Contents

1 Looking Over the Data Set

I will start with a quick summary to be sure things are imported properly in the `hw1` data set…

```
summary(hw1)
```

```
    subject             treatment        cov1            cov2      
 Min.   :101.0   Not Treated:95   Min.   :27.03   Min.   :29.49  
 1st Qu.:134.5   Treated    :40   1st Qu.:41.74   1st Qu.:46.48  
 Median :228.0                    Median :47.70   Median :53.48  
 Mean   :210.2                    Mean   :48.36   Mean   :52.68  
 3rd Qu.:261.5                    3rd Qu.:56.62   3rd Qu.:59.99  
 Max.   :295.0                    Max.   :73.28   Max.   :73.25  
      cov3            cov4           female      
 Min.   : 8.00   Min.   : 9.00   Min.   :0.0000  
 1st Qu.:17.00   1st Qu.:17.00   1st Qu.:0.0000  
 Median :20.00   Median :20.00   Median :0.0000  
 Mean   :20.28   Mean   :20.06   Mean   :0.4296  
 3rd Qu.:24.00   3rd Qu.:23.00   3rd Qu.:1.0000  
 Max.   :33.00   Max.   :33.00   Max.   :1.0000  
```

1.1 Using the describe function

Alternatively, we could use the `describe` function, which is part of the `Hmisc` library…

```r
library(Hmisc)
describe(hw1)
```

```
hw1 

 7  Variables      135  Observations
---------------------------------------------------------------------------
subject 
       n  missing distinct     Info     Mean      Gmd      .05      .10 
     135        0      135        1    210.2    70.54    107.7    114.4 
     .25      .50      .75      .90      .95 
   134.5    228.0    261.5    281.6    288.3 

lowest : 101 102 103 104 105, highest: 291 292 293 294 295
---------------------------------------------------------------------------
treatment 
       n  missing distinct 
     135        0        2 

Value      Not Treated     Treated
Frequency           95          40
Proportion       0.704       0.296
---------------------------------------------------------------------------
cov1 
       n  missing distinct     Info     Mean      Gmd      .05      .10 
     135        0      133        1    48.36    11.83    30.70    34.36 
     .25      .50      .75      .90      .95 
   41.75    47.70    56.62    61.84    64.94 
```

```
lowest : 27.03 28.30 28.39 28.73 29.60, highest: 66.37 67.30 67.89 68.39 73.28
---------------------------------------------------------------------------
cov2
       n  missing distinct     Info     Mean      Gmd      .05      .10
     135        0      133        1    52.68    11.08    36.81    39.15
     .25      .50      .75      .90      .95
   46.48    53.48    59.99    65.39    67.05

lowest : 29.49 31.93 32.24 34.24 34.25, highest: 68.90 69.55 70.10 72.03 73.25
---------------------------------------------------------------------------
cov3
       n  missing distinct     Info     Mean      Gmd      .05      .10
     135        0       24    0.995    20.28     5.87       12       14
     .25      .50      .75      .90      .95
      17       20       24       27       29

lowest :  8  9 11 12 13, highest: 28 29 30 32 33
---------------------------------------------------------------------------
cov4
       n  missing distinct     Info     Mean      Gmd      .05      .10
     135        0       23    0.994    20.06    4.929       13       15
     .25      .50      .75      .90      .95
      17       20       23       26       28

lowest :  9 11 12 13 14, highest: 28 29 31 32 33
---------------------------------------------------------------------------
female
       n  missing distinct     Info      Sum     Mean      Gmd
     135        0        2    0.735       58   0.4296   0.4938

---------------------------------------------------------------------------
```

1.2 Glimpsing the data's structure

Or, perhaps we just want to see the structure of the data and some of the first few values in each variable, in which case, the `str` command would help, or we could use the `dplyr` library's `glimpse` command…

```r
library(dplyr)
glimpse(hw1)
```

```
Observations: 135
Variables: 7
$ subject   <int> 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 11...
$ treatment <fctr> Treated, Treated, Treated, Treated, Treated, Treate...
$ cov1      <dbl> 38.38, 39.07, 67.30, 61.54, 66.37, 57.08, 50.28, 61....
$ cov2      <dbl> 53.70, 48.53, 53.86, 52.18, 55.60, 44.45, 66.83, 68....
$ cov3      <int> 13, 15, 11, 18, 19, 24, 15, 15, 19, 16, 15, 15, 20, ...
$ cov4      <int> 19, 21, 16, 21, 22, 33, 16, 26, 18, 24, 18, 23, 24, ...
$ female    <int> 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0...
```

The two `treatment` options are named "Treated" and "Not Treated", as opposed to "Treated" and "Untreated". Anything so that the thing I wanted to evaluate probabilities for (i.e. Treated as compared to Not Treated) came second alphabetically is appealing, because R, by default, treats the first level in a binary categorical variable as unsuccessful and the second level as successful and generally orders levels of binary variables alphabetically.

2 Fitting a Logistic Regression Model using the `glm` function

```
m1 <- glm(treatment == "Treated" ~
            cov1 + cov2 + cov3 + cov4 + female,
          family=binomial(), data=hw1)
```

We are fitting a model to predict the probability of "Treated" here. If we want to see what's in m1, we can type it in, and see what we get, or ask for a summary, and get some additional details.

```
m1
```

```

Call:  glm(formula = treatment == "Treated" ~ cov1 + cov2 + cov3 + cov4 +
    female, family = binomial(), data = hw1)

Coefficients:
(Intercept)         cov1         cov2         cov3         cov4
    0.23905      0.04159     -0.02512     -0.18594      0.06993
     female
    0.79492

Degrees of Freedom: 134 Total (i.e. Null);  129 Residual
Null Deviance:      164.1
Residual Deviance: 135  AIC: 147
```

```
summary(m1)
```

```

Call:
```

```
glm(formula = treatment == "Treated" ~ cov1 + cov2 + cov3 + cov4 +
    female, family = binomial(), data = hw1)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-1.9216  -0.7418  -0.4914   0.8746   2.4184  

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  0.23905    2.00561   0.119   0.9051    
cov1         0.04159    0.02198   1.892   0.0585 .  
cov2        -0.02512    0.02303  -1.091   0.2754    
cov3        -0.18594    0.04735  -3.927  8.6e-05 ***
cov4         0.06993    0.05053   1.384   0.1664    
female       0.79492    0.44205   1.798   0.0721 .  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 164.08  on 134  degrees of freedom
Residual deviance: 135.02  on 129  degrees of freedom
AIC: 147.02

Number of Fisher Scoring iterations: 4
```

2.1 Storing Probabilities and Linear Predictions

To store the linear predictions (i.e. log odds of the estimated probabilities,) and the estimated probabilities themselves as part of the hw1 data file, I'll use the following commands:

```
hw1$linpred <- m1$linear.predictors
hw1$prob <- m1$fitted.values
```

The remaining tasks in the assignment essentially require you to obtain some numerical (perhaps) and graphical (mandatory) summaries of the estimated probabilities broken down into the two treatment groups.

3 Some Numerical Summaries of the Fitted Probabilities by Treatment Group

3.1 Using dplyr and summarise

The dplyr library can be used to compare the probs across the two treatment groups, along with some piping commands, to create a little data frame of the summaries you're interested in, as follows:

```
hw1 %>%
  group_by(treatment) %>%
  summarise(avg = mean(prob), med = median(prob), sd = sd(prob)) %>%
  arrange(avg)
```

```
# A tibble: 2 x 4
  treatment     avg   med    sd
  <fctr>      <dbl> <dbl> <dbl>
1 Not Treated 0.234 0.188 0.177
2 Treated     0.443 0.430 0.207
```

3.2 Using the by command

Some of you may be more familiar with the by command - that works, as well. . .

```
by(hw1$prob, hw1$treatment, describe)
```

```
hw1$treatment: Not Treated
dd[x, ]
       n  missing distinct     Info     Mean      Gmd      .05      .10
      95        0       95        1   0.2343   0.1891  0.04319  0.05634
     .25      .50      .75      .90      .95
 0.10597  0.18805  0.31990  0.47906  0.55551

lowest : 0.01772653 0.02853116 0.03021174 0.03724054 0.04076704
highest: 0.60114577 0.60689182 0.71653717 0.82781846 0.84216789
--------------------------------------------------------
hw1$treatment: Treated
dd[x, ]
       n  missing distinct     Info     Mean      Gmd      .05      .10
      40        0       40        1   0.4435   0.2411   0.1284   0.1543
     .25      .50      .75      .90      .95
  0.2972   0.4299   0.6032   0.6858   0.7658

lowest : 0.05370511 0.10947830 0.12944077 0.15246158 0.15452007
```

```
highest: 0.68436074 0.69878775 0.76469008 0.78683381 0.86569552
```

3.3 Using the tableone library

Or, you could use the `tableone` library to produce a summarized Table 1 describing our results…

```
varlist <- c("prob", "linpred", "cov1", "cov2", "cov3", "cov4", "female")
tab1 <- CreateTableOne(data=hw1,
                       vars = varlist,
                       factorVars = c("female"),
                       strata = c("treatment"))
print(tab1)
```

```
                     Stratified by treatment
                      Not Treated   Treated        p      test
  n                      95            40                     
  prob (mean (sd))     0.23 (0.18)   0.44 (0.21)  <0.001      
  linpred (mean (sd)) -1.44 (1.09)  -0.29 (1.01)  <0.001      
  cov1 (mean (sd))    47.07 (10.20) 51.42 (10.01)  0.025      
  cov2 (mean (sd))    53.40 (9.79)  50.98 (9.30)   0.187      
  cov3 (mean (sd))    21.45 (4.98)  17.50 (4.74)  <0.001      
  cov4 (mean (sd))    19.66 (4.17)  21.00 (4.83)   0.107      
  female = 1 (%)         38 (40.0)     20 (50.0)   0.378      
```

You could even use non-parametric tests, and report quartiles for the continuous covariates…

```
print(tab1, nonnorm=c("prob", "linpred", "cov1", "cov2", "cov3", "cov4"))
```

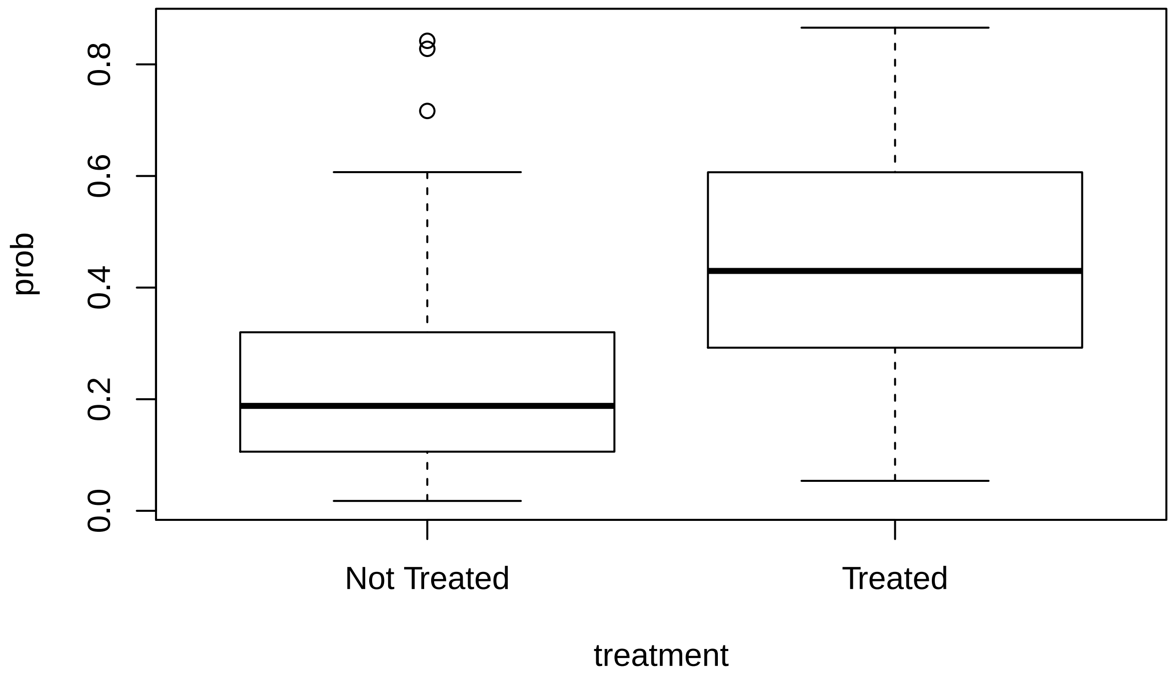
```
                        Stratified by treatment
                         Not Treated          Treated              p     
  n                         95                   40                      
  prob (median [IQR])     0.19 [0.11, 0.32]    0.43 [0.30, 0.60]   <0.001
  linpred (median [IQR]) -1.46 [-2.13, -0.75] -0.28 [-0.86, 0.42]  <0.001
  cov1 (median [IQR])    46.42 [39.97, 53.94] 51.07 [43.32, 60.79]  0.046
  cov2 (median [IQR])    54.08 [47.08, 60.36] 51.66 [44.02, 55.49]  0.122
  cov3 (median [IQR])    21.00 [18.00, 24.00] 18.00 [14.75, 21.00] <0.001
  cov4 (median [IQR])    20.00 [17.00, 23.00] 19.50 [18.00, 24.00]  0.267
  female = 1 (%)            38 (40.0)            20 (50.0)          0.378
                        Stratified by treatment
                         test   
  n                             
  prob (median [IQR])    nonnorm
  linpred (median [IQR]) nonnorm
  cov1 (median [IQR])    nonnorm
  cov2 (median [IQR])    nonnorm
  cov3 (median [IQR])    nonnorm
  cov4 (median [IQR])    nonnorm
  female = 1 (%)
```

4 Plotting the Fitted Probabilities by Treatment Group

OK. So we've seen a numerical summary - let's focus on the important issue - a plot.

The default plot in R might help - it's a boxplot.

```
with(hw1,
     plot(prob ~ treatment)
     )
```

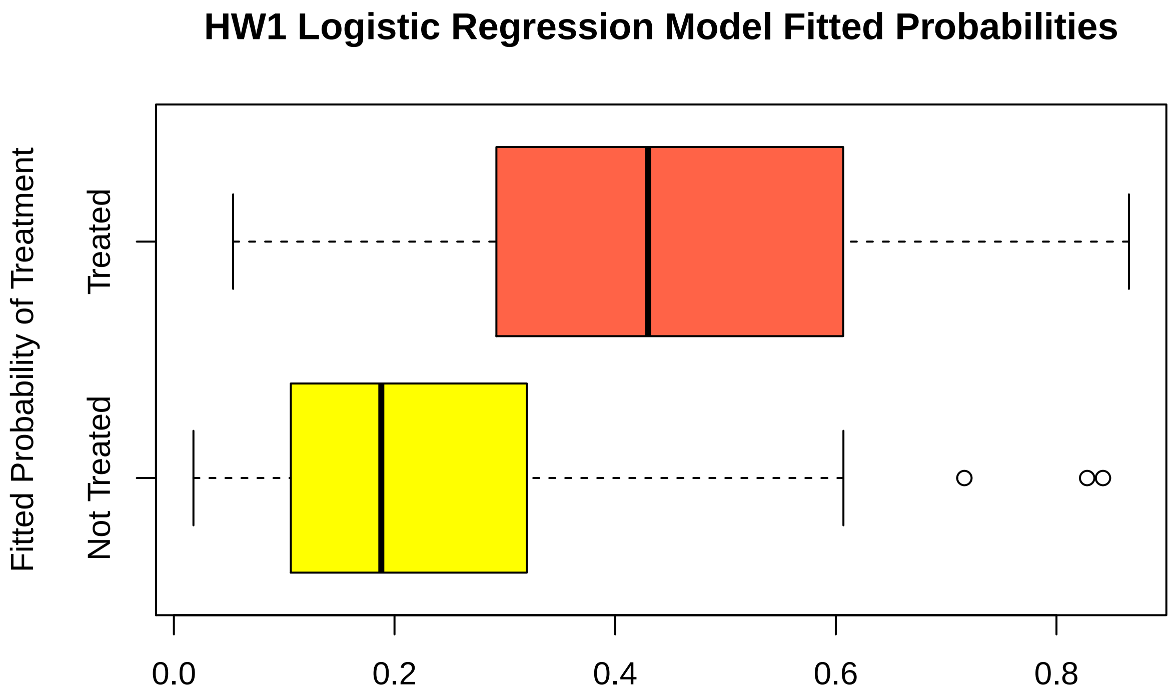


This could be improved in numerous ways with better titles, etc. but this will get the job done for our purposes in a pinch. As we can see from either the plot or the numerical summaries, we have higher probabilities generally but not inevitably in the **treated** group, and a fair amount of overlap between the two treatment groups in terms of their fitted probabilities.

Having obtained the plot you need, a rational course of action would be to save the results and your R script or Markdown file, and move on. But I'm not going to stop there.

4.1 A Fancier Boxplot

```
with(hw1,
     boxplot(prob ~ treatment, ylab="Fitted Probability of Treatment",
             main="HW1 Logistic Regression Model Fitted Probabilities",
             horizontal=T, col=c("yellow", "tomato"))
     )
```



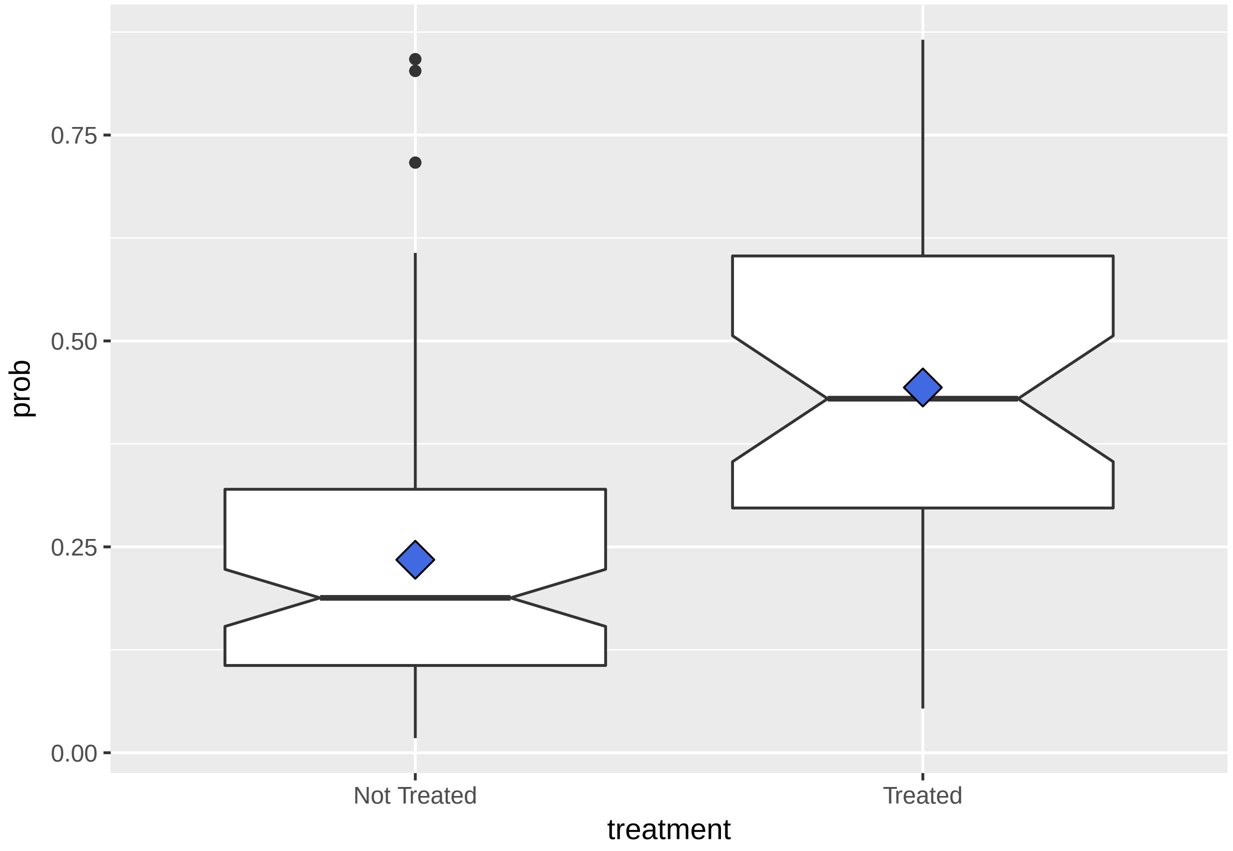
Tomato is a color? Yes[1].

There are lots and lots of ways to do this sort of thing, some of which involve slicker programming tricks than these. R provides help files for every command, and googling can often help you find the command you need.

[1] My favorite reference for R colors is the PDF file at http://goo.gl/kZqtaI

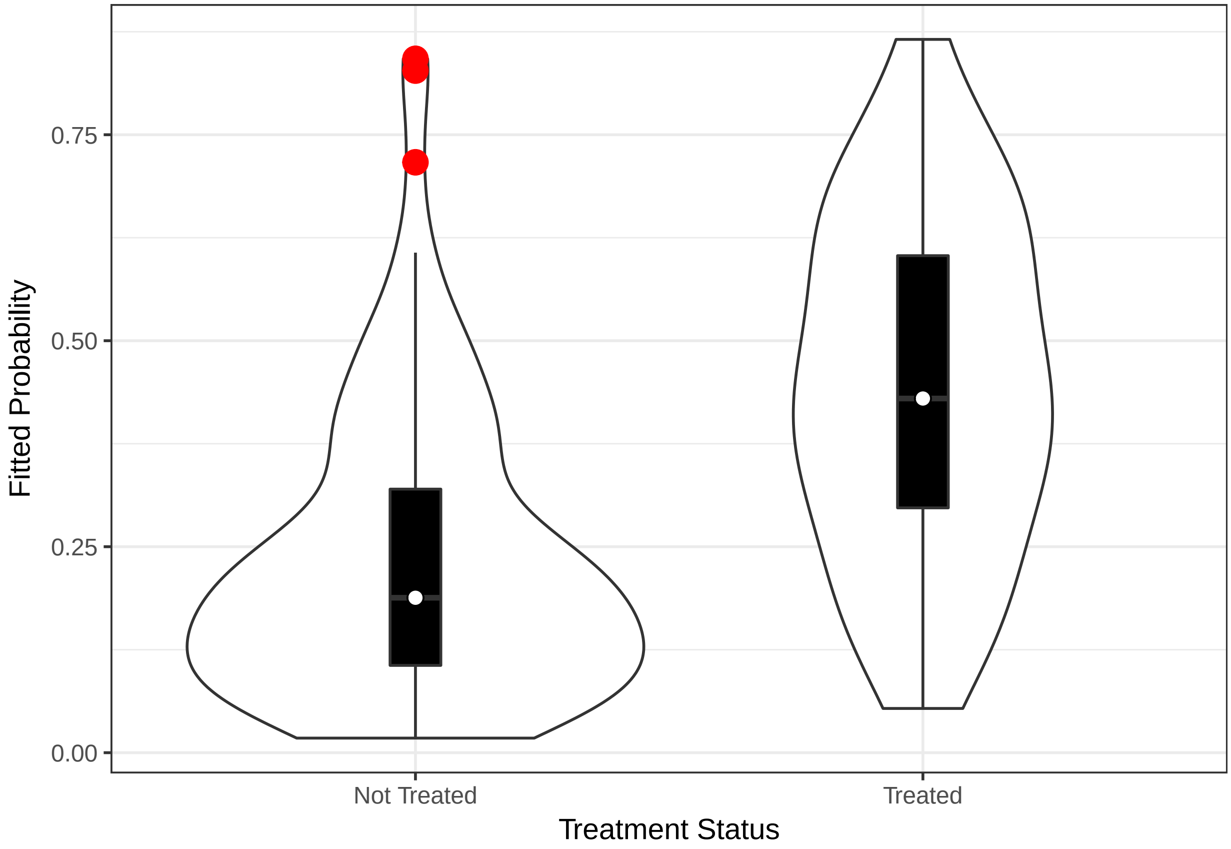
4.2 A Boxplot using ggplot2, with Notches and Means Indicated

```
ggplot(hw1, aes(x = treatment, y = prob)) +
  geom_boxplot(notch=TRUE) +
  stat_summary(fun.y="mean", geom="point", shape=23, size = 5, fill = "royalblue")
```



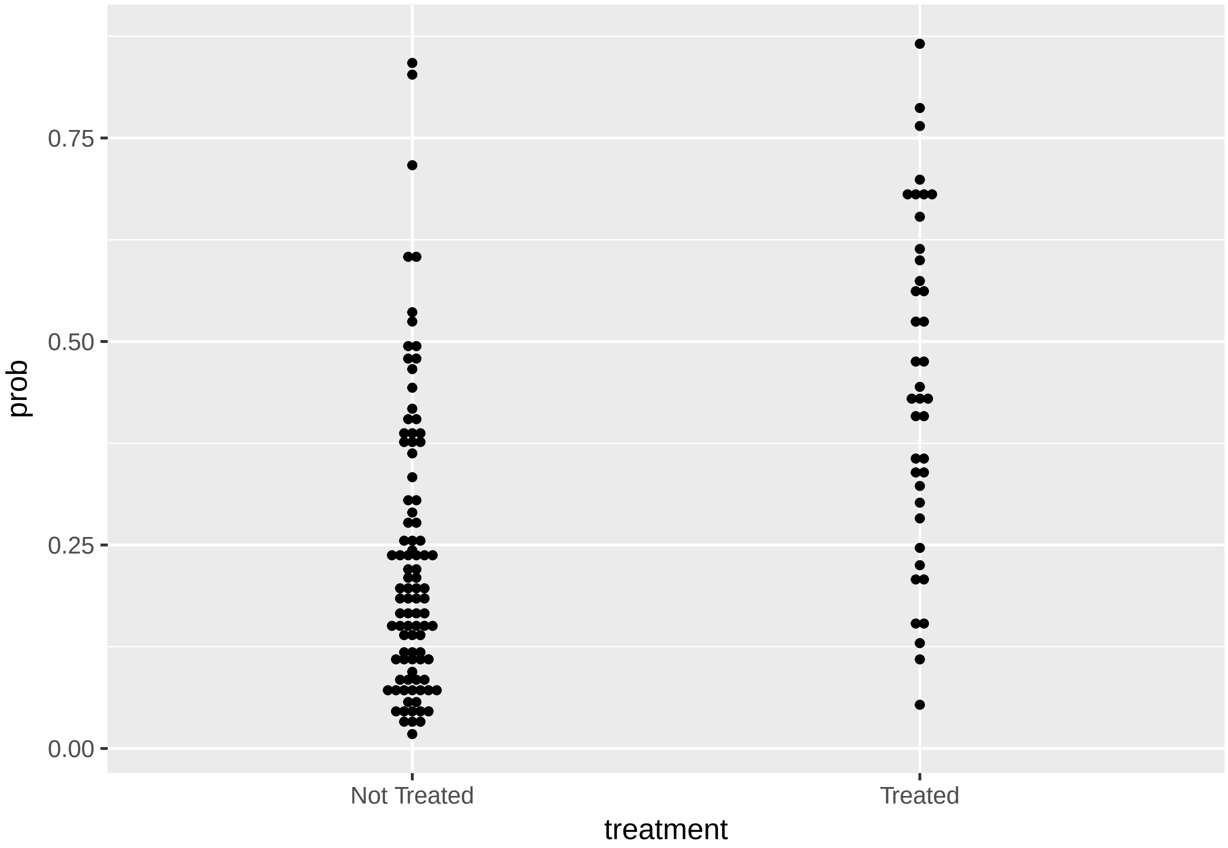
4.3 A Violin Plot

```r
ggplot(hw1, aes(x = treatment, y = prob)) +
  geom_violin() +
  geom_boxplot(width=.1, fill="black",
               outlier.colour="red",
               outlier.size=4) +
  stat_summary(fun.y=median, geom="point",
               fill="white", shape=21, size=2.5) +
  theme_bw() +
  xlab("Treatment Status") + ylab("Fitted Probability")
```



4.4 A DotPlot to compare the probabilities, via ggplot2

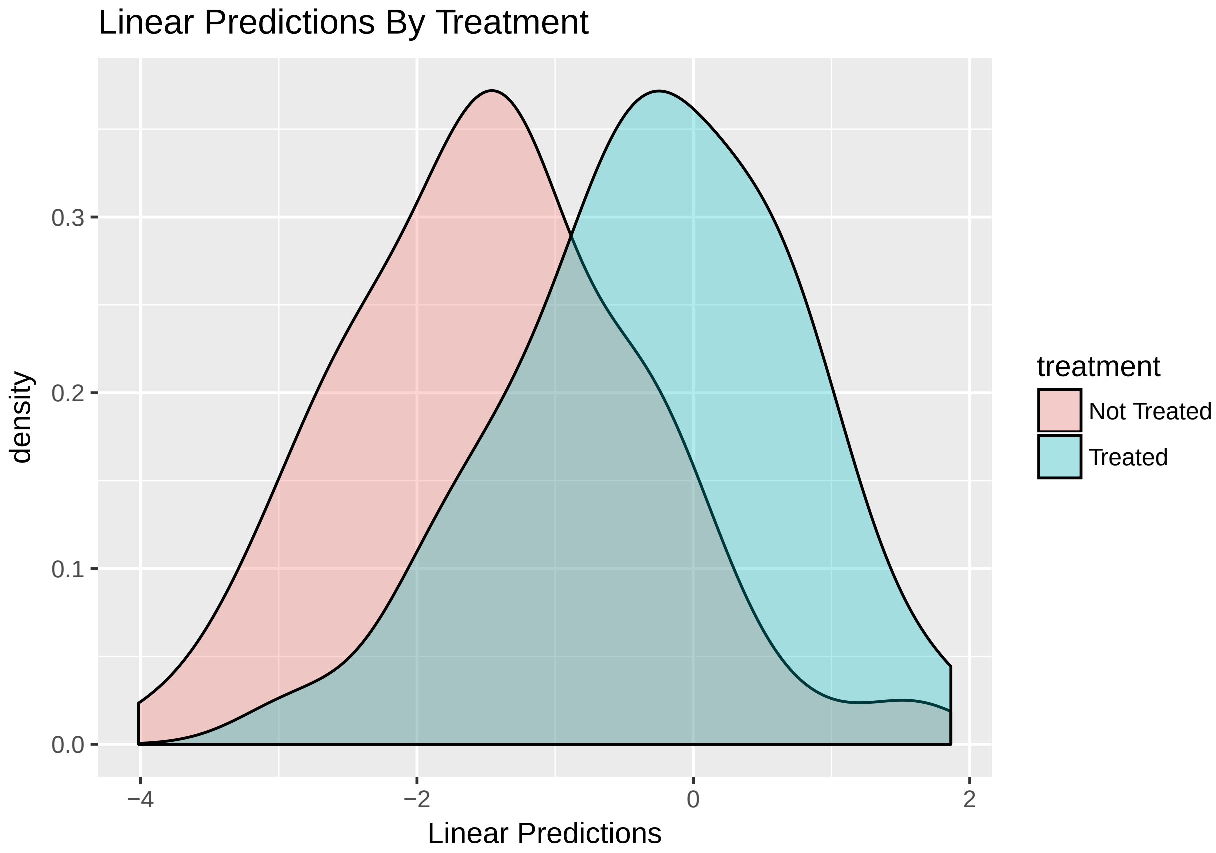
```
ggplot(hw1, aes(x = treatment, y = prob)) +
  geom_dotplot(binaxis="y", binwidth=0.01, stackdir="center")
```



4.5 A Density Plot, using ggplot2

A possibly more impressive picture would be a density plot. The best way to get this (here, I'll look at the linear probability (i.e. log odds of treatment) results rather than the raw probabilities on a 0-1 scale just to see if we observe something different) uses the ggplot2 library again. . .

```
ggplot(hw1, aes(x=linpred, fill=treatment)) +
  geom_density(alpha=0.3) +
  labs(x="Linear Predictions", colour="Type", shape="Type",
       title="Linear Predictions By Treatment")
```

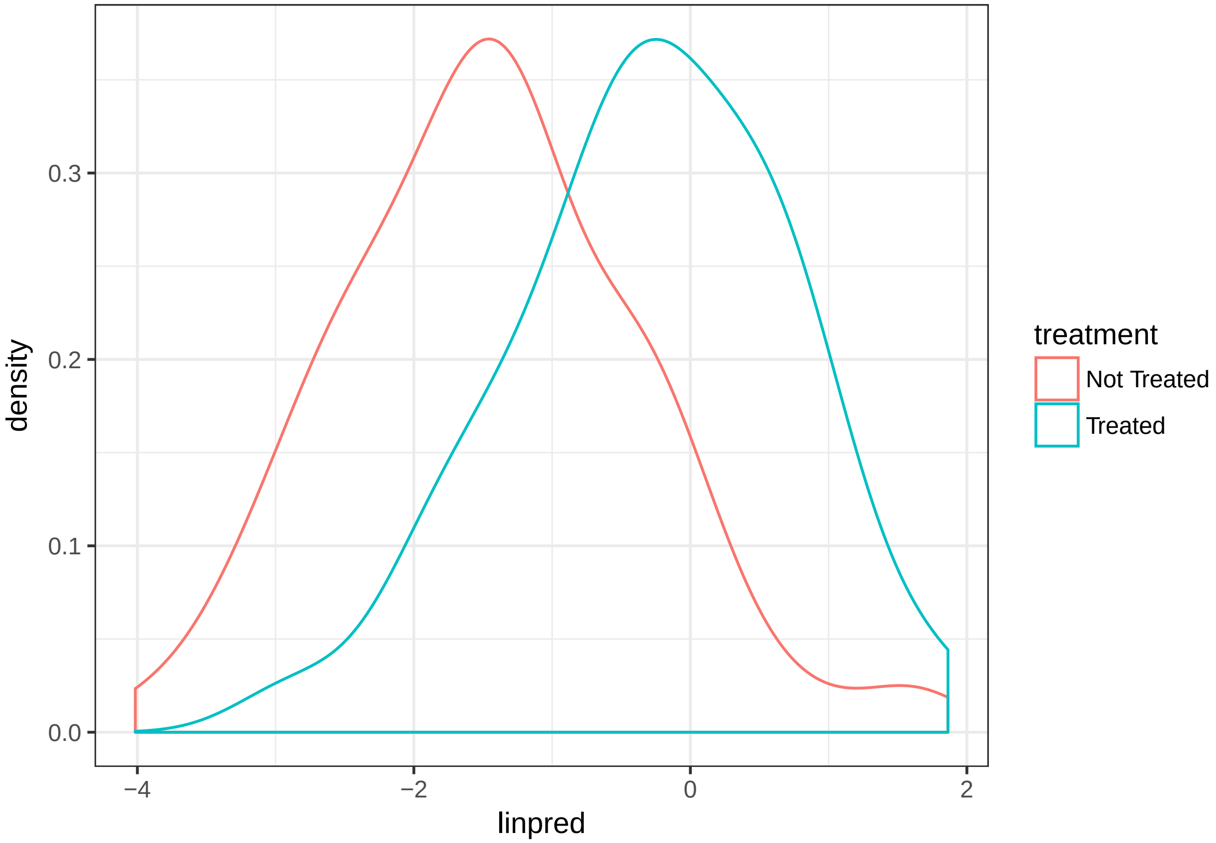


One advantage of the linear probabilities over the raw probability estimates is that the log odds results (linear probabilities) are a bit more likely to follow a normalish distribution. Again, it looks like there is fairly substantial overlap in the fitted probabilities across the treatment groups.

4.6 Another Density Plot, using ggplot2

We can use color instead of fill to indicate the densities.

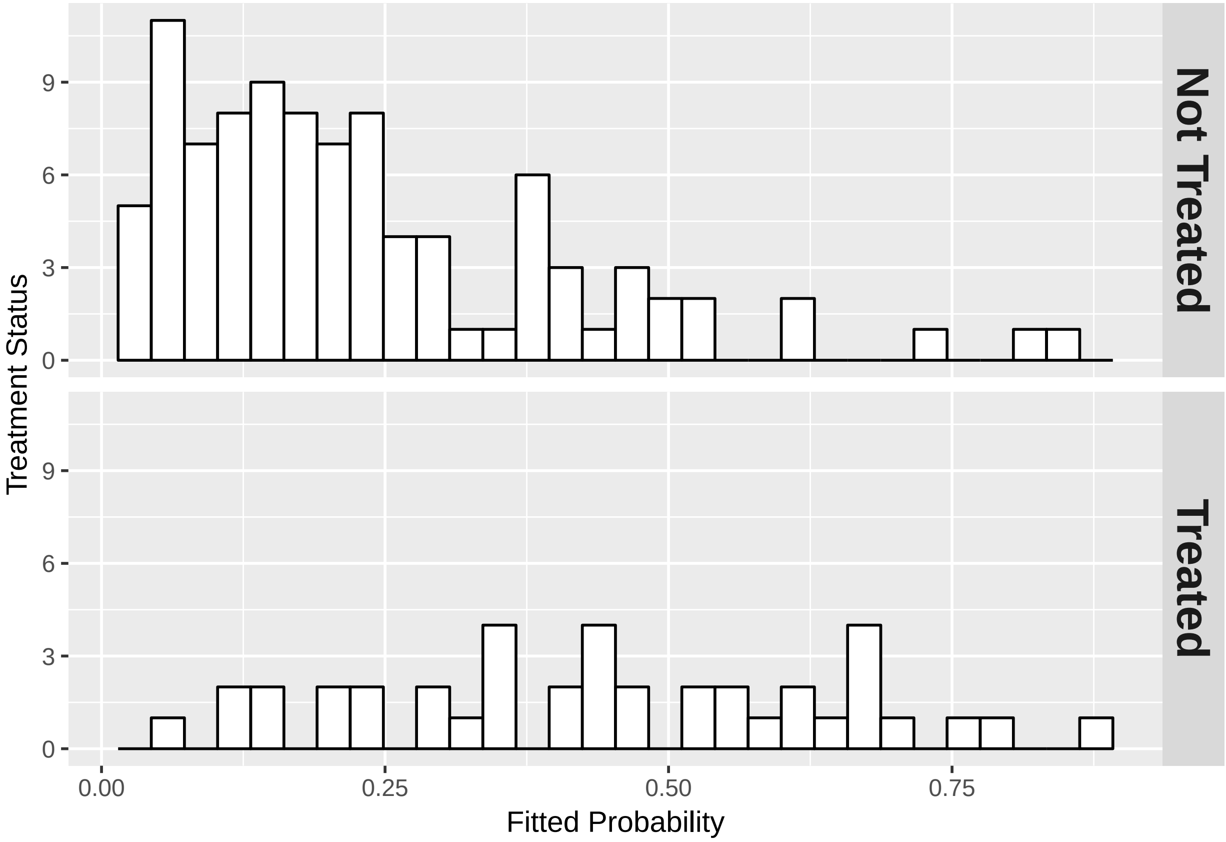
```
ggplot(hw1, aes(x=linpred, color=treatment)) +
  geom_density() +
  theme_bw()
```



4.7 Our Old Standby - Comparing Distributions via Histograms

The slickest approach I have here is this:

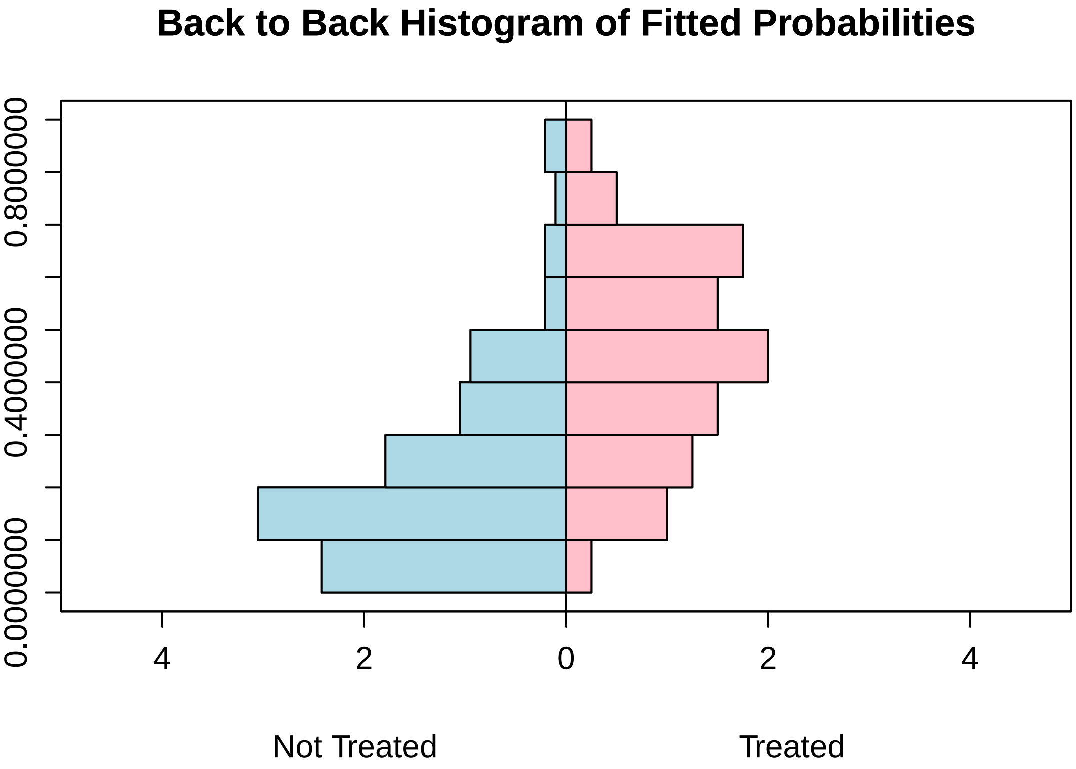
```
ggplot(hw1, aes(x = prob)) +
  geom_histogram(fill="white", color="black") +
  facet_grid(treatment ~ .) +
  theme(strip.text = element_text(face="bold", size=rel(1.5))) +
  xlab("Fitted Probability") + ylab("Treatment Status")
```



4.8 A Back-to-Back Histogram

A former student suggested this approach, from the Hmisc library. There are likely better ways to get such a plot out of R.

```
# Note: requires Hmisc library be loaded
outhist <- histbackback(split(hw1$prob, hw1$treatment), probability=TRUE,
                        xlim = c(-5,5),
                        main="Back to Back Histogram of Fitted Probabilities")
barplot(-outhist$left, col="light blue", horiz=TRUE, space=0, add=TRUE, axes=FALSE)
barplot(outhist$right, col="pink", horiz=TRUE, space=0, add=TRUE, axes=FALSE)
```



5 What About the ROC Curve and C Statistic?

Recall that our model `m1` was

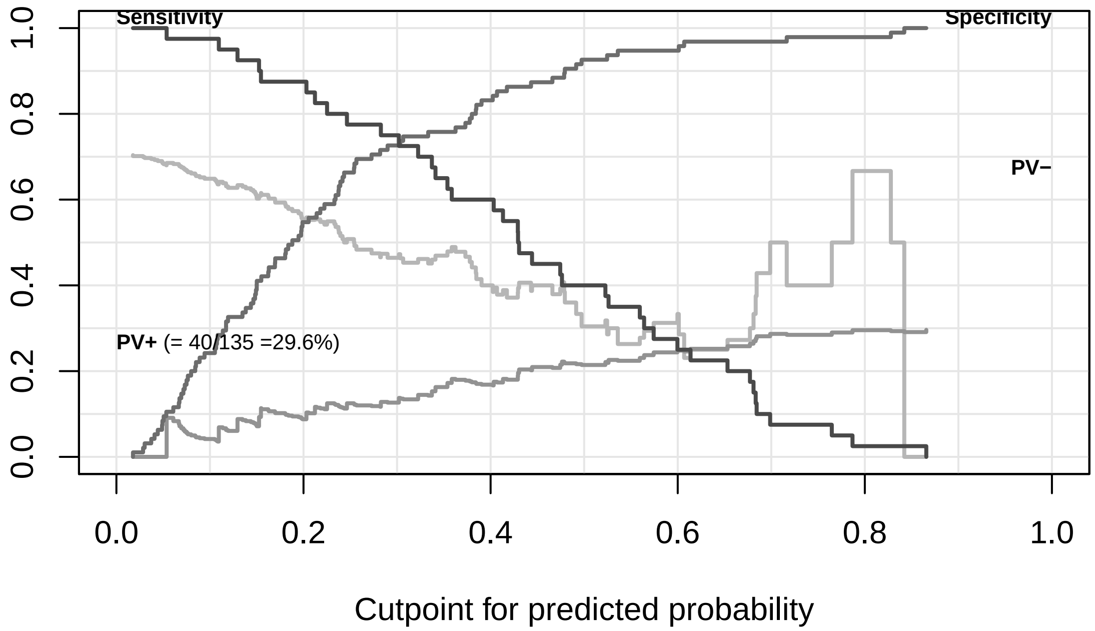
```
Call:  glm(formula = treatment == "Treated" ~ cov1 + cov2 + cov3 + cov4 +
female, family = binomial(), data = hw1)
```

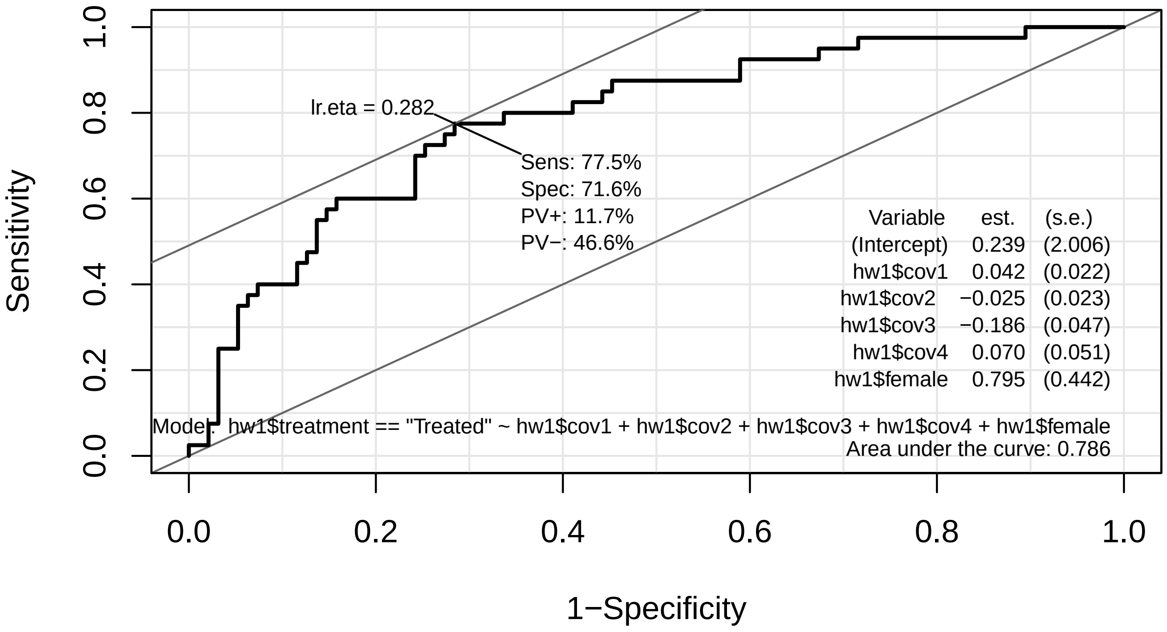
Since we're looking at a logistic regression, someone in a previous version of this class asked if I could show you how I get the C statistic (area under the ROC curve) for such a model. I usually use the `Epi` library . . .

Note that we need to specify the formula (abbreviated form in the ROC function) again, but that's it to get these results.

- The C statistic (area under the curve) for this logistic regression model is 0.786
- Very briefly, the ability of the model's predicted values to discriminate between patients with one outcome vs. the other is quantified by the area under the curve, also called the C statistic or concordance index, which ranges from 0.5 (discrimination is not better than chance) to 1.0 (perfect discriminating power.)
- The ROC procedure comes from signal detection theory and has been adopted into the language of diagnostic testing, essentially treating the response in the logistic regression model as the true status variable, and the set of predictors as the test to be evaluated by things like sensitivity, specificity, and positive and negative predictive values based on dichotomizing along the levels of the predictor set.
- For more on the ROC, visit Wikipedia for Receiver Operating Characteristic. Or try Google.
- A value of 0.786 would indicate a less-than-terrific model in terms of this issue. Values of 0.8 or even 0.9 are usually needed to declare the model to be reasonably accurate in this sense.

```r
library(Epi)
par(mfrow=c(2,1))
ROC(form=(hw1$treatment=="Treated" ~ hw1$cov1 +
              hw1$cov2 + hw1$cov3 + hw1$cov4 + hw1$female))
```


Sensitivity
Specificity
PV−
PV+ (= 40/135 =29.6%)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Cutpoint for predicted probability


lr.eta = 0.282
Sens: 77.5%
Spec: 71.6%
PV+: 11.7%
PV−: 46.6%
Variable est. (s.e.)
(Intercept) 0.239 (2.006)
hw1$cov1 0.042 (0.022)
hw1$cov2 −0.025 (0.023)
hw1$cov3 −0.186 (0.047)
hw1$cov4 0.070 (0.051)
hw1$female 0.795 (0.442)
Model: hw1$treatment == "Treated" ~ hw1$cov1 + hw1$cov2 + hw1$cov3 + hw1$cov4 + hw1$female
Area under the curve: 0.786
Sensitivity
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1−Specificity