Confusion Matrix

Manu Nellutla

8/26/2020

Confusion Matrix

Measuring a model by accuracy can be misleading as the data used to train and test can be baised or have a uneven representation of the desired population mix.

To address this issue we often use **Confusion Matrix** to validate our models performance based on few additional metrics. But first a confusion matrix

$n = TP + TN + FP + FN$		**Actual**	
		Yes	**No**
Predicted	**Yes**	TP	FP
	No	FN	TN

Intro

```
population (n) = all rows = (TP + FP + TN + FN)
```

- $Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)}$ - proportion of correct classifications.
- $Precision = \frac{TP}{TP+FP}$ - This is also called *'Pos Prediction'*
- $Recall = \frac{TP}{TP+FN}$ - This is also called *'Sensitivity'*

load data

```
library(dslabs)
data(heights)
summary(heights)
```

```
##      sex          height     
##  Female:238   Min.   :50.00  
##  Male  :812   1st Qu.:66.00  
##               Median :68.50  
##               Mean   :68.32  
##               3rd Qu.:71.00  
##               Max.   :82.68  
```

split dataset

Using **caret.createDataPartion** [https://www.rdocumentation.org/packages/caret/versions/6.0-86/topics/createDataPartition (https://www.rdocumentation.org/packages/caret/versions/6.0-86/topics/createDataPartition)]

```
library("caret")
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
y = heights$sex
X = heights$height
set.seed(2, sample.kind = "Rounding")
```

```
## Warning in set.seed(2, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
# split at 0,5
test_index = createDataPartition(y, times = 1, p = 0.5, list=FALSE)

test_set = heights[test_index,]
train_set = heights[-test_index,]

summary(test_set)
```

```
##      sex          height
##  Female:119   Min.   :50.00
##  Male  :406   1st Qu.:66.00
##                Median :68.11
##                Mean   :68.36
##                3rd Qu.:71.00
##                Max.   :82.68
```

```
summary(train_set)
```

```
##      sex          height
##  Female:119   Min.   :50.00
##  Male  :406   1st Qu.:66.00
##                Median :68.90
##                Mean   :68.28
##                3rd Qu.:71.00
##                Max.   :80.00
```

predic with best accuracy

Now that we know the best cutoff height is 64 lets predict with new cutoff

```
library(purrr)
```

```
##
## Attaching package: 'purrr'
## The following object is masked from 'package:caret':
##
##     lift
```

```
y_hat <- ifelse(test_set$height > 64, "Male", "Female") %>% factor(levels = levels(test_set$sex))
y_hat <- factor(y_hat)
mean(y_hat == test_set$sex)
```

```
## [1] 0.8171429
```

confusion matrix

```
confusionMatrix(data=y_hat, reference=test_set$sex)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Female Male
##     Female     50   27
##     Male       69  379
##
##                Accuracy : 0.8171
##                  95% CI : (0.7814, 0.8493)
##     No Information Rate : 0.7733
##     P-Value [Acc > NIR] : 0.008354
##
##                   Kappa : 0.4041
##
##  Mcnemar's Test P-Value : 2.857e-05
##
##             Sensitivity : 0.42017
##             Specificity : 0.93350
##          Pos Pred Value : 0.64935
##          Neg Pred Value : 0.84598
##              Prevalence : 0.22667
##          Detection Rate : 0.09524
##    Detection Prevalence : 0.14667
##       Balanced Accuracy : 0.67683
##
##        'Positive' Class : Female
##
```

Understanding results

If you look at the results we understand that **accuracy is about 81%**.

That looks good. But is it?

If you look at the "**Pos Prediction above it is 64%**". This is our actual precision which tells us that proportion of 'True Positives' are low.

Sensitivity/ Recall is the proportion of correct classification for a class with in the population. > *In the above table*: predicted female = 50; actual female = 119 (look at test set summary)

Why is this? it is because of the cutoff we chose. I chose to classify male if height > 64. However, the mean of female height is 65. This makes the model intutively wrong.

But ‘Male’ height cutoff was calculated as $(\mu - 2 * sd)$ which is right based on our 95% CI.

What we can infer is that the **PREVALANCE** of the dataset, which means proportion of one class within the population, is uneven..

Maximize F1 score

One preferred metric is **balanced accuracy**. Because specificity and sensitivity are rates, it is more appropriate to compute the harmonic average. In fact, the F1-score, a widely used one-number summary, is the harmonic average of precision and recall.

$$F1 = \frac{2 * [precision * recall]}{[precision + recall]}$$

```
# maximize F-score
cutoff <- seq(61, 70)
F_1 <- map_dbl(cutoff, function(x){
  y_hat <- ifelse(train_set$height > x, "Male", "Female") %>%
    factor(levels = levels(test_set$sex))
  F_meas(data = y_hat, reference = factor(train_set$sex))
})

F_1
```

```
##  [1] 0.2112676 0.3225806 0.4393064 0.5656566 0.6052632 0.6142322 0.5820433
##  [8] 0.5573770 0.5080831 0.4679089
```

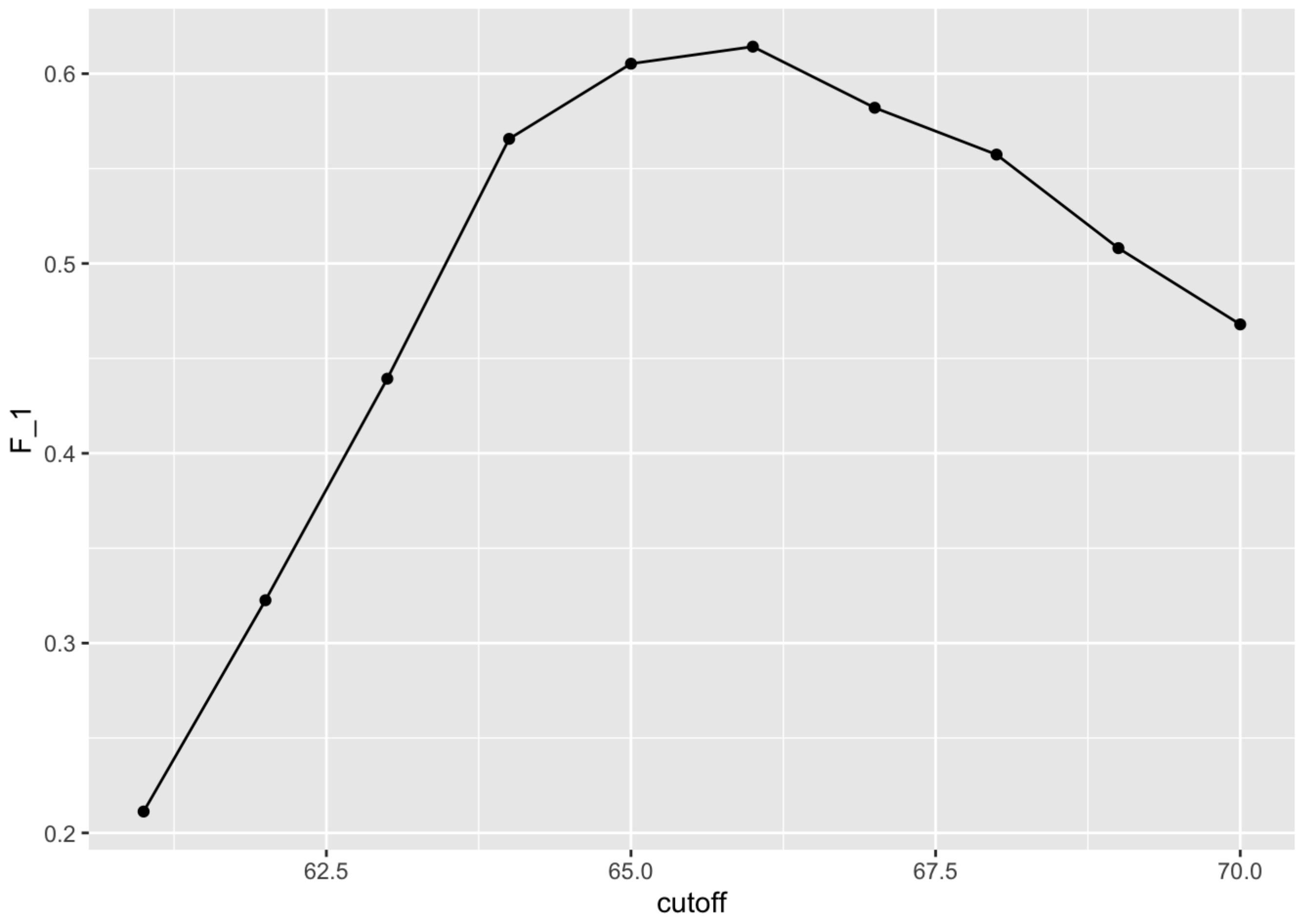
```
max(F_1)
```

```
## [1] 0.6142322
```

For the same cutoff we find F1 score using “**F_means()**” function of *caret* package. Max F1 score is around 61%.

lets plot of F1 scores

```
data.frame(cutoff, F_1) %>%
  ggplot(aes(cutoff, F_1)) +
  geom_point() +
  geom_line()
```



```
max(F_1)
```

```
## [1] 0.6142322
```

```
best_cutoff <- cutoff[which.max(F_1)]
best_cutoff
```

```
## [1] 66
```

From the R console we see the best cutoff is not at 66 and not 64. This is better as the mean(female_heights) is 65.

lets check our metrics again for best cutoff

```
y_hat <- ifelse(test_set$height > best_cutoff, "Male", "Female") %>%
  factor(levels = levels(test_set$sex))
sensitivity(data = y_hat, reference = test_set$sex)
```

```
## [1] 0.6806723
```

```
specificity(data = y_hat, reference = test_set$sex)
```

```
## [1] 0.8349754
```

what about confusion matrix

```
confusionMatrix(data=y_hat, reference=test_set$sex)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Female Male
##     Female     81   67
##     Male       38  339
##
##                Accuracy : 0.8
##                  95% CI : (0.7632, 0.8334)
##     No Information Rate : 0.7733
##     P-Value [Acc > NIR] : 0.078192
##
##                   Kappa : 0.4748
##
##  Mcnemar's Test P-Value : 0.006285
##
##             Sensitivity : 0.6807
##             Specificity : 0.8350
##          Pos Pred Value : 0.5473
##          Neg Pred Value : 0.8992
##              Prevalence : 0.2267
##          Detection Rate : 0.1543
##    Detection Prevalence : 0.2819
##       Balanced Accuracy : 0.7578
##
##        'Positive' Class : Female
##
```

Lets compare before and after

Without F1	With F1

```
Confusion Matrix and Statistics

          Reference
Prediction Female Male
    Female     50   27
    Male       69  379

               Accuracy : 0.8171
                 95% CI : (0.7814, 0.8493)
    No Information Rate : 0.7733
    P-Value [Acc > NIR] : 0.008354

                  Kappa : 0.4041

 Mcnemar's Test P-Value : 2.857e-05

            Sensitivity : 0.42017
            Specificity : 0.93350
         Pos Pred Value : 0.64935
         Neg Pred Value : 0.84598
             Prevalence : 0.22667
         Detection Rate : 0.09524
   Detection Prevalence : 0.14667
      Balanced Accuracy : 0.67683

       'Positive' Class : Female
```

```
Confusion Matrix and Statistics

          Reference
Prediction Female Male
    Female     81   67
    Male       38  339

               Accuracy : 0.8
                 95% CI : (0.7632, 0.8334)
    No Information Rate : 0.7733
    P-Value [Acc > NIR] : 0.078192

                  Kappa : 0.4748

 Mcnemar's Test P-Value : 0.006285

            Sensitivity : 0.6807
            Specificity : 0.8350
         Pos Pred Value : 0.5473
         Neg Pred Value : 0.8992
             Prevalence : 0.2267
         Detection Rate : 0.1543
   Detection Prevalence : 0.2819
      Balanced Accuracy : 0.7578

       'Positive' Class : Female
```

ROC Curves

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

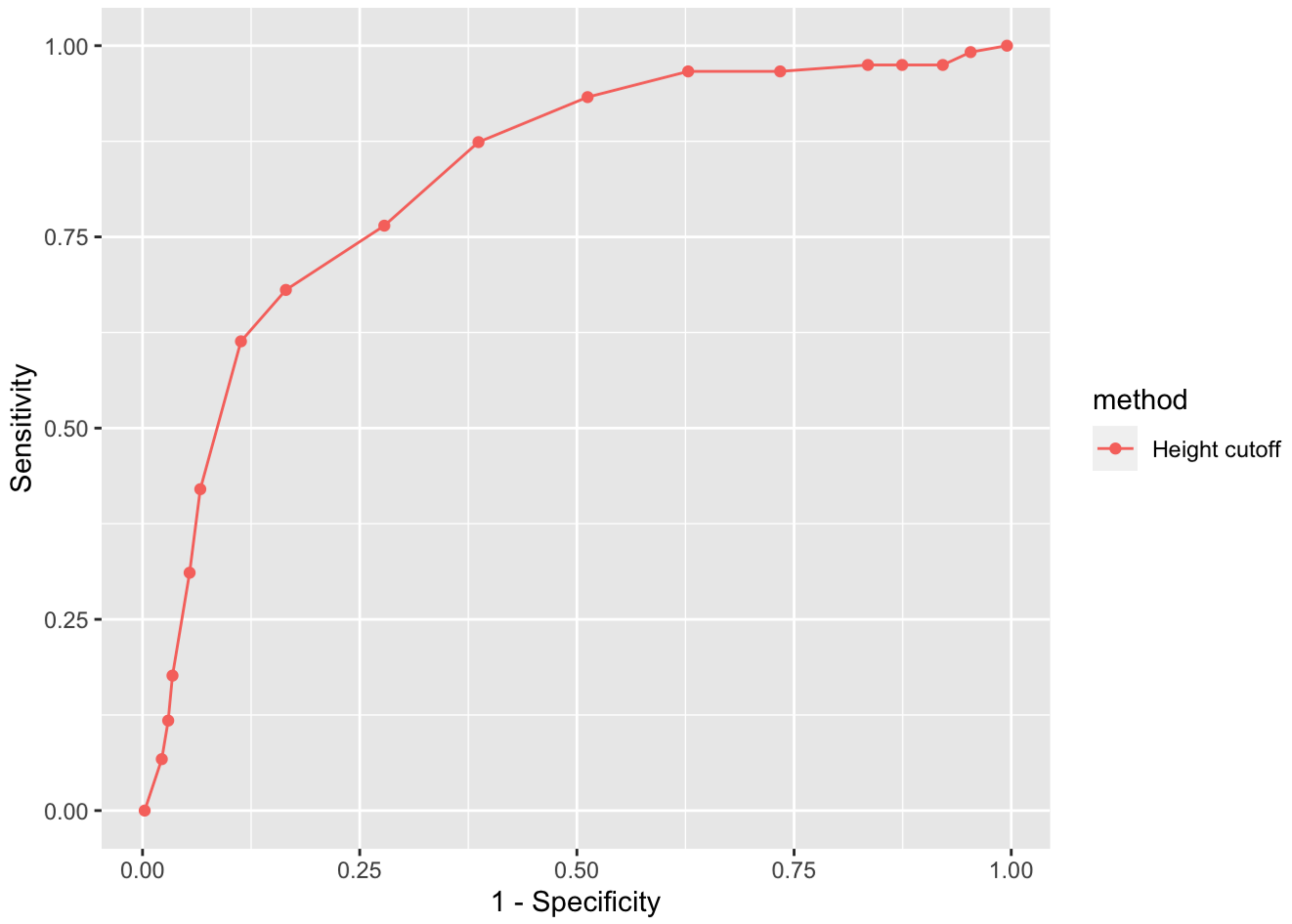
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
cutoffs <- c(50, seq(60, 75), 80)
height_cutoff <- map_df(cutoffs, function(x){
  y_hat <- ifelse(test_set$height > x, "Male", "Female") %>%
    factor(levels = c("Female", "Male"))
   list(method = "Height cutoff",
        FPR = 1-specificity(y_hat, test_set$sex),
        TPR = sensitivity(y_hat, test_set$sex))
})
```

Plotting the ROC curve

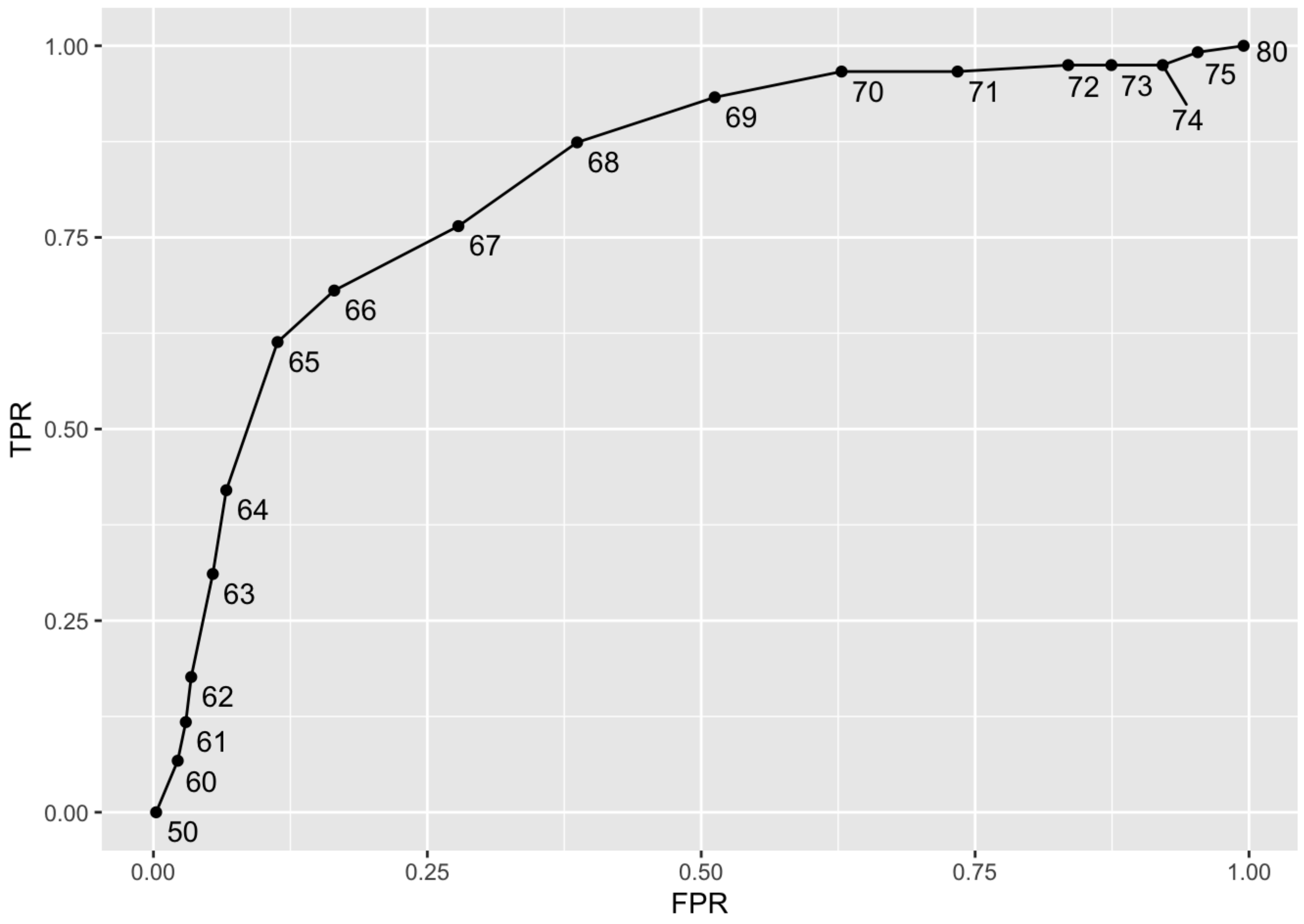
A widely used plot that does this is the **receiver operating characteristic (ROC)** curve. The ROC curve plots *sensitivity (TPR)* versus *1 - specificity or the false positive rate (FPR)*.

```
# plot both curves together
height_cutoff %>%
  ggplot(aes(FPR, TPR, color = method)) +
  geom_line() +
  geom_point() +
  xlab("1 - Specificity") +
  ylab("Sensitivity")
```



another plot

```r
library(ggrepel)
map_df(cutoffs, function(x){
  y_hat <- ifelse(test_set$height > x, "Male", "Female") %>%
    factor(levels = c("Female", "Male"))
   list(method = "Height cutoff",
        cutoff = x,
        FPR = 1-specificity(y_hat, test_set$sex),
        TPR = sensitivity(y_hat, test_set$sex))
}) %>%
  ggplot(aes(FPR, TPR, label = cutoff)) +
  geom_line() +
  geom_point() +
  geom_text_repel(nudge_x = 0.01, nudge_y = -0.01)
```



Precision-Recall plot.

However, ROC curves have one weakness and it is that neither of the measures plotted depend on prevalence. In cases in which prevalence matters, we may instead make a precision-recall plot, which has a similar idea with ROC curve.

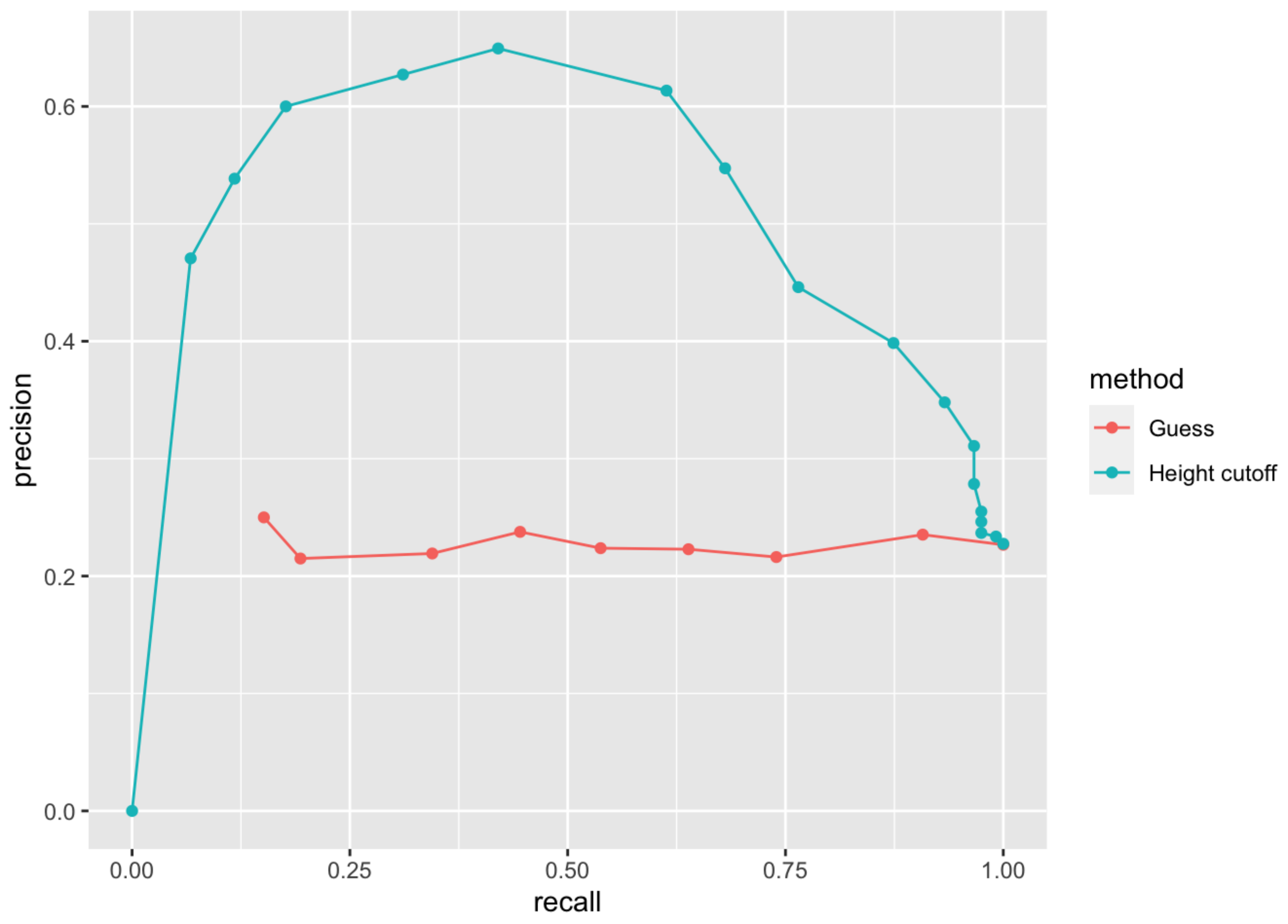
```r
probs <- seq(0, 1, length.out = 10)
guessing <- map_df(probs, function(p){
  y_hat <- sample(c("Male", "Female"), length(test_index),
                  replace = TRUE, prob=c(p, 1-p)) %>%
    factor(levels = c("Female", "Male"))
  list(method = "Guess",
    recall = sensitivity(y_hat, test_set$sex),
    precision = precision(y_hat, test_set$sex))
})

height_cutoff <- map_df(cutoffs, function(x){
  y_hat <- ifelse(test_set$height > x, "Male", "Female") %>%
    factor(levels = c("Female", "Male"))
  list(method = "Height cutoff",
       recall = sensitivity(y_hat, test_set$sex),
    precision = precision(y_hat, test_set$sex))
})

bind_rows(guessing, height_cutoff) %>%
  ggplot(aes(recall, precision, color = method)) +
  geom_line() +
  geom_point()
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



with relevel

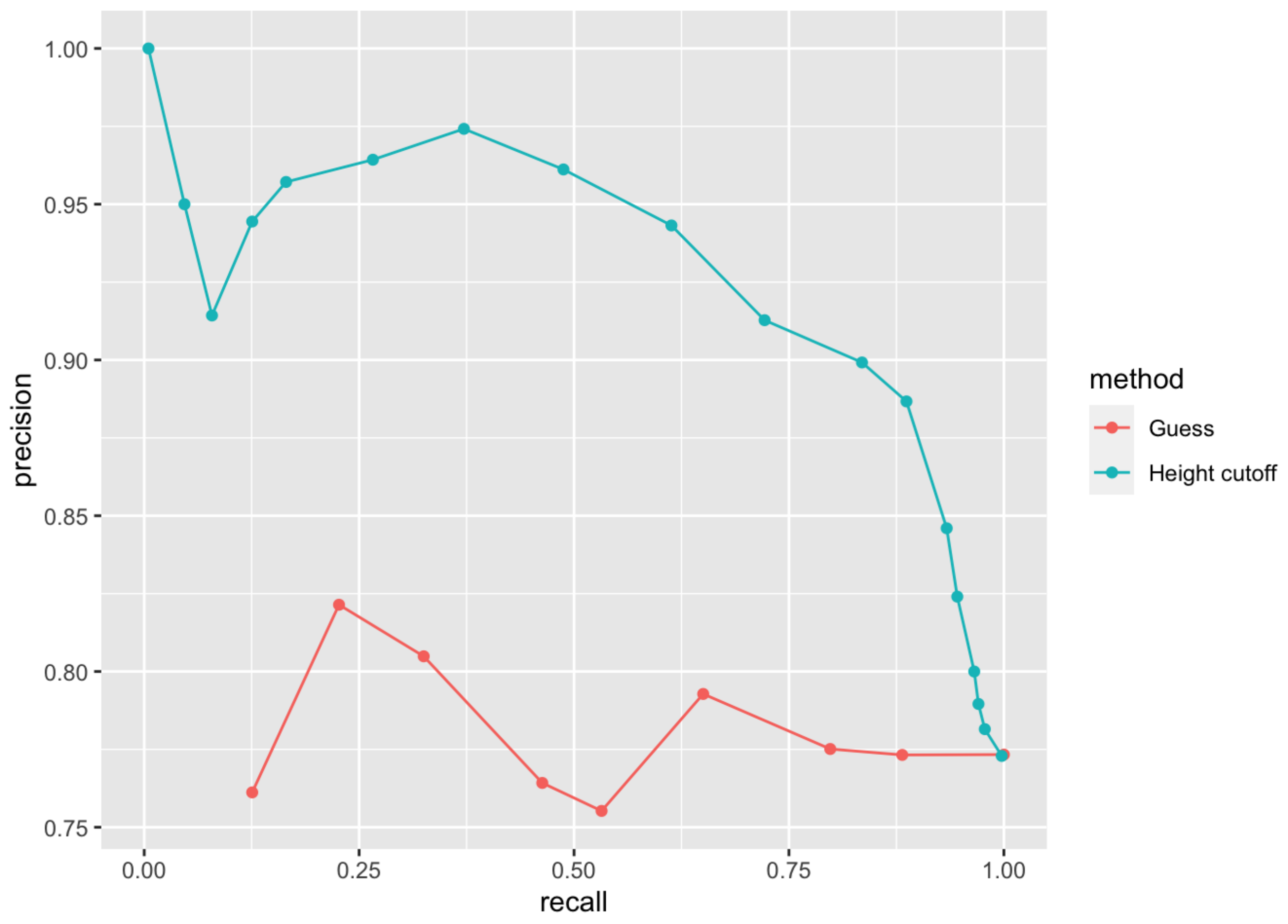
Meaning … If we change positives to mean male instead of females.

```
guessing <- map_df(probs, function(p){
  y_hat <- sample(c("Male", "Female"), length(test_index), replace = TRUE,
                  prob=c(p, 1-p)) %>%
    factor(levels = c("Male", "Female"))
  list(method = "Guess",
    recall = sensitivity(y_hat, relevel(test_set$sex, "Male", "Female")),
    precision = precision(y_hat, relevel(test_set$sex, "Male", "Female")))
})

height_cutoff <- map_df(cutoffs, function(x){
  y_hat <- ifelse(test_set$height > x, "Male", "Female") %>%
    factor(levels = c("Male", "Female"))
  list(method = "Height cutoff",
       recall = sensitivity(y_hat, relevel(test_set$sex, "Male", "Female")),
    precision = precision(y_hat, relevel(test_set$sex, "Male", "Female")))
})
bind_rows(guessing, height_cutoff) %>%
  ggplot(aes(recall, precision, color = method)) +
  geom_line() +
  geom_point()
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```


1.00
0.95
0.90
0.85
0.80
0.75
precision
0.00
0.25
0.50
0.75
1.00
recall
method
Guess
Height cutoff