ML Basics - Example1

Manu Nellutla

8/24/2020

Example1 - Predict population Sex

This Example exercise tries to “Predict Sex using height”. Based on the titel we can infer that this is a classification example. We will be predicting our **outcome as 2 classes: “Male or Female”** on the basis of our **features/covariate: “Height”**.

We are using a dataset which can be called labeled dataset which already has height and sex association.

Use library dslabs

```
library(dslabs)
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
```

Heights Dataset

From dslabs we load heights dataset.

```
#load heights data set using
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
```

Define our Parameters. Outcome (Y) and Predictors (X)

Since we are predicting “sex(Male or Female)” this is our Y based on the predictor/Feature “heights” which is our X.

```
y = heights$sex
X = heights$height
```

ML Step 1: Split data into training and test.

‘Caret’ library is used for splitting the data using **createDataPartion** function.

[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
```

```
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,]
```

ML Step2: Predict sex 2 ways

- Simple guessing without predictors (like a coin toss)
- Using factors -> 'sex'

```
y_hat = sample(c("Male","Female"), length(test_index), replace = TRUE)
table(y_hat)
```

```
## y_hat
## Female   Male
##    267    258
```

```
library(magrittr)
y_hat2 <- sample(c("Male","Female"), length(test_index), replace = TRUE) %>%
  factor(levels=levels(test_set$sex))
table(y_hat2)
```

```
## y_hat2
## Female   Male
##    265    260
```

ML Step3: Compute Accuracy

- compage our guess with test set 'sex'

```
mean(y_hat == test_set$sex)
```

```
## [1] 0.4628571
```

Less than 50%. Lets check the one with factors. Y_hat2

```
mean(y_hat2 == test_set$sex)
```

```
## [1] 0.5238095
```

About 50%. Not quite good.

General intution: Males generally are taller than females

We know males are generally taller than female. Can we use this information in anyway? One logic is: Predict male if height is within 2 standard deviations of avg height of male.

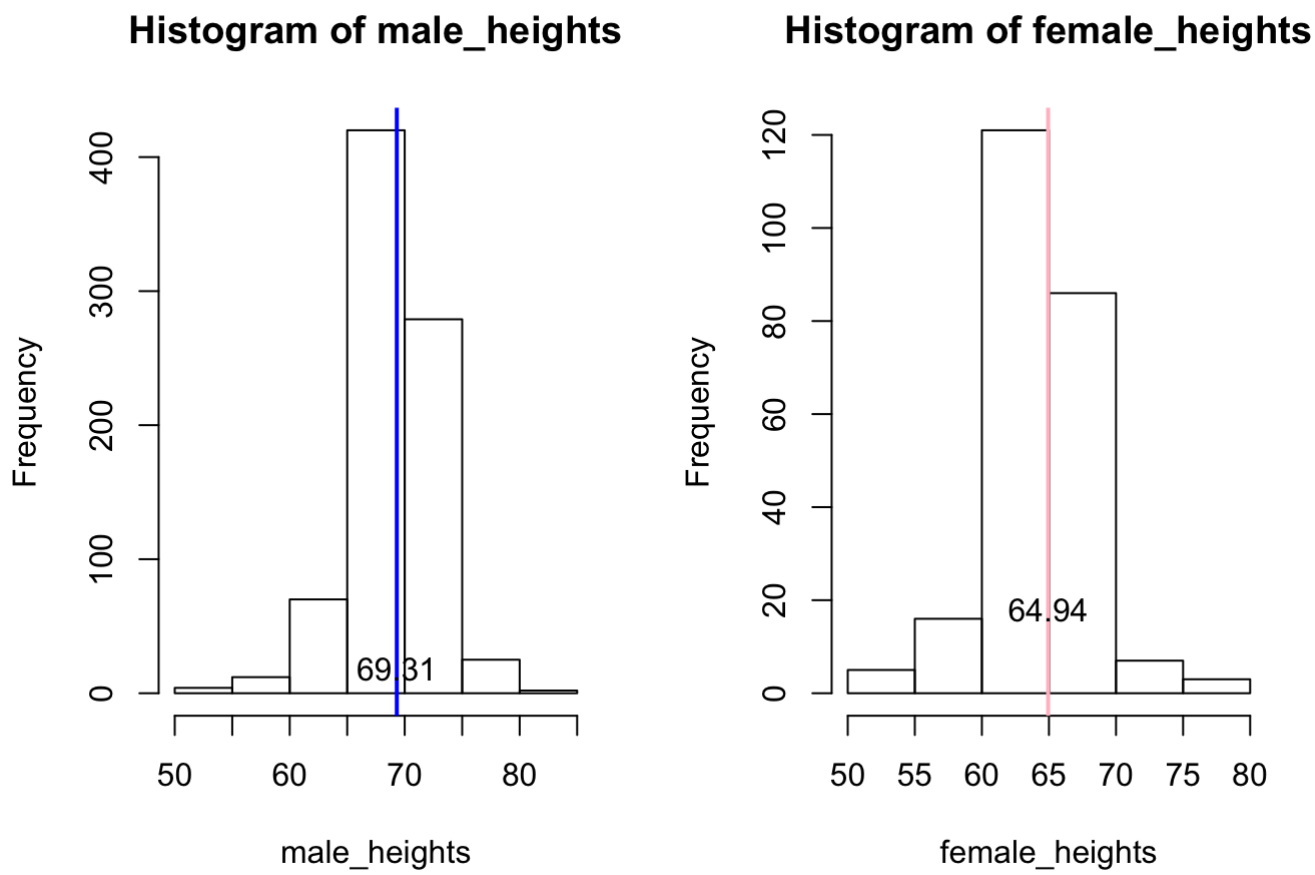
Method 1: Plot hist and calculate

Lets do this: - Separate male and female heights - plot a histogram with a mean on it.

```
male_heights = heights$height[heights$sex == "Male"]
female_heights = heights$height[heights$sex == "Female"]

par(mfrow=c(1,2))

hist(male_heights)
mx = mean(male_heights)
abline(v=mx, col="blue", lw=2)
text(mx, 18 , round(mx, 2))
hist(female_heights)
fx = mean(female_heights)
abline(v=fx, col="pink", lw=2)
text(fx, 18, round(fx,2))
```



Now the aboe plots show: Mean of male heights = 69.31. our range is (mean-2* sd to mean+2*sd)

```
mean(male_heights) - 2 * sd(male_heights)
```

```
## [1] 62.09271
```

62 is our 2 times dist lower end that we can use to say any height between this number is mostly male.

Method 2: R piping command

Another slick way of doing this is by this aggregated function calling..

from heights dataset %>% group_by sex (male, female) %>% get me MEAN of heights and Standard Deveation.

```
heights %>% group_by(sex) %>% summarize(mean(height), sd(height))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 2 x 3
##   sex    `mean(height)` `sd(height)`
##   <fct>           <dbl>        <dbl>
## 1 Female           64.9         3.76
## 2 Male             69.3         3.61
```

If we calculate mean - 2 X SD (69.31 - 2 * 3.611) + 62.088.

adding intution to our algoritm

So we will add this intution to our algorithm

```
y_hat3 <- ifelse(X > 62, "Male", "Female") %>% factor(levels = levels(test_set$sex))
mean(y == y_hat3)
```

```
## [1] 0.7933333
```

WOW! We got about 20% more accuracy. So for male height > 62 we got about 80% accuracy.

ML Step4: Running multiple accuracy using cutoff heights

Lets find accuracy numbers for each height between 61, 75 and plot them.

```
library(purrr)
```

```
##
## Attaching package: 'purrr'
```

```
## The following object is masked from 'package:magrittr':
##
##     set_names
```

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

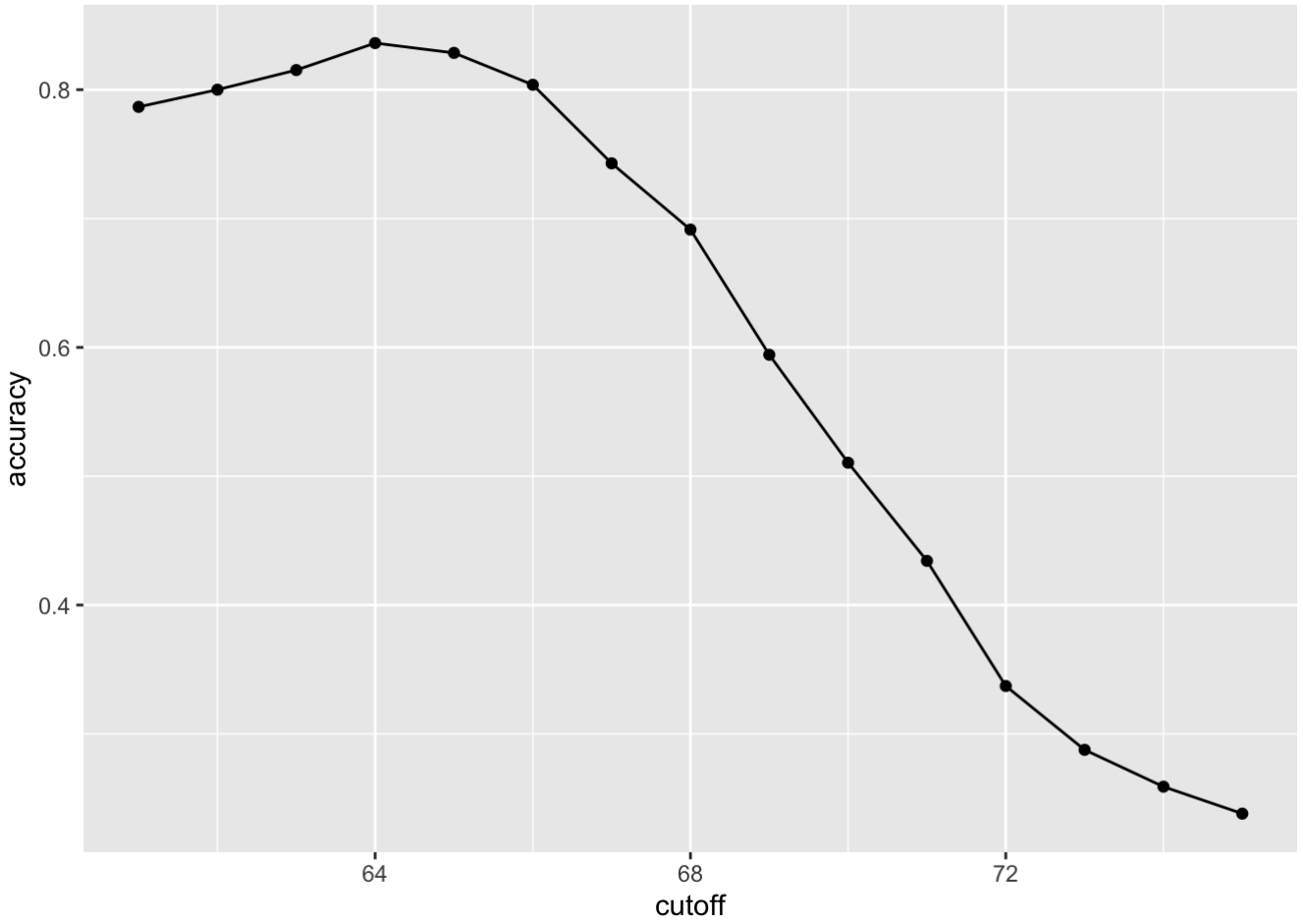
```
cutoff <- seq(61,75)
accuracy <- map_dbl(cutoff, function(x){
  y_hat4 <- ifelse(train_set$height > x, "Male", "Female") %>%
    factor(levels = levels(test_set$sex))
  mean(y_hat4 == train_set$sex)
})
summary(accuracy)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2381  0.3857  0.6914  0.5977  0.8019  0.8362
```

As we see max accuracy is 83.6%

Plot accuracy

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




```
max(accuracy)
```

```
## [1] 0.8361905
```

Best cutoff visually is at 64. Lets compute programatically

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

```
## [1] 64
```

ML Step 5: Predic with best accuracy

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

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

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

Our training accuracy was 83.61% and testing accuracy is 81.71%.

ML Step6: Evaluate accuracy

Is the accuracy we got good? How can we assess. Our logic is mediocre as Mean(female_height) is 65 and our cutoff at 64 says anyone over 64 is “Male”. There is a flaw in the logic.

Lets explore.

Confusion Matrix

is a table that shows True Positives, True Negatives, False Positives and False Negatives.

Tabulate each value of predicted v/s actual

```
table(predicted= y_hat_final, actual=test_set$sex)
```

```
##          actual
## predicted Female Male
##    Female     50   27
##    Male       69  379
```

```
test_set %>%
  mutate(y_hat_cf = y_hat_final) %>%
  group_by(sex) %>%
  summarize(accuracy = mean(y_hat_cf==sex))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 2 x 2
##   sex    accuracy
##   <fct>     <dbl>
## 1 Female    0.420
## 2 Male      0.933
```

```
prev <- mean(y=='Male')
prev
```

```
## [1] 0.7733333
```

confusion matrix separately

```
confusionMatrix(data=y_hat_final, 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
##
```