Introduction to Logistic Regression (Part 2)

Heaven Klair

5/6/2022

= ## Importing the dataset

```
url <- "http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data"
dataset = read.csv(url, header=FALSE)
head(dataset)
```

```
##   V1 V2 V3  V4  V5 V6 V7  V8 V9 V10 V11 V12 V13 V14
## 1 63  1  1 145 233  1  2 150  0 2.3   3 0.0 6.0   0
## 2 67  1  4 160 286  0  2 108  1 1.5   2 3.0 3.0   2
## 3 67  1  4 120 229  0  2 129  1 2.6   2 2.0 7.0   1
## 4 37  1  3 130 250  0  0 187  0 3.5   3 0.0 3.0   0
## 5 41  0  2 130 204  0  2 172  0 1.4   1 0.0 3.0   0
## 6 56  1  2 120 236  0  0 178  0 0.8   1 0.0 3.0   0
```

After importing the data, we see that none of the columns are labeled. So we can name the columns looking at the UCI website.

```
colnames(dataset) <- c("age", "sex", "cp", "trestbps", "chol", "fbs",
                       "restecg", "thalach", "exang", "oldpeak", "slope", "ca",
                       "thal", "hd")
head(dataset)
```

```
##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope  ca thal hd
## 1  63   1  1      145  233   1       2     150     0     2.3     3 0.0  6.0  0
## 2  67   1  4      160  286   0       2     108     1     1.5     2 3.0  3.0  2
## 3  67   1  4      120  229   0       2     129     1     2.6     2 2.0  7.0  1
## 4  37   1  3      130  250   0       0     187     0     3.5     3 0.0  3.0  0
## 5  41   0  2      130  204   0       2     172     0     1.4     1 0.0  3.0  0
## 6  56   1  2      120  236   0       0     178     0     0.8     1 0.0  3.0  0
```

```
str(dataset)
```

```
## 'data.frame':    303 obs. of  14 variables:
##  $ age     : num  63 67 67 37 41 56 62 57 63 53 ...
##  $ sex     : num  1 1 1 1 0 1 0 0 1 1 ...
##  $ cp      : num  1 4 4 3 2 2 4 4 4 4 ...
##  $ trestbps: num  145 160 120 130 130 120 140 120 130 140 ...
##  $ chol    : num  233 286 229 250 204 236 268 354 254 203 ...
##  $ fbs     : num  1 0 0 0 0 0 0 0 0 1 ...
##  $ restecg : num  2 2 2 0 2 0 2 0 2 2 ...
##  $ thalach : num  150 108 129 187 172 178 160 163 147 155 ...
##  $ exang   : num  0 1 1 0 0 0 0 1 0 1 ...
##  $ oldpeak : num  2.3 1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 ...
##  $ slope   : num  3 2 2 3 1 1 3 1 2 3 ...
##  $ ca      : chr  "0.0" "3.0" "2.0" "0.0" ...
```

```
##  $ thal    : chr  "6.0" "3.0" "7.0" "3.0" ...
##  $ hd      : int  0 2 1 0 0 0 3 0 2 1 ...
```

Now, we have column names in the dataset. But when we run str() function, we see that some of the columns are messed up. First, sex is a number in the dataset, but it is supposes to be a factor, where 0 represents "female" and 1 represents "male". cp (Chest Pain) is also supposed to be a factor where levels 1-3 represents different types of pain and 4 represents no chest pain. So let' clean the data before we start to use it.

```
dataset[dataset$sex == 0,]$sex <- "F"
dataset[dataset$sex == 1,]$sex <- "M"

# Now we convert the column into a factor
dataset$sex <- as.factor(dataset$sex)

# Convert a bunch of other columns into factors
dataset$cp <- as.factor(dataset$cp)
dataset$fbs <- as.factor(dataset$fbs)
dataset$restecg <- as.factor(dataset$restecg)
dataset$exang <- as.factor(dataset$exang)
dataset$slope <- as.factor(dataset$slope)
```

Since the ca column is a column of strings (chars), as seen when we ran str(dataset). R things that is a column of char, but instead it has values of integers. We correct that assumption by telling R that it is a column of integers.

```
dataset$ca <- as.integer(dataset$ca)
```

```
## Warning: NAs introduced by coercion
```

```
dataset$ca <- as.factor(dataset$ca)

# "Thal" column needs the similar correction
dataset$thal <- as.integer(dataset$thal)
```

```
## Warning: NAs introduced by coercion
```

```
dataset$thal <- as.factor(dataset$thal)
```

The last thing that we need to do is make *hd* (Heart Disease), a factor that is easy on the eyes. Here, we are going to use a fancy trick with ifelse() to convert the 0's to "Healthy" and 1's to "Unhealthy".

```
dataset$hd <- ifelse(test = dataset$hd == 0, yes = "Healthy", no="Unhealthy")
dataset$hd <- as.factor(dataset$hd)
```

Let us check if we have made all the neccessary changes to the data

```
str(dataset)
```

```
## 'data.frame':    303 obs. of  14 variables:
##  $ age     : num  63 67 67 37 41 56 62 57 63 53 ...
##  $ sex     : Factor w/ 2 levels "F","M": 2 2 2 2 1 2 1 1 2 2 ...
##  $ cp      : Factor w/ 4 levels "1","2","3","4": 1 4 4 3 2 2 4 4 4 4 ...
##  $ trestbps: num  145 160 120 130 130 120 140 120 130 140 ...
##  $ chol    : num  233 286 229 250 204 236 268 354 254 203 ...
##  $ fbs     : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 2 ...
##  $ restecg : Factor w/ 3 levels "0","1","2": 3 3 3 1 3 1 3 1 3 3 ...
##  $ thalach : num  150 108 129 187 172 178 160 163 147 155 ...
##  $ exang   : Factor w/ 2 levels "0","1": 1 2 2 1 1 1 1 2 1 2 ...
##  $ oldpeak : num  2.3 1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 ...
```

```
##  $ slope   : Factor w/ 3 levels "1","2","3": 3 2 2 3 1 1 3 1 2 3 ...
##  $ ca      : Factor w/ 4 levels "0","1","2","3": 1 4 3 1 1 1 3 1 2 1 ...
##  $ thal    : Factor w/ 3 levels "3","6","7": 2 1 3 1 1 1 1 1 3 3 ...
##  $ hd      : Factor w/ 2 levels "Healthy","Unhealthy": 1 2 2 1 1 1 2 1 2 2 ...
```

It looks like we have made the appropriate changes and are ready to proceed further.

Now we see how many samples (rows of data) have NA values. Later we will decide if we can just toss these samples out, or if we should impute values for the NAs.

```
nrow(dataset[is.na(dataset$ca) | is.na(dataset$thal), ])
```

```
## [1] 6
```

We get that there are 6 samples that have NAs in them. Let us view the samples with NAs by selecting those rows from the dataframe

```
dataset[is.na(dataset$ca) | is.na(dataset$thal), ]
```

```
##     age sex cp trestbps chol fbs restecg thalach exang oldpeak slope   ca thal
## 88   53   F  3      128  216   0       2     115     0     0.0     1    0 <NA>
## 167  52   M  3      138  223   0       0     169     0     0.0     1 <NA>    3
## 193  43   M  4      132  247   1       2     143     1     0.1     2 <NA>    7
## 267  52   M  4      128  204   1       0     156     1     1.0     2    0 <NA>
## 288  58   M  2      125  220   0       0     144     0     0.4     2 <NA>    7
## 303  38   M  3      138  175   0       0     173     0     0.0     1 <NA>    3
##            hd
## 88    Healthy
## 167   Healthy
## 193 Unhealthy
## 267 Unhealthy
## 288   Healthy
## 303   Healthy
```

Since there are only 6 samples, we will go ahead and remove them from the dataset and proceed further.

```
dataset <- dataset[!(is.na(dataset$ca) | is.na(dataset$thal) ), ]
nrow(dataset)
```

```
## [1] 297
```

Now, let us make sure that healthy and diseased samples come from each gender (male and female). If only male samples have heart diseases, we should probably remove all females from the model. We can do this with the xtabs() function. Since we want a table with heart diseases and sex, so we pass those two columns into the function.

```
xtabs(~ hd + sex, data = dataset)
```

```
##            sex
## hd            F   M
##   Healthy    71  89
##   Unhealthy  25 112
```

Healthy and Unhealthy patients are both represented by a lot of female and male samples.

Now, let us verify that all 4 levels of Chest pain (cp) were reported by a bunch of patients.

```
xtabs(~ hd + cp, data = dataset)
```

```
##            cp
## hd            1   2   3   4
```

```
##   Healthy    16  40  65  39
##   Unhealthy   7   9  18 103
```

We do this for all of the boolean and categorical variables that we are using to predict heart diseases.

```
xtabs(~ hd + fbs, data = dataset)
```

```
##            fbs
## hd            0   1
##   Healthy   137  23
##   Unhealthy 117  20
```

```
xtabs(~ hd + restecg, data = dataset)
```

```
##            restecg
## hd           0  1  2
##   Healthy   92  1 67
##   Unhealthy 55  3 79
```

We find here