Coronary Heart Disease in South Africa

Lavanya N

2/24/2021

Coronary Heart Disease in South Africa

1. Loading the Dataset

The dataset is a retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. There are roughly two controls per case of CHD. Many of the CHD positive men have undergone blood pressure reduction treatment and other programs to reduce their risk factors after their CHD event. In some cases the measurements were made after these treatments. These data are taken from a larger dataset, described in Rousseauw et al, 1983, South African Medical Journal.

```
SAheart <- read.csv("https://web.stanford.edu/~hastie/ElemStatLearn/datasets/SAheart.data",
                    header = TRUE)
SAheart$row.names <- NULL # removing the first column consisting of row indices as it is unnecessary
```

It is good practice use `str()`. There are 10 variables, 462 observations.

sbp systolic blood pressure tobacco cumulative tobacco (kg) ldl low density lipoprotein cholesterol adiposity waist-to-hip ratio famhist family history of heart disease (Present, Absent) typea type-A behavior obesity body mass index (BMI) alcohol current alcohol consumption age age at onset chd response, coronary heart disease

```
str(SAheart)
```

```
## 'data.frame':    462 obs. of  10 variables:
##  $ sbp      : int  160 144 118 170 134 132 142 114 114 132 ...
##  $ tobacco  : num  12 0.01 0.08 7.5 13.6 6.2 4.05 4.08 0 0 ...
##  $ ldl      : num  5.73 4.41 3.48 6.41 3.5 6.47 3.38 4.59 3.83 5.8 ...
##  $ adiposity: num  23.1 28.6 32.3 38 27.8 ...
##  $ famhist  : chr  "Present" "Absent" "Present" "Present" ...
##  $ typea    : int  49 55 52 51 60 62 59 62 49 69 ...
##  $ obesity  : num  25.3 28.9 29.1 32 26 ...
##  $ alcohol  : num  97.2 2.06 3.81 24.26 57.34 ...
##  $ age      : int  52 63 46 58 49 45 38 58 29 53 ...
##  $ chd      : int  1 1 0 1 1 0 0 1 0 1 ...
```

2. Predicting Coronary Heart Disease

R requires outcome variables in classification problems to be designated as factors. If this is not done, many supervised learning algorithms will treat them as continuous variables. While they are coded as factors in the step below, they can always be coerced to dummy variables later on, if necessary. This is done below.

While there are several algorithms that can be used for classification problems, the models that will be explored below are the linear probability model (LPM), logistic regression, and linear discriminant analysis (LDA).

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## Warning: package 'tibble' was built under R version 4.0.3

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
SAheart$chd <- factor(SAheart$chd, labels = c("yes", "no"), levels = 1:0) # designating outcome variabl
```

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------- tidymodels 0.1.2.9000 --

## v broom     0.7.2      v recipes   0.1.15
## v dials     0.0.9      v rsample   0.0.8
## v infer     0.5.3      v tune      0.1.2
## v modeldata 0.1.0      v workflows 0.2.1
## v parsnip   0.1.4      v yardstick 0.0.7

## Warning: package 'broom' was built under R version 4.0.3

## Warning: package 'dials' was built under R version 4.0.3

## Warning: package 'infer' was built under R version 4.0.3

## Warning: package 'modeldata' was built under R version 4.0.3

## Warning: package 'parsnip' was built under R version 4.0.3

## Warning: package 'recipes' was built under R version 4.0.3

## Warning: package 'rsample' was built under R version 4.0.3

## Warning: package 'tune' was built under R version 4.0.3

## Warning: package 'workflows' was built under R version 4.0.3
```

```
## Warning: package 'yardstick' was built under R version 4.0.3
```

```
## -- Conflicts ---------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

```
set.seed(20210220)

heart_split <- initial_split(SAheart, prob = 0.75, strata = chd)
heart_train <- training(heart_split)
heart_test <- testing(heart_split)
```

a. Binary Classification via a Linear Probability Model

```
heart_recipe <- recipe(chd ~ ., data = heart_train) %>% prep
lm_model <- linear_reg() %>% set_engine("lm")
lm_wflow <- workflow() %>% add_model(lm_model) %>%
  add_formula(as.integer(chd == "yes") ~ .)
lm_fit <- fit(lm_wflow, data = bake(heart_recipe, new_data = NULL))
y_hat <- predict(lm_fit, new_data = bake(heart_recipe, new_data = heart_test))
```

A matrix of confusion can be used to evaluate classifications. This is created below by making a simple contingency table between the levels of classification and the outcomes in the testing data. The model classifies 4 observations of `chd` correctly as `yes`, and 21 observations of `chd` correctly as `no`. However, the model classifies 21 observations of `chd` incorrectly as `yes`, and 4 observations of `chd` incorrectly as `no`.

```
y_hat <- mutate(y_hat, z = factor(.pred > 0.5, levels = c(TRUE, FALSE), labels = c("yes", "no")))
table(y_hat$z, heart_test$chd)
```

```
##
##       yes no
##   yes  20  8
##   no   20 67
```

Another method for evaluating classifications is to compute the accuracy of classifications. This simplifies the task of comparing different models. In this case, the accuracy of the model is 0.783.

```
(table(y_hat$z, heart_test$chd)[1] + table(y_hat$z, heart_test$chd)[4])/(table(y_hat$z, heart_test$chd)
```

```
## [1] 0.7565217
```

b. Binary Classification via Logistic Regression

The model classifies 20 observations of `chd` correctly as `yes`, and 71 observations of `chd` correctly as `no`. However, the model classifies 4 observations of `chd` incorrectly as `yes`, and 20 observations of `chd` incorrectly as `no`.

```
logit_model <- logistic_reg() %>% set_engine("glm")
logit_wflow <- workflow() %>% add_model(logit_model) %>% add_formula(chd ~ .)
logit_fit <- fit(logit_wflow, data = bake(heart_recipe, new_data = NULL))
bind_cols(chd = heart_test$chd,
          predict(logit_fit, new_data = bake(heart_recipe, new_data = heart_test))) %>%
  conf_mat(truth = chd, estimate = .pred_class)
```

```
##           Truth
## Prediction yes no
##        yes  21  8
##        no   19 67
```

The accuracy of the model is 0.791.

```
bind_cols(chd = heart_test$chd,
          predict(logit_fit, new_data = bake(heart_recipe, new_data = heart_test))) %>%
  accuracy(truth = chd, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.765
```

c. Classification via Logistic Regression with Penalisation

```
if (.Platform$OS.type == "windows") {
  doParallel::registerDoParallel()
} else doMC::registerDoMC(parallel::detectCores())
```

```
logit_recipe <-
  recipe(chd ~ ., data = heart_train) %>% step_dummy(all_nominal()) %>%
  step_normalize(all_predictors()) %>% prep

heart_rs <- bootstraps(heart_train, times = 50)
tune_spec <- logistic_reg(penalty = tune(), mixture = tune()) %>% set_engine("glmnet")
wf <- workflow() %>% add_recipe(logit_recipe) %>%
  remove_formula() %>% add_formula(chd ~ (.)^2)
```

```
## Warning: The workflow has no formula preprocessor to remove.
```

```
my_grid <- grid_regular(penalty(), mixture(), levels = 10)
results <- tune_grid(wf %>% add_model(tune_spec),
                     resamples = heart_rs, grid = my_grid)
```

The model classifies 15 observations of `chd` correctly as `yes`, and 71 observations of `chd` correctly as `no`. However, the model classifies 4 observations of `chd` incorrectly as `yes`, and 25 observations of `chd` incorrectly as `no`.

```
most_accurate <- results %>% select_best("accuracy")
final <- finalize_workflow(wf %>% add_model(tune_spec), most_accurate)
final_fit <- fit(final, data = heart_train)
baked <- bake(logit_recipe, new_data = heart_test)
bind_cols(chd = heart_test$chd,
          predict(final_fit, new_data = heart_test)) %>%
  conf_mat(truth = chd, estimate = .pred_class)
```

```
##           Truth
## Prediction yes no
##        yes  19  6
##        no   21 69
```

The accuracy of the model is 0.748.

```
bind_cols(chd = heart_test$chd,
          predict(final_fit, new_data = heart_test)) %>%
  accuracy(truth = chd, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.765
```

The Receiver Operating Characteristic (ROC) curve was used in the cross validation process to choose the best value of the tuning parameters according to which yielded the highest Area Under Curve (AUC) in the held-out fold. This can be used to evaluate the classification of the outcome variable in the testing data.

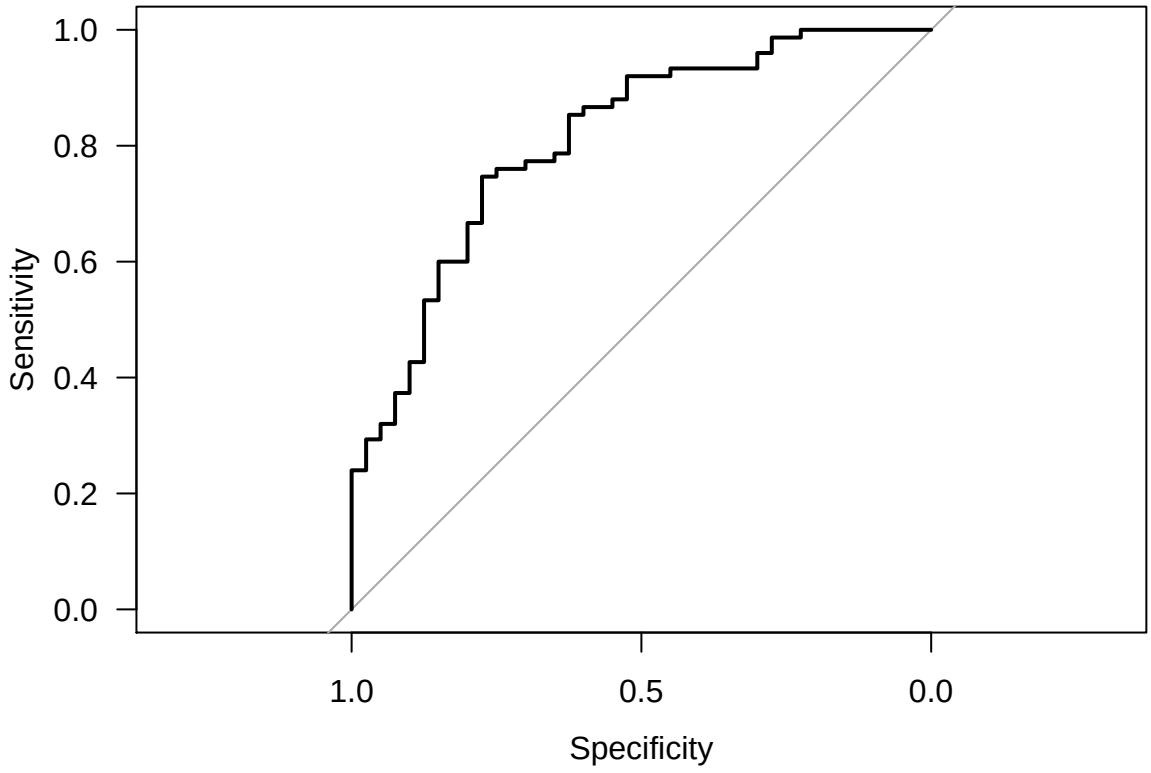
```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
penalized_logit_ROC <- roc(heart_test$chd,
                           predict(final_fit, new_data = heart_test, type = "prob")$.pred_yes,
                           levels = c("yes", "no"))
```

```
## Setting direction: controls > cases
```

```
plot(penalized_logit_ROC, las = 1)
```



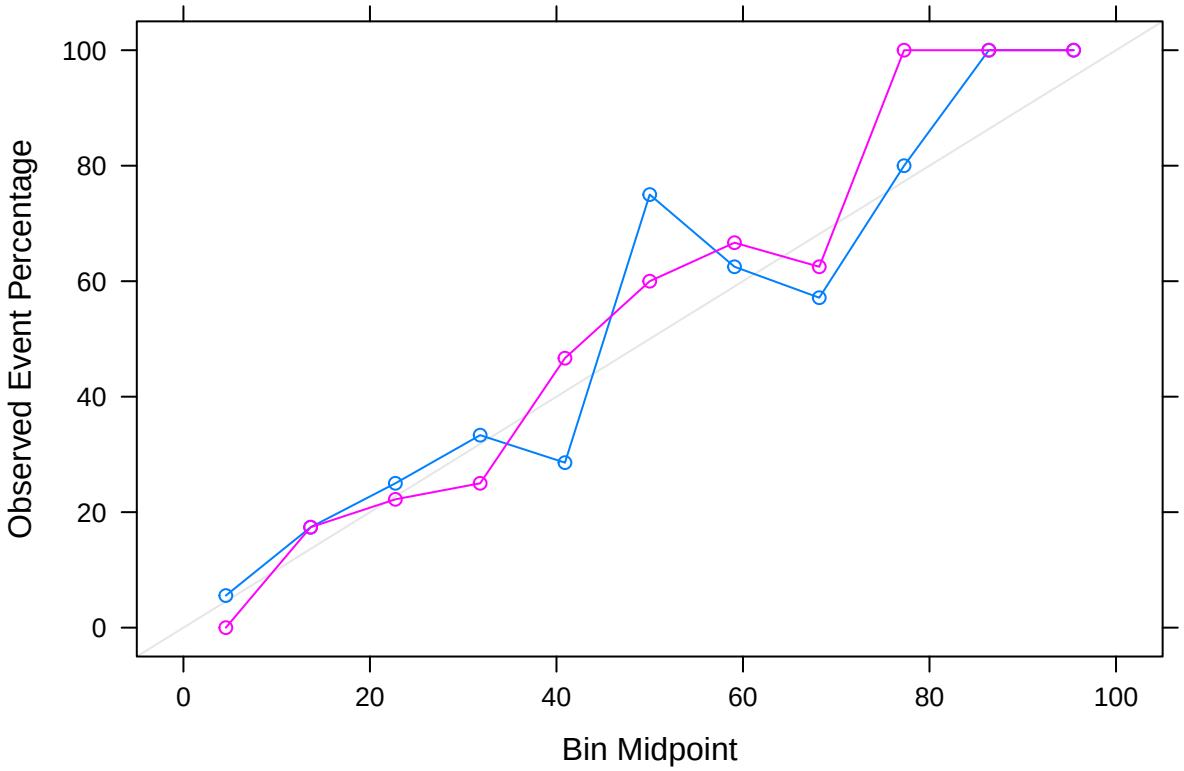
The area under the curve is 0.828.

```
auc(penalized_logit_ROC)
```

```
## Area under the curve: 0.8103
```

Calibration plots can be used to judge how well models are calibrated. If they are well-calibrated, the lines should lie close to the 45-degree line. In this case, the lines representing the logistic regression models without and with penalisation do not lie close. The purple line represents the model without penalisation, and the blue line represents the model with penalisation.

```
cal <- bind_cols(chd = heart_test,
                 glm = predict(logit_fit, new_data = heart_test, type = "prob")$.pred_yes,
                 glmnet = predict(final_fit, new_data = heart_test, type = "prob")$.pred_yes)
cc <- caret::calibration(chd ~ glm  + glmnet, data = cal)
plot(cc)
```



d. Classification via Linear Discriminant Analysis

The model classifies 20 observations of `chd` correctly as `yes`, and 69 observations of `chd` correctly as `no`. However, the model classifies 6 observations of `chd` incorrectly as `yes`, and 20 observations of `chd` incorrectly as `no`.

```
library(discrim)
```

```
## Warning: package 'discrim' was built under R version 4.0.3
```

```
##
## Attaching package: 'discrim'
```

```
## The following object is masked from 'package:dials':
##
##     smoothness
```

```
lda_model <- discrim_linear() %>% set_engine("MASS")
lda_fit <- lda_model %>% fit(chd ~ ., data = heart_train)
bind_cols(chd = heart_test$chd,
          predict(lda_fit, new_data = heart_test)) %>%
  conf_mat(truth = chd, estimate = .pred_class)
```

```
##           Truth
## Prediction yes no
##        yes  22  8
##        no   18 67
```

The accuracy of the model is 0.774.

```
bind_cols(chd = heart_test$chd,
          predict(lda_fit, new_data = heart_test)) %>%
  accuracy(truth = chd, estimate = .pred_class)
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.774
```

3. Selecting the Best Model

Based on accuracy of classification in the testing data, the logistic regression model (accuracy = 0.791) is the most suitable for predicting occurrence of coronary heart disease given this set of regressors.