Assignment5

Holly Finertie - HF2379

Due: 2/25/2020

Goal: You want to predict current alcohol consumption but it is expensive and time-consuming to administer all of the behavioral testing that produces the personality scores. You will conduct a reproducible analysis to build and test classification models using regularized logistic regression and traditional logistic regression.

### Data Import: Cleaning and Training/Testing Data Set

I imported the data, cleaned the variable names, and converted our outcome of interest to a factor variable named ‘alc\_outcome’. Then I created training and testing data sets with a 70/30 split.

alc = read\_csv("./data/alcohol\_use.csv") %>%   
 janitor::clean\_names() %>%   
 mutate(  
 alc\_outcome = case\_when(  
 alc\_consumption == "CurrentUse" ~ 1,   
 alc\_consumption == "NotCurrentUse" ~ 0),   
 alc\_outcome = as.factor(alc\_outcome)) %>%   
 select(-alc\_consumption)

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:  
## cols(  
## X1 = col\_double(),  
## neurotocism\_score = col\_double(),  
## extroversion\_score = col\_double(),  
## openness\_score = col\_double(),  
## agreeableness\_score = col\_double(),  
## conscientiousness\_score = col\_double(),  
## impulsiveness\_score = col\_double(),  
## sens\_seeking\_score = col\_double(),  
## alc\_consumption = col\_character()  
## )

head(alc)

## # A tibble: 6 x 9  
## x1 neurotocism\_sco… extroversion\_sc… openness\_score agreeableness\_s…  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 759 1.72 0.322 -1.12 -0.453   
## 2 898 1.84 -0.948 -0.976 -0.917   
## 3 1845 1.60 -0.948 1.24 -0.917   
## 4 1858 1.72 0.322 1.66 -0.155   
## 5 28 0.521 -1.23 -0.0193 -0.0173  
## 6 109 -0.467 2.13 0.141 0.131   
## # … with 4 more variables: conscientiousness\_score <dbl>,  
## # impulsiveness\_score <dbl>, sens\_seeking\_score <dbl>, alc\_outcome <fct>

train\_alc = alc %>% sample\_frac(.7)  
test\_alc = anti\_join(alc, train\_alc, by = 'x1')

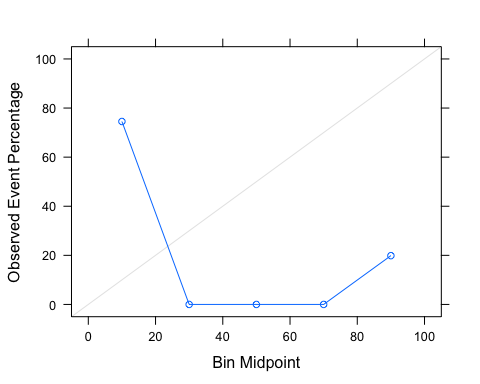
### 1. Create and Compare Models

#### Model 1: Use Caret package to choose alpha and lambda

# Apply to Model1  
model1 = train(  
 alc\_outcome ~.,   
 data = train\_alc,  
 method = "glmnet",  
 family = "binomial",  
 trControl = trainControl("cv", number = 10)  
 )  
  
# Test performance  
results1 = predict(model1,   
 test\_alc,   
 type = 'prob')  
  
results\_prob1 = ifelse(results1 > 0.5,1,0)  
  
outcome1 = (as.numeric(test\_alc$alc\_outcome) - 1)  
  
testProbs1 = data.frame(obs = test\_alc$alc\_outcome,  
 pred.logit = results\_prob1)  
  
missclass1 = mean(  
 results\_prob1 != outcome1,   
 na.rm = T)  
  
print(paste('Accuracy Model 1',1 - missclass1))

## [1] "Accuracy Model 1 0.5"

calPlotData1 = calibration(  
 obs ~ pred.logit.1,   
 data = testProbs1,   
 cuts = 5)  
  
xyplot(calPlotData1,   
 auto.key = list(columns = 2))

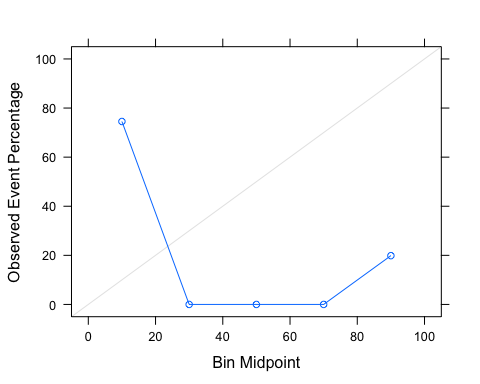


#### Model 2: Logistic Regression

# Apply to Model1  
model2 = glm(  
 alc\_outcome ~.,   
 family = binomial(link = 'logit'),  
 data = train\_alc  
 )  
  
# Test performance  
results2 = predict(model2,   
 test\_alc,   
 type = 'response')  
  
results\_prob2 = ifelse(results2 > 0.5,1,0)  
  
outcome2 = (as.numeric(test\_alc$alc\_outcome) - 1)  
  
testProbs2 = data.frame(obs = test\_alc$alc\_outcome,  
 pred.logit = results\_prob2)  
  
missclass2 = mean(  
 results\_prob2 != outcome2,   
 na.rm = T)  
  
print(paste('Accuracy Model 2',1 - missclass2))

## [1] "Accuracy Model 2 0.775221238938053"

calPlotData2 = calibration(  
 obs ~ pred.logit,   
 data = testProbs2,   
 cuts = 5)  
  
xyplot(calPlotData2,   
 auto.key = list(columns = 2))

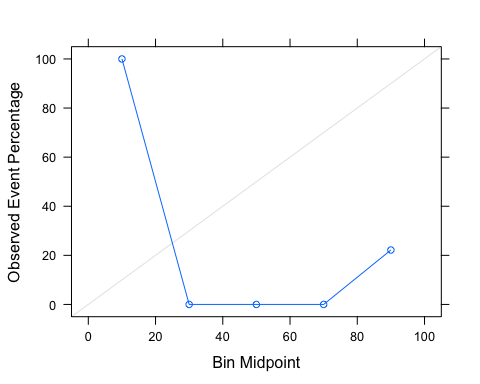


#### Model 3: LASSO using CARET package

# Apply to Model3  
lambda = 10^seq(-3,3, length = 100)  
  
model3 = train(  
 alc\_outcome ~.,   
 data = train\_alc,  
 method = "glmnet",  
 family = "binomial",  
 trControl = trainControl("cv", number = 10),   
 tuneGrid = expand.grid(alpha = 1, lambda = lambda))  
  
# Test performance  
results3 = predict(model3,   
 test\_alc,   
 type = 'prob')  
  
results\_prob3 = ifelse(results3 > 0.5,1,0)  
  
outcome3 = (as.numeric(test\_alc$alc\_outcome) - 1)  
  
testProbs3 = data.frame(obs = test\_alc$alc\_outcome,  
 pred.logit = results\_prob3)  
  
missclass3 = mean(  
 results\_prob3 != outcome3,   
 na.rm = T)  
  
print(paste('Accuracy Model 3',1 - missclass3))

## [1] "Accuracy Model 3 0.5"

calPlotData3 = calibration(  
 obs ~ pred.logit.1,   
 data = testProbs3,   
 cuts = 5)  
  
xyplot(calPlotData3,   
 auto.key = list(columns = 2))



### 2. Compare Performace of Each Model

Accuracy of each model:

performance =   
 tibble(  
 "Model 1" = c(1 - missclass1),   
 "Model 2" = c(1 - missclass2),   
 "Model 3" = c(1 - missclass3)  
 ) %>% knitr::kable()  
  
performance

|  |  |  |
| --- | --- | --- |
| Model 1 | Model 2 | Model 3 |
| 0.5 | 0.7752212 | 0.5 |

Using the above output, I would choose Model 2 which used standard logistic regression to predict.