ml\_hw5

Mohammad

2023-02-17

For the purpose of this exercise, the data has been subset to include only 7 features on personality traits and the variable which distinguishes those who reported current alcohol use (defined as alcohol use in the past month or more frequently) vs no current use. Data are stored in the csv file alcohol\_use.csv on the course site.

Feature Information: Below is a list of the 7 features and outcome variable within the dataset. In general, the higher value of the score, the greater the personality trait observed within the individual based on the questionnaire.

* alc\_consumption: CurrentUse, NotCurrentUse
* neurotocism\_score: Measure of Neuroticism
* extroversion\_score: Measure of Extroversion
* openness\_score: Measure of Openness to Experiences
* agreeableness\_score: Measure of Agreeableness
* conscientiousness\_score: Measure of Conscientiousness
* impulsiveness\_score: Measure of Impulsivity
* sens\_seeking\_score: Measure of Sensation-Seeking Behaviors.

### Goal

Predict current alcohol consumption but it is expensive and time-consuming to administer all of the behavioral testing that produces the personality scores.

# Preprocessing

## Tidying

In this step we read-in the data and convert the outcome alc\_consumption to a factor variable.

au <-  
 read\_csv("data/alcohol\_use.csv") %>%   
 mutate(alc\_consumption = factor(alc\_consumption)) %>%   
 select(- ...1)

## New names:  
## Rows: 1885 Columns: 9  
## ── Column specification  
## ──────────────────────────────────────────────────────── Delimiter: "," chr  
## (1): alc\_consumption dbl (8): ...1, neurotocism\_score, extroversion\_score,  
## openness\_score, agreea...  
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ  
## Specify the column types or set `show\_col\_types = FALSE` to quiet this message.  
## • `` -> `...1`

## Scaling and partioning

Here we standardize the data by scaling it then we split it into 70/30 training and testing datasets for further model fit.

set.seed(123)  
   
train.index <- createDataPartition(au$alc\_consumption, p = 0.7, list = FALSE)  
  
au\_train <- au[train.index, ]  
au\_test <- au[-train.index, ]  
  
#tidyverse way to create data partition  
#au\_train <- au$alc\_consumption %>% createDataPartition(p = 0.7, list = F)

## Comparing models

Next we conduct a reproducible analysis to build and test classification models using regularized logistic regression and traditional logistic regression.

### Elastic Net

A model that chooses alpha and lambda via cross-validation using all of the features. This will be an elastic net model. The best tune for this model is alpha = 0.7 and a lambda = 0.2578. The accuracy of this model is 0.8515.

set.seed(123)  
  
enet <-   
 train(alc\_consumption ~., data = au\_train, method = "glmnet",   
 trControl = trainControl("cv", number = 10),   
 preProc = c("center", "scale"), tuneLength = 10)  
  
#Print the values of alpha and lambda that gave best prediction  
enet$bestTune

## alpha lambda  
## 63 0.7 0.2578427

#Print all of the options examined  
enet$results

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 0.1 0.0003178806 0.7970334 0.5922755 0.02965711 0.05908586  
## 2 0.1 0.0007343454 0.7970334 0.5922755 0.02965711 0.05908586  
## 3 0.1 0.0016964331 0.7955240 0.5892831 0.03048477 0.06045782  
## 4 0.1 0.0039189805 0.7940260 0.5860820 0.03329813 0.06640520  
## 5 0.1 0.0090533531 0.7955526 0.5890117 0.03581935 0.07136193  
## 6 0.1 0.0209144200 0.7993176 0.5963373 0.03252125 0.06511704  
## 7 0.1 0.0483150228 0.7932512 0.5839484 0.03152748 0.06336552  
## 8 0.1 0.1116139691 0.7955182 0.5877890 0.03090766 0.06183682  
## 9 0.1 0.2578427450 0.7985486 0.5931573 0.02797842 0.05632717  
## 10 0.2 0.0003178806 0.7970334 0.5922755 0.02965711 0.05908586  
## 11 0.2 0.0007343454 0.7970334 0.5922755 0.02965711 0.05908586  
## 12 0.2 0.0016964331 0.7962816 0.5908274 0.03119624 0.06199068  
## 13 0.2 0.0039189805 0.7940260 0.5860820 0.03329813 0.06640520  
## 14 0.2 0.0090533531 0.7947951 0.5875179 0.03536480 0.07049000  
## 15 0.2 0.0209144200 0.7978025 0.5931851 0.03087561 0.06194862  
## 16 0.2 0.0483150228 0.7932626 0.5839188 0.03505606 0.07044360  
## 17 0.2 0.1116139691 0.7970391 0.5910252 0.03125055 0.06251593  
## 18 0.2 0.2578427450 0.8068824 0.6098077 0.02734268 0.05491528  
## 19 0.3 0.0003178806 0.7970334 0.5922755 0.02965711 0.05908586  
## 20 0.3 0.0007343454 0.7970334 0.5922755 0.02965711 0.05908586  
## 21 0.3 0.0016964331 0.7962816 0.5908274 0.03119624 0.06199068  
## 22 0.3 0.0039189805 0.7955355 0.5892145 0.03141828 0.06242366  
## 23 0.3 0.0090533531 0.7963103 0.5906471 0.03410132 0.06795373  
## 24 0.3 0.0209144200 0.7970449 0.5916829 0.02912407 0.05853813  
## 25 0.3 0.0483150228 0.7932511 0.5839972 0.03308696 0.06623580  
## 26 0.3 0.1116139691 0.7993121 0.5955828 0.02607438 0.05196487  
## 27 0.3 0.2578427450 0.8046039 0.6052494 0.02820461 0.05603620  
## 28 0.4 0.0003178806 0.7962816 0.5908274 0.03119624 0.06199068  
## 29 0.4 0.0007343454 0.7962816 0.5908274 0.03119624 0.06199068  
## 30 0.4 0.0016964331 0.7962816 0.5908274 0.03119624 0.06199068  
## 31 0.4 0.0039189805 0.7955355 0.5892145 0.03141828 0.06242366  
## 32 0.4 0.0090533531 0.7970622 0.5923645 0.03190289 0.06335897  
## 33 0.4 0.0209144200 0.7955412 0.5889347 0.03098414 0.06209828  
## 34 0.4 0.0483150228 0.7955181 0.5886655 0.03155523 0.06323441  
## 35 0.4 0.1116139691 0.7993178 0.5957966 0.02721298 0.05392959  
## 36 0.4 0.2578427450 0.8190153 0.6330834 0.03321156 0.06624385  
## 37 0.5 0.0003178806 0.7962816 0.5908274 0.03119624 0.06199068  
## 38 0.5 0.0007343454 0.7962816 0.5908274 0.03119624 0.06199068  
## 39 0.5 0.0016964331 0.7955297 0.5893822 0.03283554 0.06508187  
## 40 0.5 0.0039189805 0.7947779 0.5877288 0.03110428 0.06179737  
## 41 0.5 0.0090533531 0.7963046 0.5908815 0.03183529 0.06321393  
## 42 0.5 0.0209144200 0.7962931 0.5905766 0.02966742 0.05933984  
## 43 0.5 0.0483150228 0.7962816 0.5903371 0.03059261 0.06115143  
## 44 0.5 0.1116139691 0.7947549 0.5868572 0.02831284 0.05609767  
## 45 0.5 0.2578427450 0.8242727 0.6429997 0.02282130 0.04619018  
## 46 0.6 0.0003178806 0.7962816 0.5908274 0.03119624 0.06199068  
## 47 0.6 0.0007343454 0.7962816 0.5908274 0.03119624 0.06199068  
## 48 0.6 0.0016964331 0.7955297 0.5893822 0.03283554 0.06508187  
## 49 0.6 0.0039189805 0.7947779 0.5877288 0.03110428 0.06179737  
## 50 0.6 0.0090533531 0.7963046 0.5908815 0.03183529 0.06321393  
## 51 0.6 0.0209144200 0.7947722 0.5876089 0.03055479 0.06106083  
## 52 0.6 0.0483150228 0.7940032 0.5857226 0.02837112 0.05652733  
## 53 0.6 0.1116139691 0.7947550 0.5869340 0.02498258 0.04929182  
## 54 0.6 0.2578427450 0.8500368 0.6929549 0.02898407 0.05981528  
## 55 0.7 0.0003178806 0.7962816 0.5908274 0.03119624 0.06199068  
## 56 0.7 0.0007343454 0.7962816 0.5908274 0.03119624 0.06199068  
## 57 0.7 0.0016964331 0.7955297 0.5893822 0.03283554 0.06508187  
## 58 0.7 0.0039189805 0.7932684 0.5848007 0.03256549 0.06452421  
## 59 0.7 0.0090533531 0.7970565 0.5924466 0.03094309 0.06127956  
## 60 0.7 0.0209144200 0.7947550 0.5875843 0.03071425 0.06129246  
## 61 0.7 0.0483150228 0.7939860 0.5858020 0.02852679 0.05658826  
## 62 0.7 0.1116139691 0.7947550 0.5868802 0.02523654 0.04963366  
## 63 0.7 0.2578427450 0.8515577 0.6958634 0.02859210 0.05921660  
## 64 0.8 0.0003178806 0.7962816 0.5908274 0.03119624 0.06199068  
## 65 0.8 0.0007343454 0.7962816 0.5908274 0.03119624 0.06199068  
## 66 0.8 0.0016964331 0.7955355 0.5893930 0.03260164 0.06457215  
## 67 0.8 0.0039189805 0.7940318 0.5863751 0.03286864 0.06515696  
## 68 0.8 0.0090533531 0.7955412 0.5895906 0.03333955 0.06595571  
## 69 0.8 0.0209144200 0.7932627 0.5847833 0.03596884 0.07163601  
## 70 0.8 0.0483150228 0.7879481 0.5742003 0.03090815 0.06104744  
## 71 0.8 0.1116139691 0.7788167 0.5565013 0.03393988 0.06660184  
## 72 0.8 0.2578427450 0.8515577 0.6958634 0.02859210 0.05921660  
## 73 0.9 0.0003178806 0.7962816 0.5908274 0.03119624 0.06199068  
## 74 0.9 0.0007343454 0.7962816 0.5908274 0.03119624 0.06199068  
## 75 0.9 0.0016964331 0.7955355 0.5893930 0.03260164 0.06457215  
## 76 0.9 0.0039189805 0.7947837 0.5879402 0.03206453 0.06340573  
## 77 0.9 0.0090533531 0.7947894 0.5881441 0.03559544 0.07040461  
## 78 0.9 0.0209144200 0.7909841 0.5802541 0.03459672 0.06879721  
## 79 0.9 0.0483150228 0.7818874 0.5625260 0.03245312 0.06360267  
## 80 0.9 0.1116139691 0.7682103 0.5367870 0.03529765 0.06886148  
## 81 0.9 0.2578427450 0.8515577 0.6958634 0.02859210 0.05921660  
## 82 1.0 0.0003178806 0.7955240 0.5893417 0.03090031 0.06140333  
## 83 1.0 0.0007343454 0.7955240 0.5893417 0.03090031 0.06140333  
## 84 1.0 0.0016964331 0.7955355 0.5893930 0.03260164 0.06457215  
## 85 1.0 0.0039189805 0.7947837 0.5880098 0.03361792 0.06657718  
## 86 1.0 0.0090533531 0.7940318 0.5867479 0.03511970 0.06939124  
## 87 1.0 0.0209144200 0.7902208 0.5788537 0.03366995 0.06686985  
## 88 1.0 0.0483150228 0.7720327 0.5436359 0.03521306 0.06893907  
## 89 1.0 0.1116139691 0.7697255 0.5399013 0.03688648 0.07211384  
## 90 1.0 0.2578427450 0.7053189 0.3836457 0.04673938 0.10045329

# Model coefficients  
coef(enet$finalModel, enet$bestTune$lambda)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) -0.1348075  
## neurotocism\_score .   
## extroversion\_score .   
## openness\_score .   
## agreeableness\_score .   
## conscientiousness\_score .   
## impulsiveness\_score -0.3681763  
## sens\_seeking\_score .

#Model performance  
confusionMatrix(enet)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction CurrentUse NotCurrentUse  
## CurrentUse 53.3 14.8  
## NotCurrentUse 0.0 31.9  
##   
## Accuracy (average) : 0.8515

### Traditional logisitc regression

A model that uses all the features and traditional logistic regression. The accuracy for this model is 0.7962

set.seed(123)   
  
glm <-  
 train(alc\_consumption ~., data = au\_train, method = "glm",   
 trControl = trainControl("cv", number = 10), family = "binomial",  
 preProc = c("center", "scale"))  
  
#Model performance  
confusionMatrix(glm)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction CurrentUse NotCurrentUse  
## CurrentUse 43.1 10.2  
## NotCurrentUse 10.2 36.5  
##   
## Accuracy (average) : 0.7962

#Additional measures, such as sensitivity, specificity, and AUC, can be requested by using the twoClassSummary function in trainControl with the argument classProbs = TRUE  
#summaryFunction = twoClassSummary, classProbs = TRUE

### LASSO

A lasso model using all of the features. The best tune for this model uses an alpha = 1 (default for lasso) and a lamda = 0.2310. The accuracy of this model is 0.8515

#Create grid to search lambda  
lambda <- 10^seq(-3, 3, length = 100)  
  
set.seed(123)  
  
lasso <-  
 train(alc\_consumption ~., data = au\_train, method = "glmnet",   
 trControl = trainControl("cv", number = 10),   
 preProc = c("center", "scale"), tuneGrid = expand.grid(alpha = 1, lambda = lambda))  
  
#Print the values of alpha and lambda that gave best prediction  
lasso$bestTune

## alpha lambda  
## 40 1 0.231013

#Print all options examined  
lasso$results

## alpha lambda Accuracy Kappa AccuracySD KappaSD  
## 1 1 1.000000e-03 0.7955240 0.5893417 0.030900309 0.06140333  
## 2 1 1.149757e-03 0.7955240 0.5893417 0.030900309 0.06140333  
## 3 1 1.321941e-03 0.7955240 0.5893417 0.030900309 0.06140333  
## 4 1 1.519911e-03 0.7955355 0.5893930 0.032601636 0.06457215  
## 5 1 1.747528e-03 0.7955355 0.5893930 0.032601636 0.06457215  
## 6 1 2.009233e-03 0.7955355 0.5893930 0.032601636 0.06457215  
## 7 1 2.310130e-03 0.7947779 0.5879100 0.032495974 0.06435381  
## 8 1 2.656088e-03 0.7947779 0.5879100 0.032495974 0.06435381  
## 9 1 3.053856e-03 0.7947779 0.5879100 0.032495974 0.06435381  
## 10 1 3.511192e-03 0.7955412 0.5894844 0.032760706 0.06490448  
## 11 1 4.037017e-03 0.7947837 0.5880098 0.033617916 0.06657718  
## 12 1 4.641589e-03 0.7947837 0.5880272 0.033236375 0.06570712  
## 13 1 5.336699e-03 0.7947837 0.5880272 0.033236375 0.06570712  
## 14 1 6.135907e-03 0.7940203 0.5865367 0.032955970 0.06514571  
## 15 1 7.054802e-03 0.7947837 0.5881077 0.033236375 0.06574692  
## 16 1 8.111308e-03 0.7925223 0.5837208 0.035864154 0.07091235  
## 17 1 9.326033e-03 0.7940261 0.5866285 0.033317123 0.06596552  
## 18 1 1.072267e-02 0.7955299 0.5895691 0.031091447 0.06145984  
## 19 1 1.232847e-02 0.7955241 0.5892757 0.030933126 0.06149104  
## 20 1 1.417474e-02 0.7902265 0.5787957 0.035815847 0.07116078  
## 21 1 1.629751e-02 0.7902266 0.5787736 0.033635954 0.06684807  
## 22 1 1.873817e-02 0.7909841 0.5803411 0.034039268 0.06762396  
## 23 1 2.154435e-02 0.7909784 0.5804184 0.034259649 0.06804337  
## 24 1 2.477076e-02 0.7894632 0.5774050 0.034559556 0.06846229  
## 25 1 2.848036e-02 0.7894689 0.5774562 0.036479143 0.07210445  
## 26 1 3.274549e-02 0.7879538 0.5746953 0.035105152 0.06902485  
## 27 1 3.764936e-02 0.7826506 0.5642324 0.035088189 0.06873761  
## 28 1 4.328761e-02 0.7780993 0.5556593 0.035298213 0.06882597  
## 29 1 4.977024e-02 0.7727903 0.5451968 0.034396212 0.06724710  
## 30 1 5.722368e-02 0.7705002 0.5408553 0.035460507 0.06935018  
## 31 1 6.579332e-02 0.7682160 0.5367254 0.037519110 0.07324395  
## 32 1 7.564633e-02 0.7674585 0.5352420 0.036284315 0.07082550  
## 33 1 8.697490e-02 0.7689679 0.5383456 0.036021230 0.07033575  
## 34 1 1.000000e-01 0.7697255 0.5399013 0.036886478 0.07211384  
## 35 1 1.149757e-01 0.7697255 0.5399013 0.036886478 0.07211384  
## 36 1 1.321941e-01 0.7697255 0.5399013 0.036886478 0.07211384  
## 37 1 1.519911e-01 0.7697255 0.5399013 0.036886478 0.07211384  
## 38 1 1.747528e-01 0.8515577 0.6958634 0.028592105 0.05921660  
## 39 1 2.009233e-01 0.8515577 0.6958634 0.028592105 0.05921660  
## 40 1 2.310130e-01 0.8515577 0.6958634 0.028592105 0.05921660  
## 41 1 2.656088e-01 0.6878947 0.3464001 0.016242714 0.03482724  
## 42 1 3.053856e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 43 1 3.511192e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 44 1 4.037017e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 45 1 4.641589e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 46 1 5.336699e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 47 1 6.135907e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 48 1 7.054802e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 49 1 8.111308e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 50 1 9.326033e-01 0.5325765 0.0000000 0.002654595 0.00000000  
## 51 1 1.072267e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 52 1 1.232847e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 53 1 1.417474e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 54 1 1.629751e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 55 1 1.873817e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 56 1 2.154435e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 57 1 2.477076e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 58 1 2.848036e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 59 1 3.274549e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 60 1 3.764936e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 61 1 4.328761e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 62 1 4.977024e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 63 1 5.722368e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 64 1 6.579332e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 65 1 7.564633e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 66 1 8.697490e+00 0.5325765 0.0000000 0.002654595 0.00000000  
## 67 1 1.000000e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 68 1 1.149757e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 69 1 1.321941e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 70 1 1.519911e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 71 1 1.747528e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 72 1 2.009233e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 73 1 2.310130e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 74 1 2.656088e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 75 1 3.053856e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 76 1 3.511192e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 77 1 4.037017e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 78 1 4.641589e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 79 1 5.336699e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 80 1 6.135907e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 81 1 7.054802e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 82 1 8.111308e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 83 1 9.326033e+01 0.5325765 0.0000000 0.002654595 0.00000000  
## 84 1 1.072267e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 85 1 1.232847e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 86 1 1.417474e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 87 1 1.629751e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 88 1 1.873817e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 89 1 2.154435e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 90 1 2.477076e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 91 1 2.848036e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 92 1 3.274549e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 93 1 3.764936e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 94 1 4.328761e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 95 1 4.977024e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 96 1 5.722368e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 97 1 6.579332e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 98 1 7.564633e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 99 1 8.697490e+02 0.5325765 0.0000000 0.002654595 0.00000000  
## 100 1 1.000000e+03 0.5325765 0.0000000 0.002654595 0.00000000

# Model coefficients  
coef(lasso$finalModel, lasso$bestTune$lambda)

## 8 x 1 sparse Matrix of class "dgCMatrix"  
## s1  
## (Intercept) -0.1329242  
## neurotocism\_score .   
## extroversion\_score .   
## openness\_score .   
## agreeableness\_score .   
## conscientiousness\_score .   
## impulsiveness\_score -0.2730990  
## sens\_seeking\_score .

#Model performance  
confusionMatrix(lasso)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction CurrentUse NotCurrentUse  
## CurrentUse 53.3 14.8  
## NotCurrentUse 0.0 31.9  
##   
## Accuracy (average) : 0.8515

## Model choice

The elastic net and lasso model have similar accuracy = 0.8515. However, the lambda for the LASSO model is slightly lower. Since our goal is to reduce cost and time spent in administering all of the behavioral testing, the LASSO model would likely be a better choice since it can shrink down features to zero thus help select fewer but more relevant features. However, in the context of this problem, the coefficients for all features are identical. A more targeted approach in tuning the hyperparameters in the elastic model or using a more robust metric to evaluate model performance such as area under the curve or AUC might help make the decision on model choice easier.

## Predictions

The final model, LASSO, is used to evaluate the performance using the testing dataset. The accuracy of the model is 0.8549 with is relatively high though nit ideal. The sensitivity of the model is 100% but the specificity 69%. 82 observations were misclassified as “CurrentUse” in the prediction.

pred <- lasso %>% predict(au\_test)  
  
# Model prediction performance  
confusionMatrix(pred, au\_test$alc\_consumption, positive = "CurrentUse")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction CurrentUse NotCurrentUse  
## CurrentUse 301 82  
## NotCurrentUse 0 182  
##   
## Accuracy : 0.8549   
## 95% CI : (0.8231, 0.8829)  
## No Information Rate : 0.5327   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7028   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 1.0000   
## Specificity : 0.6894   
## Pos Pred Value : 0.7859   
## Neg Pred Value : 1.0000   
## Prevalence : 0.5327   
## Detection Rate : 0.5327   
## Detection Prevalence : 0.6779   
## Balanced Accuracy : 0.8447   
##   
## 'Positive' Class : CurrentUse   
##

# Research applications

The data could be used to train models to directly predict the risk of alcohol use in adults with medical conditions that may be complicated by alcohol use, based on individual personality traits. This could consequently inform future clinical decisions for these patients. Indirectly, it could be used to predict risk of other drug use among adults based on their individual personality traits which could be used to inform the development of social and health education programs and properly directing resources to those at higher risk.