CYO Project - Liver Disease Prediction

Harry Terris

2019-05-07

Introduction

In this project, we develop and evaluate two machine learning models for predicting the presence of liver disease in patients based on age, gender and a range of diagnostic tests.

We use the Indian Liver Patient Dataset (http://archive.ics.uci.edu/ml/datasets/ILPD+%28Indian+Liver+Patient+Dataset%29) downloaded from the UCI Machine Learning Repository.[1]

The dataset comprises records on 582 patients, and “was collected from north east of Andhra Pradesh, India,” according to the UCI Machine Learning Repository website. The records contain information on age, gender, diagnostic test results, and whether the patient was a liver patient or not.

The main steps for this project are:

- Load the data, explore it and split it into training and test sets
- Train the machine learning algorithms
- Evaluate the algorithms using the test set

Methods and Analysis

First, we load the libraries used in this project.

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-proje
ct.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

Next, we download the dataset and feed it into a dataframe object.

```
dl <- tempfile()
download.file("http://archive.ics.uci.edu/ml/machine-learning-databases/00225/Indian%
20Liver%20Patient%20Dataset%20(ILPD).csv", dl)
dat <- read.csv(dl, col.names = c("age", "gender", "tBilirubin", "dBilirubin",
                                  "aAP", "sgptAA",
                                  "sgotAA", "tProtiens",
                                  "Albumin", "AlbuminGRatio", "Selector"))
rm(dl)
```

We check for missing values.

```
colSums(is.na(dat))
```

```
##           age        gender    tBilirubin    dBilirubin           aAP 
##             0             0             0             0             0 
##        sgptAA        sgotAA     tProtiens       Albumin AlbuminGRatio 
##             0             0             0             0             4 
##      Selector 
##             0
```

There are four missing values in the dataset, all for AlbuminGRatio. We remove these records.

```
dat <- dat[-which(is.na(dat$AlbuminGRatio)), ]
```

That leaves ***578*** records, with ***413*** for patients with liver disease and ***165*** for patients without liver disease. ***439*** are for male patients and ***139*** are for female patients.

The following table shows correlations among the diagnostic test results.

```
##               tBilirubin dBilirubin   aAP sgptAA sgotAA tProtiens Albumin
## tBilirubin          1.00       0.87  0.21   0.21   0.24     -0.01   -0.22
## dBilirubin          0.87       1.00  0.23   0.23   0.26      0.00   -0.23
## aAP                 0.21       0.23  1.00   0.12   0.17     -0.03   -0.16
## sgptAA              0.21       0.23  0.12   1.00   0.79     -0.04   -0.03
## sgotAA              0.24       0.26  0.17   0.79   1.00     -0.03   -0.08
## tProtiens          -0.01       0.00 -0.03  -0.04  -0.03      1.00    0.78
## Albumin            -0.22      -0.23 -0.16  -0.03  -0.08      0.78    1.00
## AlbuminGRatio      -0.21      -0.20 -0.23   0.00  -0.07      0.23    0.69
##               AlbuminGRatio
## tBilirubin            -0.21
## dBilirubin            -0.20
## aAP                   -0.23
## sgptAA                 0.00
## sgotAA                -0.07
## tProtiens              0.23
## Albumin                0.69
## AlbuminGRatio          1.00
```

We find that tBilirubin is highly correlated with dBilirubin; sgptAA is highly correlated with sgotAA; and tProtiens is highly correlated with Albumin. We remove the second feature from each pair as a preprocessing step. We also convert the field classifying patients by presence of liver disease into a Boolean variable.

```
dat <- dat %>% select("age", "gender", "tBilirubin", "aAP", "sgptAA", "tProtiens",
                      "AlbuminGRatio", "Selector") %>%
  mutate(Disease = as.factor(ifelse(Selector == 1, 0, 1))) %>% select(-Selector)
```

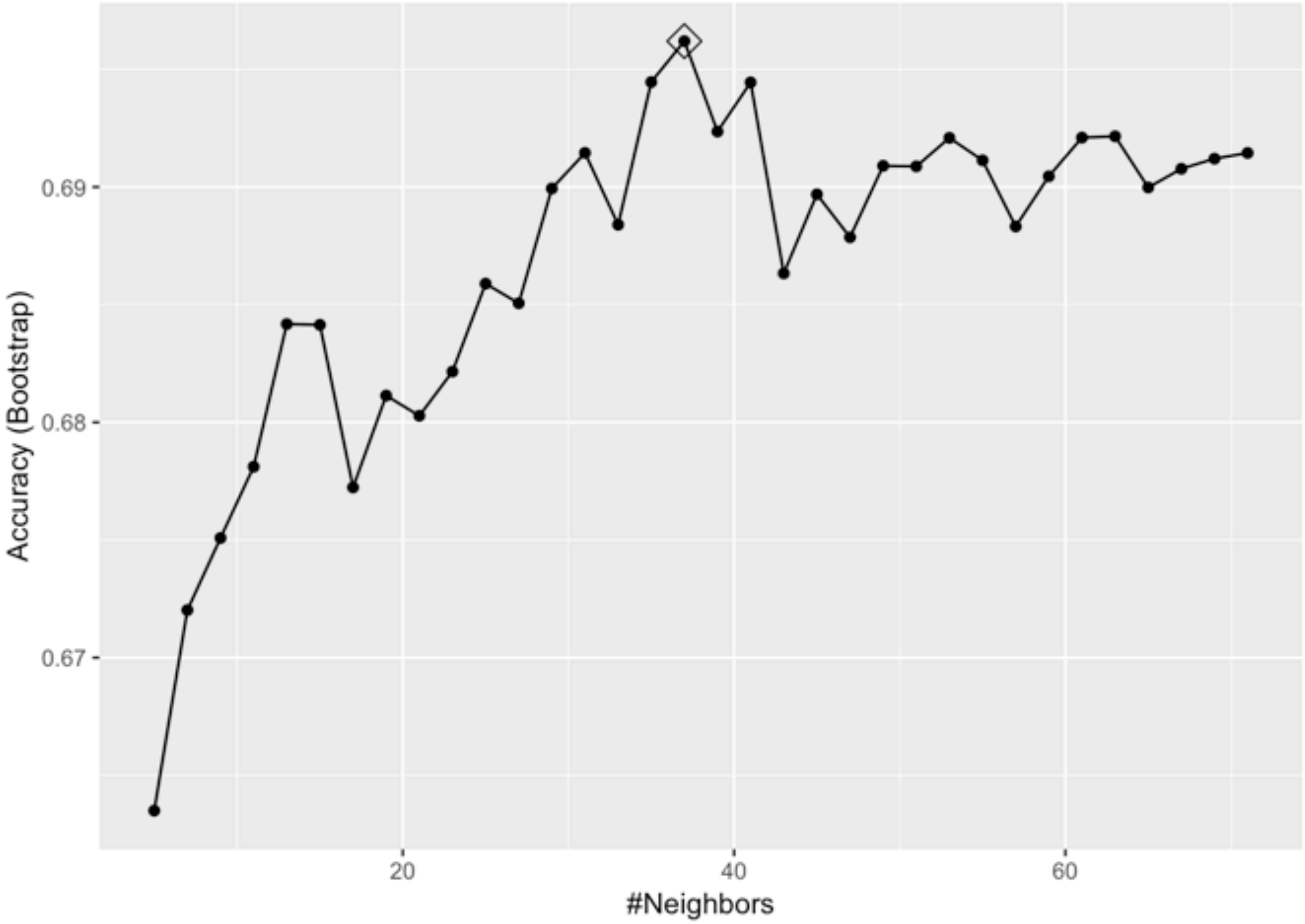
Now we create training and test sets.

```
set.seed(2019)
test_index <- createDataPartition(y = dat$Disease, times = 1, p = 0.2, list = FALSE)
train_set <- dat[-test_index, ]
test_set <- dat[test_index, ]
```

We are ready to train a k-nearest neighbors model. We use the train function in the caret package, performing cross validation with the default 25 bootstrap samples, each including 25% of the training set observations. We optimize for accuracy across 34 values for k ranging from 5 to 71.

```
train_knn <- train(Disease ~ ., method = "knn",
                   data = train_set,
                   tuneGrid = data.frame(k = seq(5, 71, 2)))
```

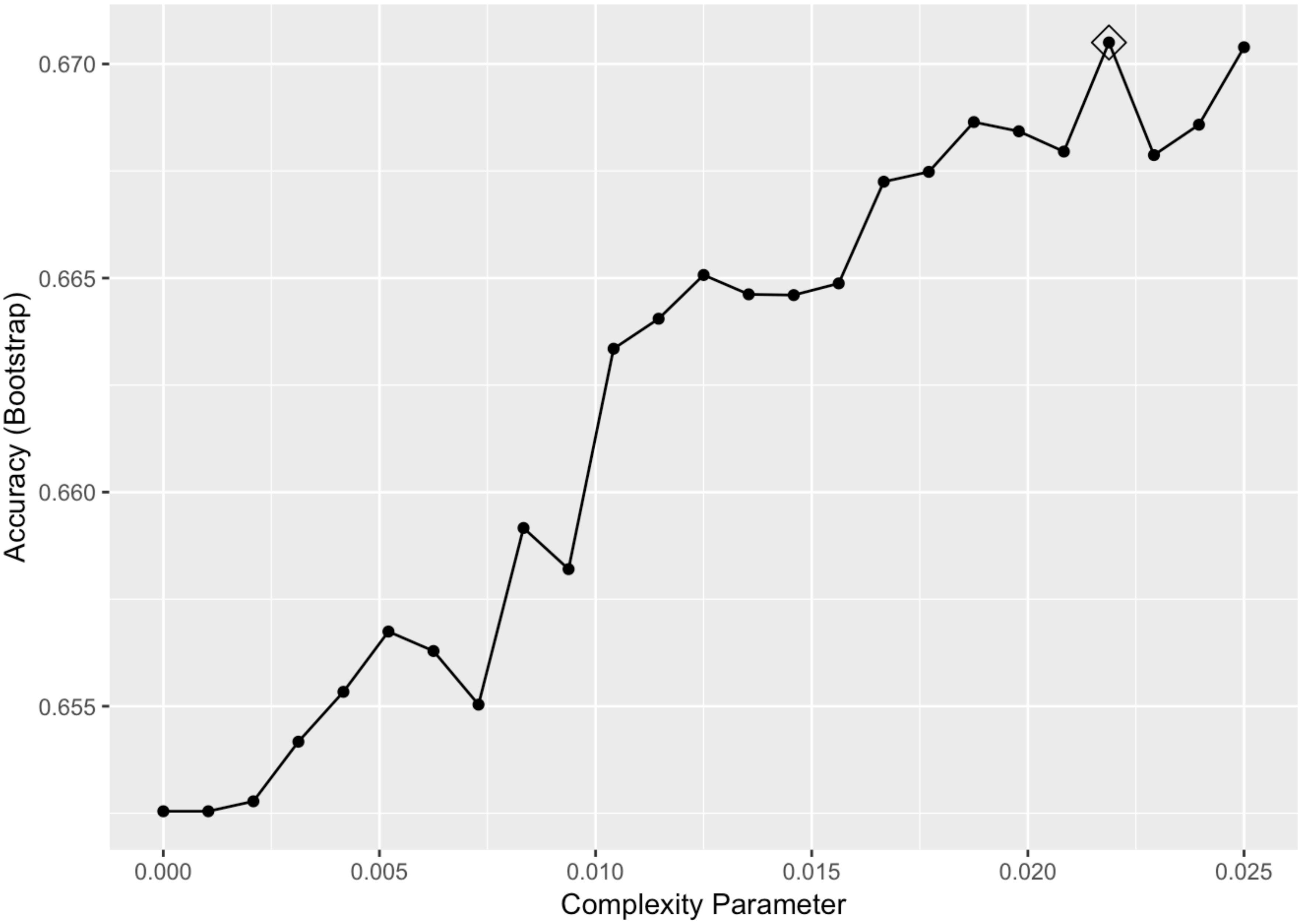
The following plot shows that k=***37*** optimizes for accuracy across the values considered. We note that that seems like a large neighborhood.



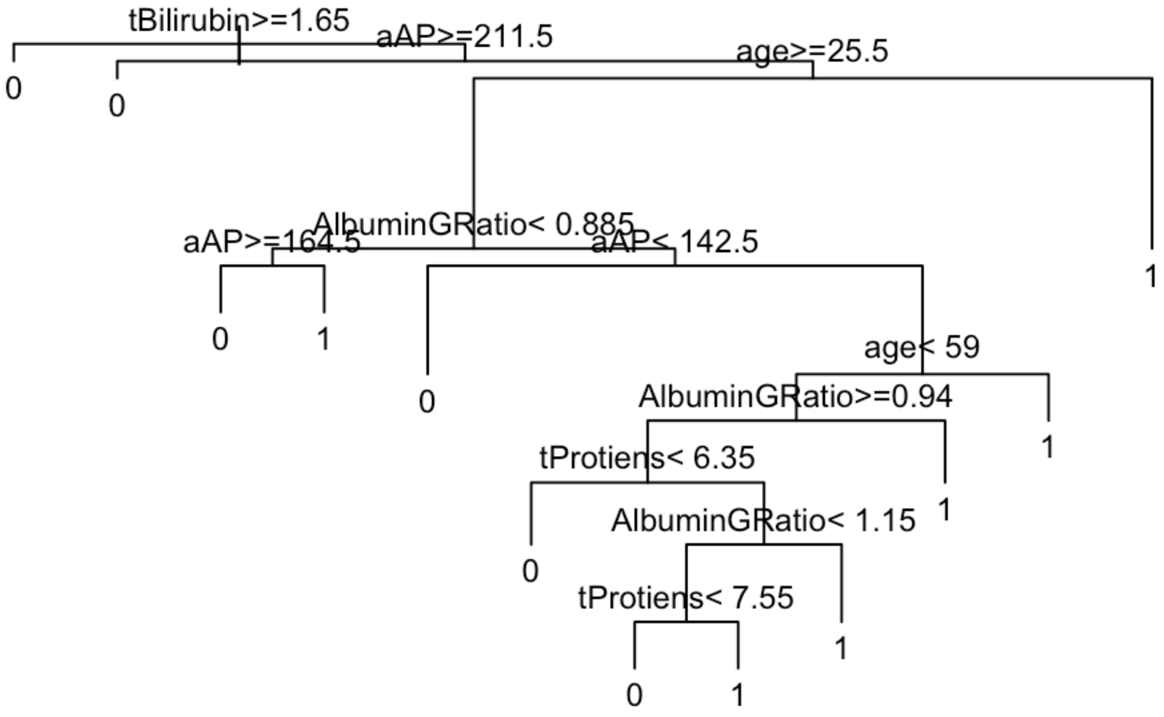
We also train a classification tree. Again, we use caret's default for cross validation. We optimize for accuracy across a range of values for the complexity parameter, or a minimum for how much the loss function must improve for another partition to be added.

```
train_rpart <- train(Disease ~ .,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0, 0.025, len = 25)),
                     data = train_set)
```

The following plot shows that cp=***0.022*** optimizes for accuracy across the values considered.



Here is an image of decision tree for the optimized complexity parameter. The partitions appear to be quite messy.



Results

Evaluating the k-nearest neighbors model against the test set shows moderate accuracy (***0.73***), the result of high sensitivity and low specificity and a high prevalance of patients in the data with liver disease. Here are the evaluation metrics produced by the confusionMatrix function.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 79 27
##          1  4  6
##
##                Accuracy : 0.7328
##                  95% CI : (0.6426, 0.8107)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.384
##
##                   Kappa : 0.1691
##  Mcnemar's Test P-Value : 7.772e-05
##
##             Sensitivity : 0.9518
##             Specificity : 0.1818
##          Pos Pred Value : 0.7453
##          Neg Pred Value : 0.6000
##              Prevalence : 0.7155
##          Detection Rate : 0.6810
##    Detection Prevalence : 0.9138
##       Balanced Accuracy : 0.5668
##
##        'Positive' Class : 0
##
```

The classification tree shows comparable accuracy (***0.72***), but with less of an imbalance between sensitivity and specificity.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 71 20
##          1 12 13
##
##                Accuracy : 0.7241
##                  95% CI : (0.6334, 0.803)
##     No Information Rate : 0.7155
##     P-Value [Acc > NIR] : 0.4649
##
##                   Kappa : 0.269
##  Mcnemar's Test P-Value : 0.2159
##
##             Sensitivity : 0.8554
##             Specificity : 0.3939
##          Pos Pred Value : 0.7802
##          Neg Pred Value : 0.5200
##              Prevalence : 0.7155
##          Detection Rate : 0.6121
##    Detection Prevalence : 0.7845
##       Balanced Accuracy : 0.6247
##
##        'Positive' Class : 0
##
```

Conclusion

The accuracy of both models generated and evaluated here was modest. Both also showed low specificity against the test set, with the k-nearest neighbors model performing especially poorly on this dimension.

Further experimentation is clearly warranted to develop a more robust model. The task could also be made easier with a larger dataset, and one that is more balanced between patients with liver disease and patients without liver disease.

[1] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml (http://archive.ics.uci.edu/ml)]. Irvine, CA: University of California, School of Information and Computer Science.↩