Classification: Predicting Heart Disease

Sujay Talanki

8/1/2022

This project uses several different classification methods to predict whether or not a patient has heart disease.

Loading the dataset:

```
heart <- read.csv("C:/Users/talan/Downloads/heart.csv")
```

I created a generic function below to calculate the average K-fold Cross validation accuracy (5 trials of K folds) of each model. This model takes in a function (the function for each model will be different), the dataset, the target column we are trying to predict (these two parameters will stay constant), and the value of K (the number of test splits):

```
CalculateCVAccuracy = function(FUN, dataframe, target, K) {
  #Vector that stores each K-fold CV score for each trial
  cvs = rep(0,5)
  for (i in 1:5) {
    set.seed(1 + i)
    n = nrow(dataframe)
    vad.index = sample(rep(1:K,n/K))
    accuracy = 0
    #For loop that computes K-fold CV score
    for (k in 1:K) {
      set.seed(10+k)
      train = dataframe[vad.index!=k, ]
      test = dataframe[vad.index==k, ]
      nk = sum(vad.index==k)
      accuracyk = FUN(dataframe, train, test, target)
      accuracy = accuracy + (nk/n) * accuracyk
    }
    cvs[i] = accuracy
  }
  #Returns the average K-fold CV score
  return (mean(cvs))
}
```

- **(a)**: Using **Logistic Regression**:

```
#Function to simulate logistic regression
logisticRegression = function(dataframe, train.set, test.set, target) {
  #Creating the logistic regression model
  logistic.fit = glm(target ~ ., data = train.set, family = "binomial")

  #Using the training model on the test set
  probability = predict(logistic.fit, test.set, type = "response");
  prediction = rep(0, 1)
  prediction[probability > 0.5] = 1
  test_pred <- ifelse(predict(logistic.fit, newdata = test.set,
  type = "response") > 0.5, 1, 0)
  test.accuracy = (mean(test_pred == test.set$target))*100
  return (test.accuracy)
}
CVAccuracy = CalculateCVAccuracy(logisticRegression, heart, heart$target, 5)
```

  The average 5-fold Cross-validation accuracy of the logistic regression model is ~ 82%.

- **(b)**: Using **Linear Discriminant Analysis (LDA)**:

```
#Function that trains the LDA model
LDA = function(dataframe, train.set, test.set, target) {
  lda.fit = lda(target ~ ., data = train.set)
  pred <- predict(lda.fit, test.set)
  test.accuracy = (mean(pred$class == test.set$target))*100
  return (test.accuracy)
}
CVAccuracy = CalculateCVAccuracy(LDA, heart, heart$target, 5)
```

  The average 5-fold Cross-validation accuracy of the LDA model is ~ 82%.

- **(c)**: Using **Quadratic Discriminant Analysis (QDA)**:

```
#Function that trains the QDA model
QDA = function(dataframe, train.set, test.set, target) {
  qda.fit = qda(target ~ ., data = train.set)
  pred <- predict(qda.fit, test.set)
  test.accuracy = (mean(pred$class == test.set$target))*100
  return (test.accuracy)
}
CVAccuracy = CalculateCVAccuracy(QDA, heart, heart$target, 5)
```

  The average 5-fold Cross-validation accuracy of the QDA model is ~ 80%.

This next part uses the Support Vector Machine (SVM) classifier. As it is more algorithmically intensive, I will compute a 2-fold Cross Validation accuracy for each classifier.

- **(d)**: Using a **Support Vector Machine (SVM)**. This will be a **linear** SVM and I will tune the model in order to find the best cost.

```
#Function that trains the SVM linear model
SVMlinear = function(dataframe, train.set, test.set, target) {
  #Finding the optimal cost
  tune.fit = tune(svm, target ~ ., data = train.set,
  kernel = "linear", type = 'C-classification', ranges = list(cost =
  seq(0.01, 10, 0.5)))
  best.fit = tune.fit$best.model

  #Finding test accuracy
  test_pred = predict(best.fit, test.set)
  test.accuracy = (mean(test_pred == test.set$target))*100
  return (test.accuracy)
}
CVAccuracy = CalculateCVAccuracy(SVMlinear, heart, heart$target, 2)
```

The average 2-fold Cross-validation accuracy of the SVM linear model is ~ 82%.

- **(e)**: Using a **Support Vector Machine (SVM)**. This will be a **radial** SVM and I will tune the model in order to find the best cost.

```
#Function that trains the SVM radial model
SVMRadial = function(dataframe, train.set, test.set, target) {
  #Finding the optimal cost
  tune.fit = tune(svm, target ~ ., data = train.set,
  kernel = "radial", type = 'C-classification',
  ranges = list(cost = seq(0.01, 10, 0.5)))
  best.fit = tune.fit$best.model

  #Finding test accuracy
  test_pred = predict(best.fit, test.set)
  test.accuracy = (mean(test_pred == test.set$target))*100
  return (test.accuracy)
}
CVAccuracy = CalculateCVAccuracy(SVMRadial, heart, heart$target, 2)
```

The average 2-fold Cross-validation accuracy of the SVM radial model is ~ 82%.

- **(f)**: Using a **Support Vector Machine (SVM)**. This will be a **polynomial** SVM and I will tune the model in order to find the best cost.

```
#Function that trains the SVM polynomial model
SVMPolynomial = function(dataframe, train.set, test.set, target) {
  #Finding the optimal cost
  tune.fit = tune(svm, target ~ ., data = train.set,
  kernel = "polynomial", type = 'C-classification',
  ranges = list(cost = seq(0.01, 10, 0.5)))
  best.fit = tune.fit$best.model

  #Finding test accuracy
  test_pred = predict(best.fit, test.set)
  test.accuracy = (mean(test_pred == test.set$target))*100
  return (test.accuracy)
}
CVAccuracy = CalculateCVAccuracy(SVMPolynomial, heart, heart$target, 2)
```

The average 2-fold Cross-validation accuracy of the SVM polynomial model is ~ 80%.

Conclusion: Most of the classification methods result in a CV test accuracy of around 80% - 82%. It appears that the Logistic Regression, LDA, SVM Linear, and SVM Radial models predict heart disease the best, but all classifiers generally work equally as well.