Human Activity Recognition

attackgnome

7/12/2020

Executive Summary

This notebook takes test and training data from the Weight Lifting Exercise Dataset at http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) to develop a model that can predict from accelerometer data, whether an exercise is being performed correctly or incorrectly.

##Load Data Download the relevant test and train datasets. Remove columns from the train data set that are composed entirely of NA's. Also remove time stamp data, window data, and the trainee's name.

```
#training data
url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
temp <- tempfile()
download.file(url,temp)
train <- fread(temp)
train <- train %>% select_if(~all(!is.na(.)))
train <- train[,8:60]
train <- train %>% mutate_if(is.character,as.factor)

#testing data
url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
temp <- tempfile()
download.file(url,temp)
test <- fread(temp)

unlink(temp)
rm(temp, url)
```

Validation dataset

The testing data provided is for the purposes of the course, but in order for us validate our model before submitting our results, we will want to test the model we create on a set of data where the answer is know. We will take the train data set and split it into one dataset, `train_df` that we will use to build the model, and a second dataset, `valid_df` that we will use to validate the model before submission.

```
set.seed(3456)
trainIndex <- createDataPartition(train$classe, p = .7,
                                  list = FALSE,
                                  times = 1)
train_df <- train[ trainIndex,]
valid_df  <- train[-trainIndex,]
```

Feature selection

Even after removing the column features composed entirely of Na's in the train dataset, There are approximately 60 potential features that we can use to predict the `classe` variable. We will use recursive feature elimination with cross validation use 10 folds to select the subset of predictors that we will want to use.

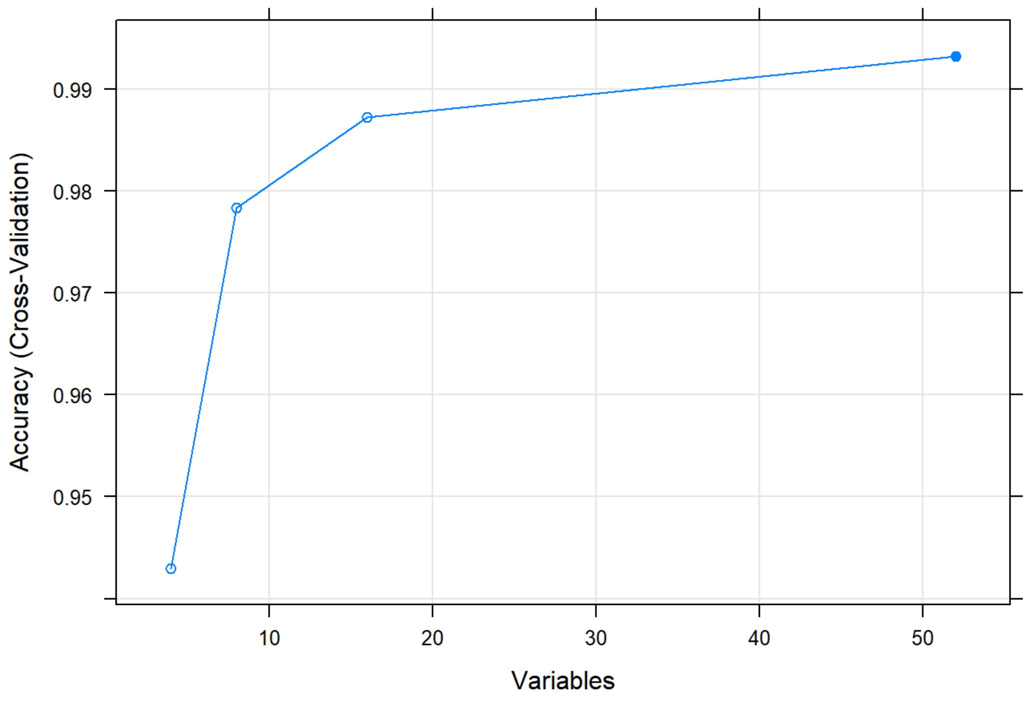
```
# define the control using a random forest selection function
control <- rfeControl(functions=rfFuncs, method="cv", number=5)
# run the RFE algorithm
results <- rfe(train_df[,1:52], train_df[[53]], rfeControl=control)
# summarize the results
print(results)
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (5 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD  KappaSD Selected
##          4   0.9429 0.9278   0.003514 0.004436
##          8   0.9783 0.9726   0.005597 0.007072
##         16   0.9873 0.9839   0.003086 0.003905
##         52   0.9932 0.9914   0.001301 0.001647        *
##
## The top 5 variables (out of 52):
##    roll_belt, yaw_belt, magnet_dumbbell_z, magnet_dumbbell_y, pitch_belt
```

```
# list the chosen features
predictors(results)
```

```
##  [1] "roll_belt"            "yaw_belt"             "magnet_dumbbell_z"
##  [4] "magnet_dumbbell_y"    "pitch_belt"           "pitch_forearm"
##  [7] "accel_dumbbell_y"     "roll_dumbbell"        "roll_forearm"
## [10] "magnet_forearm_z"     "magnet_dumbbell_x"    "roll_arm"
## [13] "accel_dumbbell_z"     "magnet_belt_y"        "magnet_belt_z"
## [16] "gyros_belt_z"         "yaw_arm"              "accel_forearm_x"
## [19] "magnet_belt_x"        "gyros_arm_y"          "magnet_forearm_y"
## [22] "yaw_dumbbell"         "accel_forearm_z"      "gyros_dumbbell_y"
## [25] "accel_belt_z"         "magnet_arm_z"         "total_accel_dumbbell"
## [28] "accel_forearm_y"      "gyros_forearm_y"      "gyros_arm_x"
## [31] "yaw_forearm"          "accel_dumbbell_x"     "pitch_arm"
## [34] "gyros_dumbbell_x"     "magnet_forearm_x"     "accel_arm_y"
## [37] "gyros_forearm_z"      "total_accel_forearm"  "accel_arm_x"
## [40] "accel_arm_z"          "gyros_dumbbell_z"     "total_accel_arm"
## [43] "magnet_arm_x"         "gyros_belt_x"         "magnet_arm_y"
## [46] "accel_belt_x"         "gyros_forearm_x"      "pitch_dumbbell"
## [49] "gyros_belt_y"         "total_accel_belt"     "accel_belt_y"
## [52] "gyros_arm_z"
```

```
# plot the results
plot(results, type=c("g", "o"))
```



From this feature selection method

combined with removing NA values, time stamp data and the training window data, we end up using 52 of the original in our model.

Model Validation

In the next step we will predict the class on the validation data that we held out and see what type of accuracy our model achieves.

```
predicted <- predict(results, valid_df, type = "raw")
```

```
## Warning in predict.rfe(results, valid_df, type = "raw"): ... are ignored for
## predict.rfe
```

```
accuracy(valid_df$classe, predicted$pred )
```

```
## [1] 0.9949023
```

It looks like our error rate is less than 1%. Considering that this is from data that we held out. I would expect to be able to achieve a similar out of sample error rate.

##Predict Test data

The final step is to use the model to predict the classes for the test data downloaded at the beginning of the notebook and predict those classes.

```
predict(results, test, type = "raw")
```

```
## Warning in predict.rfe(results, test, type = "raw"): ... are ignored for
## predict.rfe
##    pred     A     B     C     D     E
## 1     B 0.052 0.828 0.094 0.018 0.008
## 2     A 0.930 0.040 0.022 0.002 0.006
## 3     B 0.208 0.512 0.182 0.010 0.088
## 4     A 0.916 0.018 0.046 0.018 0.002
## 5     A 0.942 0.020 0.024 0.002 0.012
## 6     E 0.012 0.094 0.130 0.054 0.710
## 7     D 0.010 0.016 0.056 0.900 0.018
## 8     B 0.040 0.736 0.074 0.110 0.040
## 9     A 1.000 0.000 0.000 0.000 0.000
## 10    A 0.996 0.004 0.000 0.000 0.000
## 11    B 0.072 0.700 0.134 0.048 0.046
## 12    C 0.016 0.044 0.840 0.026 0.074
## 13    B 0.028 0.904 0.014 0.012 0.042
## 14    A 1.000 0.000 0.000 0.000 0.000
## 15    E 0.010 0.038 0.020 0.014 0.918
## 16    E 0.012 0.032 0.006 0.008 0.942
## 17    A 0.980 0.000 0.000 0.000 0.020
## 18    B 0.040 0.896 0.004 0.052 0.008
## 19    B 0.196 0.762 0.012 0.026 0.004
## 20    B 0.000 0.998 0.000 0.000 0.002
```