Project

gfmgn101

December 7, 2018

Problem statement

Develop a model that predicts whether or not an exercise is being done correctly. Use a data set from a study called the Qualitative Activity Recognition of Weight Lifting Exercises to train and predict the form of the exercise for 20 new observations.

Approach

The problem statement requires a model that predict whether someone doing an exercise correctly and if incorrectly what is the incorrect form. This problem does not require a model with high interpretability, because what is of interest is the predictive aspect rather than understanding any one particular predictor. As such, a random forest model will be used due to their low bias.

Libraries

```
library(readr)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

Loading

```
filenameTrain <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
filenameTest <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read_csv(filenameTrain, na = c("","#DIV/0!", "NA")) ##DIV/0! comes in as character
```

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

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   X1 = col_integer(),
##   user_name = col_character(),
##   raw_timestamp_part_1 = col_integer(),
##   raw_timestamp_part_2 = col_integer(),
##   cvtd_timestamp = col_character(),
##   new_window = col_character(),
##   num_window = col_integer(),
##   total_accel_belt = col_integer(),
##   kurtosis_yaw_belt = col_character(),
##   skewness_yaw_belt = col_character(),
##   max_picth_belt = col_integer(),
##   min_pitch_belt = col_integer(),
##   amplitude_pitch_belt = col_integer(),
##   accel_belt_x = col_integer(),
##   accel_belt_y = col_integer(),
##   accel_belt_z = col_integer(),
##   magnet_belt_x = col_integer(),
##   magnet_belt_y = col_integer(),
##   magnet_belt_z = col_integer(),
##   total_accel_arm = col_integer()
##   # ... with 28 more columns
## )
```

```
## See spec(...) for full column specifications.
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 3 parsing failures.
## row # A tibble: 3 x 5 col     row col               expected               actual file
        expected   <int> <chr>             <chr>                  <chr>  <chr>              actu
al 1  5373 magnet_dumbbell_z no trailing characters .6     'https://d396qusz~ file 2  5373 magne
t_forearm_y  no trailing characters .123   'https://d396qusz~ row 3  5373 magnet_forearm_z  no t
railing characters .0917  'https://d396qusz~
```

```
testing <- read_csv(filenameTest)
```

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

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   X1 = col_integer(),
##   raw_timestamp_part_1 = col_integer(),
##   raw_timestamp_part_2 = col_integer(),
##   num_window = col_integer(),
##   roll_belt = col_double(),
##   pitch_belt = col_double(),
##   yaw_belt = col_double(),
##   total_accel_belt = col_integer(),
##   gyros_belt_x = col_double(),
##   gyros_belt_y = col_double(),
##   gyros_belt_z = col_double(),
##   accel_belt_x = col_integer(),
##   accel_belt_y = col_integer(),
##   accel_belt_z = col_integer(),
##   magnet_belt_x = col_integer(),
##   magnet_belt_y = col_integer(),
##   magnet_belt_z = col_integer(),
##   roll_arm = col_double(),
##   pitch_arm = col_double(),
##   yaw_arm = col_double()
##   # ... with 37 more columns
## )
## See spec(...) for full column specifications.
```

Cleaning

```
length(training$X1)
```

```
## [1] 19622
```

```
training <- training[,colSums(is.na(training)) < nrow(training)] #for each col, sum the number o
f NAs. If the sum is less than the total number of rows then keep the column. creates a logical
 vector same length as number of columns. If the NAs equal the number or rows (entire row of NA)
 then exclude the column.
dim(training)
```

```
## [1] 19622   154
```

There are 19622 observations. Removing any columns that are completely NAs reduces the total columns to 154. However there are still NAs in the rows. Trying complete cases brings us the following:

```
complete <-complete.cases(training) #gives logical vector of complete cases
completeTraining <- training[complete,] #filters for those with complete cases
dim(completeTraining) #217 complete observations for 157 variables. Will there be enough overser
vations if we want to have a validation data set?
```

```
## [1] 217 154
```

```
wideTrain <- completeTraining[c(-1,-3,-4,-5,-6)] #get rid of X1, timestamps (assuming when exerc
ises recorded are irrelevant) and new window which are all 'yes'
dim(wideTrain)
```

```
## [1] 217 149
```

Because there are number of columns that are mostly NAs, complete cases reduces a huge number of rows. going from 19622 to 217, with 154 columns. Preference would be to have more observations proportionally to the number of columns. What if we ignore those columns that are mostly NAs? Ignore columns that have more than 19000 NAs.

```
omitNaColTrain <- training[,colSums(is.na(training))<19000]
dim(omitNaColTrain) #now left with 60 variables but a fuller set of observations
```

```
## [1] 19622    60
```

```
compNaColTrain <- omitNaColTrain[complete.cases(omitNaColTrain),]
dim(compNaColTrain) #complete cases took out one row to make it 19621 rows instead of 19622
```

```
## [1] 19621    60
```

```
tallTrain <- compNaColTrain[c(-1,-3,-4,-5,-6)] #take out X1, dates and new window
dim(tallTrain)
```

```
## [1] 19621    55
```

The resulting tallTrain dataframe is 19621 rows and 55 columns.

Random forest

As mentioned above, random forest is being used as a model as it has a high predictive capability though at the cost of interpretability, which is not as important in this use case. The Caret library is used to train the data on ‘classe’ and then the predict function is used to predict the classes using the model. Cross validation is not needed

```
#Random forest. Cross validation is already baked in as not all observations are used to create
 the decision trees.

tallTrainRF <- train(classe ~ ., data=tallTrain, method = 'rf')
tallTrainRF
```

```
## Random Forest
##
## 19621 samples
##    54 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 19621, 19621, 19621, 19621, 19621, 19621, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9947535  0.9933627
##   30    0.9977535  0.9971581
##   58    0.9948843  0.9935274
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 30.
```

```
dim(testing)
```

```
## [1]  20 160
```

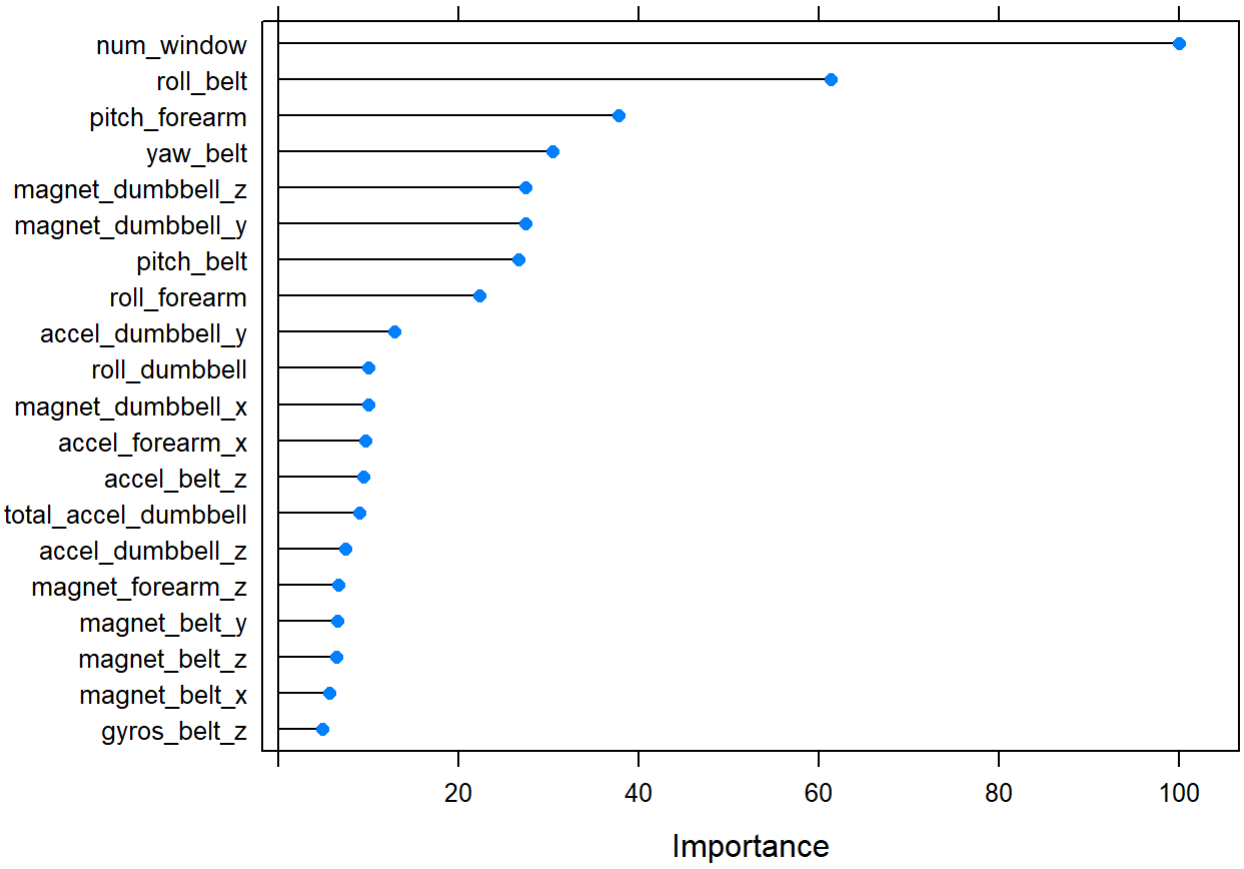
```
testdf <- testing[,colnames(tallTrain[,-55])] #testing dataframe has all the original columns, n
eed to subset just to the ones that were used in the training set
predRF <- predict(tallTrainRF, testdf)
predRF
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

PredRF produces predictions for each of the 20 new observations.

Determining which variables are the most important

```
RFimp <- varImp(tallTrainRF)
plot(RFimp, top = 20)
```



Conclusion

The twenty test cases follow the classes: B A B A A E D B A A B C B A E E A B B B The num_window, roll belt, pitch forearm, yaw belt, and magnet dumbbell z are the five top contributing variales to the model. The first suggesting that the test cases were taken from the same window of time as similar exercises in the training test set.