ML_Project_jdhuffaker

jdhuffaker

November 21, 2015

Project Objective and Abstract

The objective of this project is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict the manner in which they did the exercise. They were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). This is the "classe" variable in the training set. Numerous predictor variables were measured at each accelerometer location. These variables are preprocessed and used to develop machine learning models to predict 20 different test cases. The methods for building the models are described along with the final model chosen based on estimated out of sample accuracy.

More information and data is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Load, Clean, and Prepare the Data

```
## Loading required package: lattice
## Loading required package: ggplot2
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
## Loading required package: RGtk2
## Rattle: A free graphical interface for data mining with R.
## Version 3.5.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
##
## Attaching package: 'combinat'
##
## The following object is masked from 'package:utils':
##
##     combn
##
##
## Attaching package: 'pls'
##
## The following object is masked from 'package:caret':
##
##     R2
```

```
##
## The following object is masked from 'package:stats':
##
##     loadings
```

There are 160 original columns and 19622 rows in the training data set.

The following cleaning and transformations were done on the training and testing data sets:

1. All NA columns were removed (this removed 5-6 empty columns and all the summary statistics columns) for a total of 100 columns removed. The column names removed are:

kurtosis_roll_belt, kurtosis_picth_belt, kurtosis_yaw_belt, skewness_roll_belt, skewness_roll_belt.1, skewness_yaw_belt, max_roll_belt, max_picth_belt, max_yaw_belt, min_roll_belt, min_pitch_belt, min_yaw_belt, amplitude_roll_belt, amplitude_pitch_belt, amplitude_yaw_belt, var_total_accel_belt, avg_roll_belt, stddev_roll_belt, var_roll_belt, avg_pitch_belt, stddev_pitch_belt, var_pitch_belt, avg_yaw_belt, stddev_yaw_belt, var_yaw_belt, var_accel_arm, avg_roll_arm, stddev_roll_arm, var_roll_arm, avg_pitch_arm, stddev_pitch_arm, var_pitch_arm, avg_yaw_arm, stddev_yaw_arm, var_yaw_arm, kurtosis_roll_arm, kurtosis_picth_arm, kurtosis_yaw_arm, skewness_roll_arm, skewness_pitch_arm, skewness_yaw_arm, max_roll_arm, max_picth_arm, max_yaw_arm, min_roll_arm, min_pitch_arm, min_yaw_arm, amplitude_roll_arm, amplitude_pitch_arm, amplitude_yaw_arm, kurtosis_roll_dumbbell, kurtosis_picth_dumbbell, kurtosis_yaw_dumbbell, skewness_roll_dumbbell, skewness_pitch_dumbbell, skewness_yaw_dumbbell, max_roll_dumbbell, max_picth_dumbbell, max_yaw_dumbbell, min_roll_dumbbell, min_pitch_dumbbell, min_yaw_dumbbell, amplitude_roll_dumbbell, amplitude_pitch_dumbbell, amplitude_yaw_dumbbell, var_accel_dumbbell, avg_roll_dumbbell, stddev_roll_dumbbell, var_roll_dumbbell, avg_pitch_dumbbell, stddev_pitch_dumbbell, var_pitch_dumbbell, avg_yaw_dumbbell, stddev_yaw_dumbbell, var_yaw_dumbbell, kurtosis_roll_forearm, kurtosis_picth_forearm, kurtosis_yaw_forearm, skewness_roll_forearm, skewness_pitch_forearm, skewness_yaw_forearm, max_roll_forearm, max_picth_forearm, max_yaw_forearm, min_roll_forearm, min_pitch_forearm, min_yaw_forearm, amplitude_roll_forearm, amplitude_pitch_forearm, amplitude_yaw_forearm, var_accel_forearm, avg_roll_forearm, stddev_roll_forearm, var_roll_forearm, avg_pitch_forearm, stddev_pitch_forearm, var_pitch_forearm, avg_yaw_forearm, stddev_yaw_forearm, var_yaw_forearm.

2. Added one epoch time column to replace the cvtd_timestamp column since it read in as a factor column.

3. Removed the columns "X", "user_name", "new_window", and "cvtd_timestamp". Removed "user_name" and "cvtd_timestap" because you probably wouldn't use these for future predictions of new participants at future dates. Column "X" is a row index. Column "new_window" was used to identify the optimal overlapping time span (2.5s as indicated in one of the reference reports) for the summary statistics.

4. Checked for near zero variance predictor variables after the prevous cleansing and there were none to remove.

The number of columns left in the training an testing data sets is 57.

I then performed a pairwise correlation and removed predictor variables that had a correlation coefficient of 0.90 or higher with other predictor variables. There were 8 variables removed: accel_belt_z, roll_belt, accel_belt_y, accel_belt_x, cvtd_timestamp_epoch, gyros_dumbbell_x, gyros_dumbbell_z, gyros_arm_x

The same variables were removed in the testing data set. The final remaining variables in the train data set is 49. This includes the response variable classe.

Plotting the Features (Variables)

Note that feature plots were created and reviewed but not added to this report. They took too much time to generate and difficult to display in the report. However, the R code is provided to generate them if desired.

Machine Learning Algorithms/Models

Recursive Partitioning and Regression Trees (rpart) Model

I first tried the recursive partitioning and regression trees (rpart) package for classifying the classe response. I used the 10-fold cross validation method and changed the number of cross validation (cv) repeats and tuning length. Below are the results for three cv repeats and tuning length combinations. The third one had the best accuracy of 95.3%.

1. Accuracy for 3 repeats of 10-fold cv and tuneLength=10, cp=0.0189, accuracy=0.684, Kappa=0.599
2. Accuracy for 5 repeats of 10-fold cv and tuneLength=30, cp=0.0050, accuracy=0.892, Kappa=0.863
3. Accuracy for 10 repeats of 10-fold cv and tuneLength=50, cp=0.0015, accuracy=0.953, Kappa=0.940

The model results of the third rpart model is given below:

```
## CART
##
## 19622 samples
##    48 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 17660, 17659, 17662, 17660, 17660, 17659, ...
## Resampling results across tuning parameters:
##
##   cp           Accuracy   Kappa      Accuracy SD  Kappa SD
```

```
##   0.001495513  0.9528741  0.9403889  0.005173345  0.006542922
##   0.001637943  0.9499284  0.9366646  0.005793658  0.007330147
##   0.001709158  0.9482620  0.9345558  0.006093031  0.007709791
##   0.001780373  0.9467582  0.9326546  0.006463522  0.008178683
##   0.001922803  0.9442047  0.9294247  0.006969599  0.008819659
##   0.001994018  0.9425535  0.9273370  0.007100593  0.008984360
##   0.002029625  0.9422222  0.9269180  0.007101086  0.008984839
##   0.002207663  0.9392001  0.9230976  0.006543343  0.008278161
##   0.002278878  0.9376661  0.9211631  0.006395226  0.008090738
##   0.002777382  0.9295221  0.9108326  0.007877491  0.009980442
##   0.002848597  0.9283802  0.9093819  0.008309020  0.010528992
##   0.002991027  0.9256794  0.9059551  0.008404543  0.010654067
##   0.003062242  0.9239414  0.9037495  0.008765502  0.011111059
##   0.003133457  0.9222087  0.9015512  0.008880735  0.011259370
##   0.003703176  0.9159199  0.8935839  0.010196387  0.012918609
##   0.003916821  0.9130201  0.8899155  0.010091450  0.012784719
##   0.004059251  0.9107776  0.8870950  0.010822020  0.013691127
##   0.004415325  0.9058643  0.8808863  0.011477844  0.014531051
##   0.004628970  0.9013080  0.8751201  0.012994052  0.016446509
##   0.004913830  0.8938573  0.8657014  0.013333570  0.016874083
##   0.004985045  0.8927104  0.8642453  0.013529642  0.017140910
##   0.005269905  0.8854839  0.8550692  0.014376907  0.018249460
##   0.005483549  0.8768658  0.8441082  0.014561321  0.018542097
##   0.005697194  0.8679685  0.8328349  0.014291172  0.018191532
##   0.005768409  0.8650332  0.8291059  0.013553960  0.017254348
##   0.006195699  0.8522207  0.8128232  0.014513075  0.018408651
##   0.006338128  0.8496012  0.8094696  0.014489348  0.018370318
##   0.006409343  0.8488061  0.8084550  0.014859815  0.018834820
##   0.006907848  0.8396329  0.7967654  0.015132974  0.019155518
##   0.007976072  0.8317032  0.7866207  0.012287369  0.015545333
##   0.008973081  0.8205221  0.7724672  0.010605434  0.013474772
##   0.009115511  0.8181366  0.7694163  0.011128404  0.014157355
##   0.011251958  0.8065831  0.7546845  0.015206180  0.019170211
##   0.011964108  0.7933426  0.7378565  0.018693871  0.023582653
##   0.012391397  0.7838122  0.7257578  0.019463268  0.024529543
##   0.012533827  0.7801074  0.7210481  0.019690184  0.024804823
##   0.014456630  0.7333278  0.6621508  0.025055004  0.031532356
##   0.014599060  0.7281259  0.6556208  0.022352722  0.028153552
##   0.014670275  0.7245128  0.6510862  0.020322778  0.025628329
##   0.015952144  0.7085921  0.6311087  0.015679563  0.019746525
##   0.018943170  0.6853376  0.6008867  0.018625575  0.023461641
##   0.019014385  0.6832840  0.5982118  0.019167964  0.024107077
##   0.020296254  0.6660530  0.5764527  0.020116492  0.025118253
##   0.020865974  0.6544996  0.5619122  0.020199937  0.025323925
##   0.023358496  0.6270162  0.5268632  0.026938408  0.033716460
##   0.027951859  0.5902303  0.4799700  0.024390434  0.030707309
##   0.030266344  0.5679332  0.4516288  0.013449222  0.016916280
##   0.035322604  0.5395888  0.4161261  0.015498714  0.019373129
##   0.038847742  0.5062679  0.3660828  0.031411427  0.050305370
##   0.066123059  0.3889283  0.1731853  0.091398249  0.151360383
```

```
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.001495513.
```

The predicted values of the 20 test cases for the third rpart model are:

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

Note that I generated a fancy rplot model showing the tree structure, but it is too busy to show in this report. See my R code to generate the plot if desired.

Partial Least Squares (PLS) Model

Next I fit a PLS model using 5 repeats of 10-fold cross validation resampling and a tune length of 15. The accuracy was only 61.8% which is worse than the accuracies of the three rpart models. Note that I should have used the partial least squares discriminant analysis model (plsda) but didn't have time. Therefore, I'm assuming that the PLS method in the caret train fuction converted the classe categorical values to numeric values for the modeling. I also tried 10 repeats of 10-fold cross validation resampling and got the same accuracy.

The predicted values of the 20 test cases for the PLS model are: D, A, A, A, A, C, D, D, A, A, D, A, E, A, E, B, A, B, B, B.

Linear Discriminant Analysis (LDA) Model

Next I fit the LDA model using 10 repeats of 10-fold cross validation resampling. The accuracy was 69.0% which is still less than two of the rpart models.

The predicted values of the 20 test cases for the PLS model are: B, A, A, A, A, C, D, D, A, A, D, A, E, A, E, A, A, B, B, B.

Random Forest (RF) Model

```
## mtry = 6  OOB error = 0.05%
## Searching left ...
## mtry = 4     OOB error = 0.09%
## -0.7 0.01
## Searching right ...
## mtry = 9     OOB error = 0.05%
## 0 0.01
```

The final model I tried is the random forest model. Note that I tried running it using the caret train function, but it only ran on my 32GB RAM desktop and took at least 1 hour to finish. Therefore, I used the randomForest fuction which was much faster.

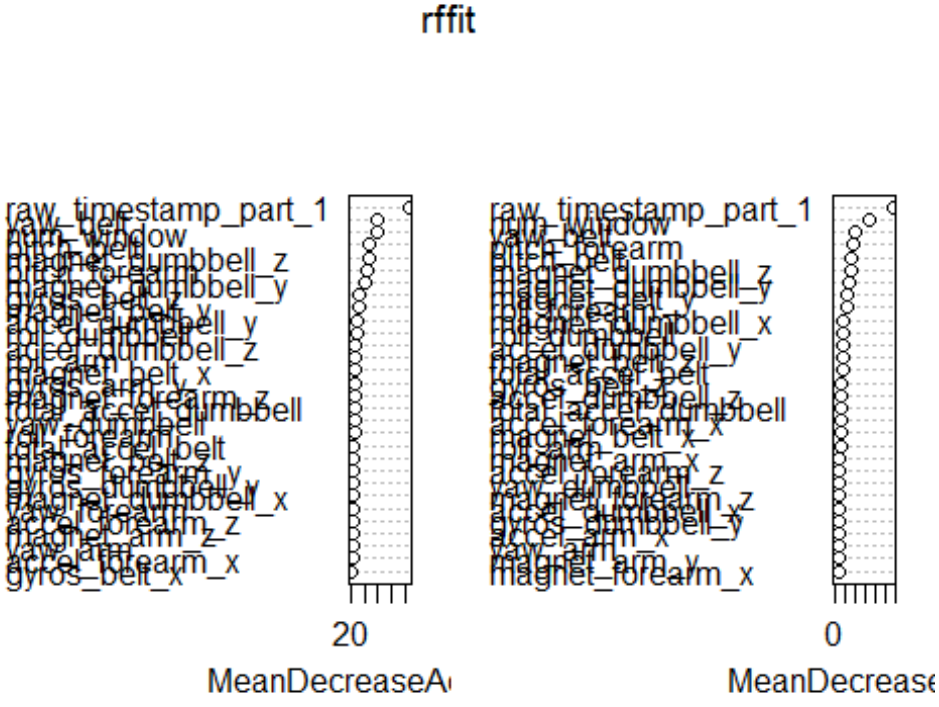
I used the tuneRF function to find the optimal mtry setting using ntreeTry set to 500. The optimal mtry was 13 with the minimum out-of-bag (OOB) error as shown below:

4, 6, 9, 8.663744810^{-4}, 5.096320510^{-4}, 5.096320510^{-4}

Note that the optimal mtry value changes if ntreeTry is set to different values. For example, when I set ntreeTry to 100 the optimal mtry was 6. With mtry=6, I tried ntree=10, 50, 100, 200, and 500 and OOB error rate was 1.5%, 0.1%, 0.07%, 0.06%, and 0.05%, respectively. However, the run time increased also for each incremental increase in ntree value. With mtry=13 and ntree=500, OOB error rate is 0.04% but takes a lot longer to run.

The random forest modeling results including variable importance are as follows;

```
##
## Call:
##  randomForest(formula = classe ~ ., data = htrain1, ntree = 500,      mtry
= 13, importance = TRUE, trace = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 13
##
##         OOB estimate of  error rate: 0.04%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5580    0    0    0    0 0.0000000000
## B    2 3795    0    0    0 0.0005267316
## C    0    3 3419    0    0 0.0008766803
## D    0    0    2 3213    1 0.0009328358
## E    0    0    0    0 3607 0.0000000000
```



The random forest had the highest accuracy of 99.96% (assuming that accuracy is 100% - OOB error rate). This generated the best results out of all the models. The predicted values of the 20 test cases for the PLS model are: B, A, B, A, A, E, D, B, A, A, B, C, B, A, E, E, A, B, B, B.

Combined Predicted Results of the 20 Test Use Cases for all Models

The combined predicted values of the 20 test use cases for all models is given in the table below. The results of the random forest predicted values were used as the answers in course project submission assignment since that model gave the highest accuracy. All of them were the correct predicted values.

```
##    PLS Acc=61.8% LDA Acc=69.0% RPart_1 Acc=68.4% RPart_2 Acc=86.3%
## 1              D              B                D                 c
## 2              A              A                A                 A
## 3              A              A                c                 B
## 4              A              A                A                 A
## 5              A              A                A                 A
## 6              c              c                c                 E
## 7              D              D                D                 D
## 8              D              D                B                 D
## 9              A              A                A                 A
## 10             A              A                A                 A
## 11             D              D                c                 c
## 12             A              A                c                 c
## 13             E              E                B                 B
## 14             A              A                A                 A
## 15             E              E                c                 E
## 16             B              A                B                 E
## 17             A              A                A                 A
## 18             B              B                A                 A
## 19             B              B                B                 B
## 20             B              B                B                 B
##    RPart_3 Acc=95.3% RandomForest Acc=99.96%
## 1                 B                       B
## 2                 A                       A
## 3                 B                       B
## 4                 A                       A
## 5                 A                       A
## 6                 E                       E
## 7                 D                       D
## 8                 D                       B
## 9                 A                       A
## 10                A                       A
## 11                c                       B
## 12                c                       c
## 13                B                       B
## 14                A                       A
## 15                E                       E
## 16                E                       E
## 17                A                       A
```

```
## 18                   A                    B
## 19                   B                    B
## 20                   B                    B
```

Conclusion and Recommendations

Overall, there are many models and tuning variations of those models that can be used to predict the classe response variable. From this exercise, random forest proved to be the best modeling method. Below is a list of modeling methods that could also be tried given more time:

1. Boosted trees (gbm)
2. Discriminant Analysis PLS (plsda)
3. Regularized discriminant Analysis (rda)
4. Etc.

It may also be worthwhile to try principle components (PCA) preprocessing on the predictor variables first.

References

1. Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.