Practical Machine Learning

Project Write-up

Summary: The purpose of this project is to use a machine learning algorithm to predict exercise mode. Various sensors on the body of the person provide readings and these readings have to be used to ***predict*** the exercise mode. In the dataset there are 5 exercise modes in the variable *classe*, titled A, B, C, D and E. I spent MOST of the time cleaning up and understanding the data. The raw data set had 160 variables, but many of the columns contained no information at all. Also, some of the variables were numeric, but many variables were qualitative (e.g., name of individual). I took a two phase approach to the project (to be described below). The best algorithm was a **Random Forest algorithm**, which provided an **OUT-OF-SAMPLE classification accuracy** of about **94% on one particular TEST data set**. CROSS VALIDATION was also performed by tuning the **trControl** parameter in the caret package. **For the twenty test cases (much smaller TEST SAMPLE), this algorithm was PERFECT and predicted all twenty cases accurately.**

Machine Learning Algorithms:

- **At least TWO machine learning algorithms were tried: a) a DECISION TREE (using method rpart) and b) a RANDOM FOREST (method = rf)**

TRAINING AND TEST DATA SETS:

- **The training data set provided for this project had 19622 rows (and 160 columns). These 19622 cases were further split into a "training" set consisting of 13737 observations and a "test" set consisting of 5885 observations. Note that this test set is an OUT-OF-SAMPLE test set and different from the 20 test cases provided for the project.**
- **Any cross-validation performed was therefore based upon the 13737 observations in the "training" set and the best cross-validated model was applied to the out-of-sample test data set (5885 observations).**
- **The classification accuracy on the 5885 OUT-OF-SAMPLE test set was about 94% and the classification accuracy was 100% for the 20 test cases.**

CROSS-VALIDATION:

- **The best algorithm was the Random Forest. The Caret package performs its own cross-validation, but in addition, cross-validation was "forced" using the following commands in the trControl parameter:**

```
fitControl <- trainControl(method = "repeatedcv", number = 5, repeats = 5)

modFit.rf <- train(classe ~
pitch_forearm+pitch_arm+magnet_belt_y+magnet_belt_z+accel_arm_x+
```

accel_forearm_x+ magnet_arm_x+ magnet_arm_y+ magnet_dumbbell_z+ magnet_forearm_x, data=training2 , method = "rf", trControl = fitControl, PROX=TRUE)

The cross-validation improved prediction accuracy just a little bit (when compared to directly using the RandomForest routine, without using Caret at all).

-

Description of Two Phase Approach taken for the Project:

PHASE 1: EXPLORATORY DATA ANALYSIS

This phase of the project took the most time. Upon exploratory analysis, it was apparent that there were many columns of data that were empty, or had many missing values. There were also some Nan variables (e.g., division by 0). If an entire column was empty, the variable was dropped from the analysis. I also tried using only quantitative variables in the prediction model. A very preliminary random forest run with all variables took an inordinately long period of time to train the model. Hence, it was clear that the features had to be pruned in some manner. I created a quantitative variable to mimic the 5 classed (A = 1, B = 2 and so on). **I performed a correlation analysis of the dependent variable against all of the features to see if any key predictors could be found.** This exercise led to about 10 predictors that seemed important as listed below (see APPENDIX FOR SOME EXPLORATORY PLOTS PERFORMED DURING THIS PHASE).

Sample Feature correlations with dependent variable (only some variables are included here for illustration):

"magnet_belt_x"

[1] 0.030325

[1] "magnet_belt_y"

[1] -0.314773

[1] "magnet_belt_z"

[1] -0.154828

[1] "roll_arm"

[1] 0.04246

[1] "pitch_arm"

[1] -0.200614

[1] "yaw_arm"

[1] 0.011793

The final (extended) features selected from the exploratory analyses in PHASE 1 were:

pitch_forearm + pitch_arm + magnet_belt_y + magnet_belt_z + accel_arm_x + accel_forearm_x + magnet_arm_x + magnet_arm_y + magnet_dumbbell_z + magnet_forearm_x

PHASE 2:

MODEL BUILDING:

Various machine learning algorithms were attempted in Phase 2. The strategy can be summarized as:

- Create two feature sets, one **SIMPLE (with 4 features)** and one **EXTENDED** (with *all* the features identified at the end of Phase I). The 4 features included in the *simple model* were **pitch_forearm + magnet_belt_y + magnet_belt_z + magnet_dumbbell_z. These 4 features were selected based upon their correlations with the classe variable.**
- Try various machine learning algorithms in combination with these feature sets. I'll report on just two algorithms a) decision tree (the rpart method) and b) random forests (RandomForest method).
- **THE BEST COMBINATION WAS THE EXTENDED FEATURE SET AND THE RANDOM FOREST ALGORITHM (94% accuracy for the out-of-sample test set of size 5885 and a 100% accuracy for the TWENTY test cases provided in the project submission).**

A) SAMPLE OUTPUTS FROM THE DECISION TREE (MACHINE LEARNING ALGORITHM 1)

1) root 983 704 A (0.28 0.19 0.17 0.16 0.18)

2) pitch_forearm < -31.7 90 1 A (0.99 0.011 0 0 0) *

3) pitch_forearm >= -31.7 893 703 A (0.21 0.21 0.19 0.18 0.2)

6) magnet_belt_y >= 555 812 622 A (0.23 0.23 0.21 0.18 0.14)

12) magnet_dumbbell_z < -18.5 300 181 A (0.4 0.27 0.097 0.17 0.06) *

13) magnet_dumbbell_z >= -18.5 512 369 C (0.14 0.21 0.28 0.19 0.19)

26) magnet_belt_y >= 592.5 376 257 C (0.1 0.21 0.32 0.24 0.12) *

27) magnet_belt_y < 592.5 136 87 E (0.24 0.19 0.18 0.037 0.36)*

7) magnet_belt_y < 555 81 14 E (0 0.012 0 0.16 0.83) *

The decision tree algorithm provided a poor classification accuracy of only about 48%.

Here is a cross-tabulation to illustrate:

predtree	**A**	**B**	**C**	**D**	**E**
A	3309	650	278	207	256
B	431	582	23	45	60
C	709	1091	1554	624	473
D	548	705	720	1567	657
E	304	579	675	612	1980

B) PERFORMANCE OF RANDOM FORESTS (MACHINE LEARNING ALGORITHM 2)

THREE SUB-CASES WERE CONSIDERED WITHIN RANDOM FOREST

B- RF-CASE 1: The Random Forest built DIRECTLY using the randomForest function (i.e., no use of CARET)

The random forest algorithm did much better (than rpart) and provided a classification accuracy (for a test data set held out from the training sample) of about 94%.

Here is some sample output (**confusion Matrix**) from the random forest procedure.

yhat.rf	**1**	**2**	**3**	**4**	**5**
A	**1622**	45	3	6	4
B	26	**1040**	26	7	16
C	10	36	**962**	46	18
D	12	12	25	**889**	23
E	4	6	10	16	**1021**

The above confusion Matrix can be summarized by adding the diagonal elements where there is agreement in the predictions as below (TRUE = agreement):

Mode FALSE TRUE NA's

logical 351 5534 0

So the accuracy is 5534/(5534+351) = 94% for this specific cross validation procedure only.

B- RF-CASE 2: The Random Forest built using CARET, but with DEFAULT Caret cross validation:

CROSS-VALIDATION: This accuracy can be improved (very slightly) by K-fold cross-validation, however for the twenty test cases, this Random Forest algorithm was sufficient to get 100% accuracy.

The same Random Forest Model was built using CARET (with default CROSS VALIDATION controlled by CARET itself). The results were marginally better, but not by much (320 false vs. 351). This model too had a 100% prediction accuracy for the 20 TEST cases.

CASE 2: CROSS-VALIDATION WITH DEFAULT SETTINGS IN CARET: Training data set size = 13737 records

OUT OF SAMPLE ERROR CALCULATION FOR TEST SET WITH 5585 OBSERVATIONS

Mode	FALSE	TRUE	NA's
logical	320	5565	0

CONFUSION MATRIX:

yhat.rf	1	2	3	4	5
A	1634	37	2	7	5
B	17	1061	29	5	14
C	9	29	961	57	19
D	11	8	21	884	19
E	3	4	13	11	1025

B- RF-CASE 3: The Random Forest built using CARET, but an attempt was made to control the cross validation procedure with the trControl parameter in caret:

The setting of the trControl can be described in these lines of code:

```
# SET UP PARAMETER FOR CROSS VALIDATION - fitControl passed to TRAIN later

 fitControl <- trainControl(method = "repeatedcv", number = 5, repeats = 5)

 modFit.rf <- train(classe ~
pitch_forearm+pitch_arm+magnet_belt_y+magnet_belt_z+accel_arm_x+
accel_forearm_x+ magnet_arm_x+ magnet_arm_y+ magnet_dumbbell_z+
magnet_forearm_x, data=training2 , method = "rf", trControl = fitControl, PROX=TRUE)
```

CASE 3 - CROSS VALIDATION CONTROLLED USING TR CONTROL:

Mode	FALSE	TRUE	NA's
logical	321	5564	0

yhat.rf	1	2	3	4	5
A	1627	36	3	8	3
B	21	1062	27	7	14
C	9	27	962	55	20
D	14	10	23	887	19
E	3	4	11	7	1026

We can conclude that cross validation using CARET improved the prediction accuracy somewhat.

FINAL PREDICTION FOR THE 20 TEST CASES:

This final discussion pertains to the prediction for the 20 test cases provided in the project. The best random forest algorithm was applied to these twenty cases. The prediction provided for these twenty cases was completely accurate (as verified by the Project submission scorecard system using twenty text files with the answers).

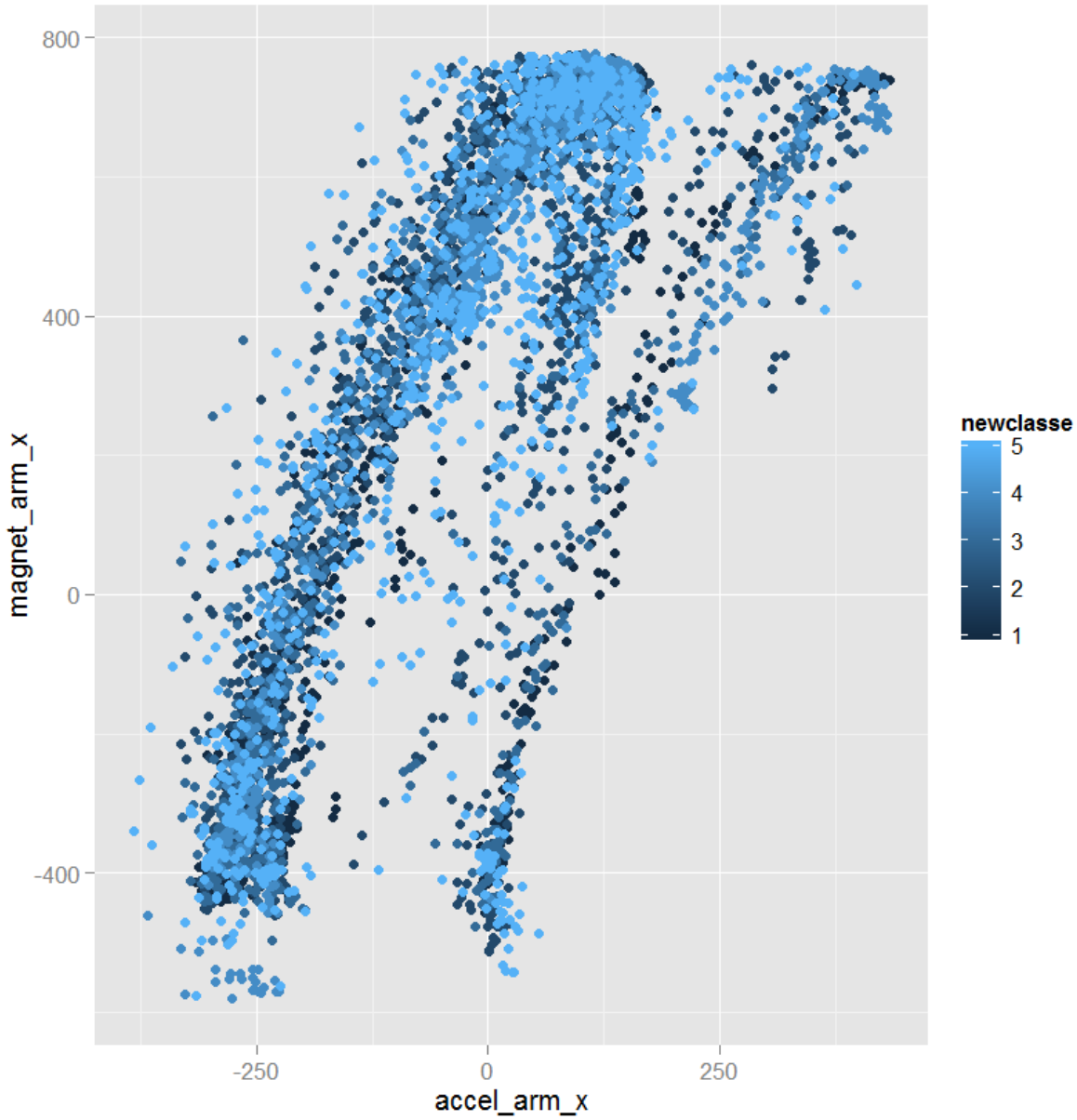
Predictions for the 20 test cases:

B A B A A E D B A A B C B A E E A B B B

APPENDIX (EXPLORATORY PLOTS DURING DATA ANALYSIS):

SAMPLE EXPLORATORY PLOTS (OBTAINED DURING EXPLORATORY ANALYSES):

PLOT 1: TWO DIMENSIONAL PLOT OF ACCEL_ARM_X vs. MAGNET_ARM_X WITH DEPENDENT VARIABLE COLORED (magnet_arm_x is strongly correlated with dependent variable)



PLOT 2: MAGNET_BELT_Y VS. NEWCLASSE