Prediction-Assignment

Deborah Passey

7/28/2019

INTRODUCTION

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it.

The data set for this analysis is from:
Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.

METHODS

This project uses data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. This data is used to predict whether participants did the exercise correctly or incorrectly.

This project uses two datsets: a training dataset with 19,622 observations of 53 variables, and a testing dataset of 20 observations of 53 variables. The data sets were cleaned up to removed variables that most of the observations missing.

```
    setwd("C:/Users/Deborah Passey/Desktop")

    pml_training <- read.csv("pml-training.csv", na.strings = c("","NA","#DIV/0!"))
    pml_testing <- read.csv("pml-testing.csv", na.strings = c("","NA","#DIV/0!"))

    training <- pml_training[lapply(pml_training, function(x) sum(is.na(x)) / length(x) ) < 0.05
]
    testing <- pml_testing[lapply(pml_testing, function(x) sum(is.na(x)) / length(x)) < 0.05]

    training_data <- training[,-c(1:7)]
    testing_data <- testing[,-c(1:7)]

    training_data$classe <- unclass(training_data$classe)
    testing_data$classe <- unclass(testing_data$classe)

    inTrain <- createDataPartition(y=training_data$classe, p=0.7,list = FALSE)
    train <- training_data[inTrain,]
    test <- training_data[-inTrain,]
```

Data Summary for Participants

In order to cut down on processing time and predictors, the dataset was trimmed down to a select number of variables. The research from Ugulino et al (2012) was used a selection algorithm to identify 16 variables that are potentially the best at predicting whether participants did the exercise correctly: (1) Sensor on the belt - acceleration, pitch, yaw, and roll, (2) Sensor on the arm - acceleration, pitch, yaw, and roll, (3) Sensor on the forearm - acceleration, pitch, yaw, and roll, and (4) Sensor on the dumbbell - accleraton, pitch, yaw, and roll.
The data table below shows the average for each of the six participants. The table reports the average of the Euler angles: roll, pitch and yaw, and accelerometer (accel) data for the arm, dumbbell, forearm, and belt.

```
    means <- training[,c("user_name", "roll_belt","roll_arm", "roll_dumbbell", "roll_forearm",
"pitch_belt", "pitch_arm", "pitch_dumbbell", "pitch_forearm", "yaw_belt", "yaw_arm", "yaw_dumbbe
ll", "yaw_forearm","total_accel_arm", "total_accel_belt", "total_accel_dumbbell","total_accel_fo
rearm")]
    dataset <- means %>% group_by(user_name) %>% summarise_each(funs(mean))
    summ <- kable(dataset) %>%
    kable_styling(bootstrap_options = c("striped", "hover", "condensed", "responsive")) %>% scro
ll_box(width = "100%", height = "200px")
    summ
```

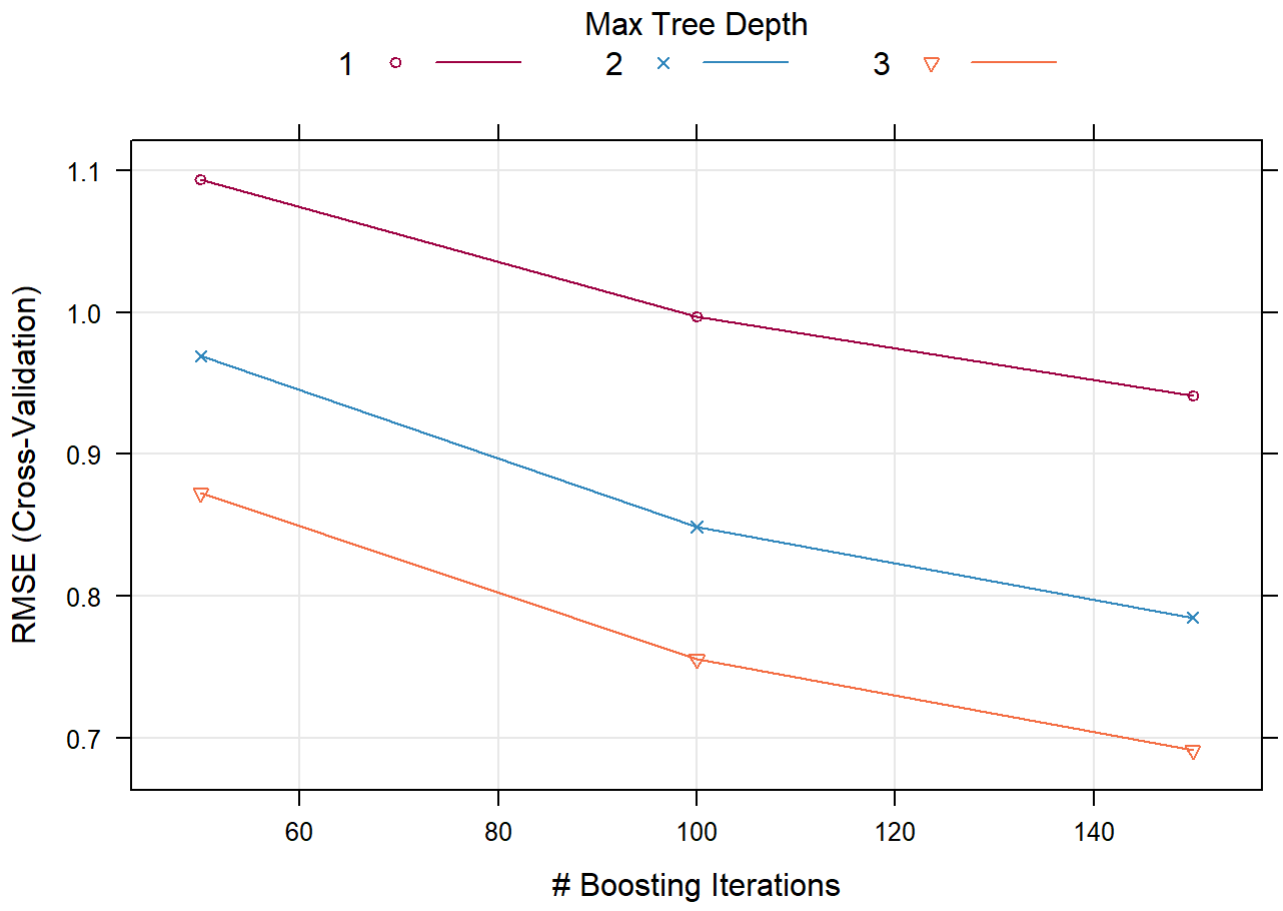
user_name	roll_belt	roll_arm	roll_dumbbell	roll_forearm	pitch_belt	pitch_arm	pitch_dumb
adelmo	128.2869990	-15.70269	61.688812	0.000000	-41.396737	5.204404	-22.903
carlitos	1.1677410	50.81837	28.207232	39.716886	6.582005	-13.245909	3.884
charles	122.0277149	-45.72347	-3.803530	-11.705710	15.762226	-12.492098	-7.397
eurico	-0.0102345	90.25534	6.832875	-1.639225	3.358449	-6.694892	6.481
jeremy	0.7217578	0.00000	61.315148	108.166149	4.103906	0.000000	-47.904

RESULTS

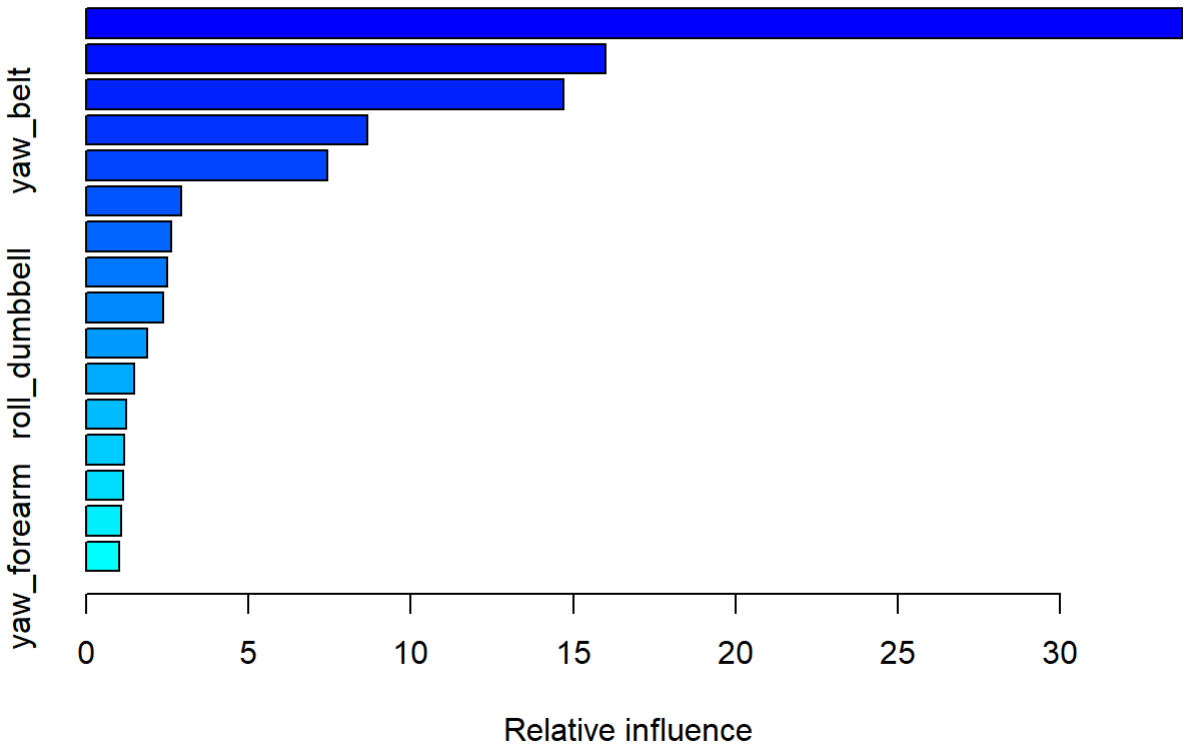
Three models were fit with the training data set: (1) gradient boosting model (GBM), (2) linear model, and (3) random forest. To increase repoducibility, the “set.seed(150)” function was used for each model. The GBM used a k-fold cross-validation, where the dataset is split into k-subsets. Each subset is held out while the model is trained on the other subsets. This process is completed to determine accuracy for each of the datasets, and an overall accuracy estimate is provided. The linear model and random forest models were fit with “classe” as the outcome and 16 variables as predictors.

```
## GBM Model
  set.seed(200)
  control <- trainControl(method = "cv", number = 5)
  gbm_model <- train(classe ~ ., method = "gbm", data = train[,c("classe", "roll_belt","roll_ar
m", "roll_dumbbell", "roll_forearm", "pitch_belt", "pitch_arm", "pitch_dumbbell", "pitch_forear
m", "yaw_belt", "yaw_arm", "yaw_dumbbell", "yaw_forearm","total_accel_arm", "total_accel_belt",
"total_accel_dumbbell","total_accel_forearm")], trControl= control, verbose=FALSE)

## Linear Model
  set.seed(200)
  lm_model <- train(classe ~.,data = train[,c("classe", "roll_belt","roll_arm", "roll_dumbbell",
"roll_forearm", "pitch_belt", "pitch_arm", "pitch_dumbbell", "pitch_forearm", "yaw_belt", "yaw_a
rm", "yaw_dumbbell", "yaw_forearm","total_accel_arm", "total_accel_belt", "total_accel_dumbbell"
,"total_accel_forearm")],method="lm")

## Random Forest
  set.seed(200)
  control <- trainControl(method = "repeatedcv", number = 5, repeats = 3)
  rf_model <- train(classe ~ ., data = train[,c("classe", "roll_belt","roll_arm", "roll_dumbbel
l", "roll_forearm", "pitch_belt", "pitch_arm", "pitch_dumbbell", "pitch_forearm", "yaw_belt", "y
aw_arm", "yaw_dumbbell", "yaw_forearm","total_accel_arm", "total_accel_belt", "total_accel_dumbb
ell","total_accel_forearm")], method = "rf", ntree = 10, trControl = control, verbose=FALSE)
```

```
  trellis.par.set(caretTheme())
  plot(gbm_model)
```



```
summary(gbm_model$finalModel)
```




```
##                                   var   rel.inf
## roll_belt                   roll_belt 33.752094
## pitch_forearm           pitch_forearm 15.998537
## roll_forearm             roll_forearm 14.699880
## yaw_belt                     yaw_belt  8.675100
## pitch_belt                 pitch_belt  7.445905
## total_accel_belt     total_accel_belt  2.928083
## yaw_dumbbell             yaw_dumbbell  2.622590
## yaw_arm                       yaw_arm  2.502491
## roll_arm                     roll_arm  2.363111
## roll_dumbbell           roll_dumbbell  1.870073
## total_accel_arm       total_accel_arm  1.497838
## pitch_dumbbell         pitch_dumbbell  1.221761
## pitch_arm                   pitch_arm  1.187460
## total_accel_dumbbell total_accel_dumbbell  1.135138
## total_accel_forearm   total_accel_forearm  1.069536
## yaw_forearm                 yaw_forearm  1.030404
```

```
gbm_predict <- predict(gbm_model, testing_data)
gbm_prediction <- chartr("12345", "ABCDE", round(gbm_predict, digits=0))

summary(lm_model$finalModel)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7241 -0.9839 -0.0674  0.8922  3.7157
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           3.601e-01  7.944e-02   4.533 5.87e-06 ***
## roll_belt            -4.737e-03  1.216e-03  -3.896 9.82e-05 ***
## roll_arm              1.657e-03  2.010e-04   8.243  < 2e-16 ***
## roll_dumbbell         7.636e-04  2.172e-04   3.516  0.00044 ***
## roll_forearm          9.796e-04  1.145e-04   8.552  < 2e-16 ***
## pitch_belt           -1.392e-02  1.482e-03  -9.393  < 2e-16 ***
## pitch_arm            -7.721e-03  3.767e-04 -20.497  < 2e-16 ***
## pitch_dumbbell       -5.788e-04  4.293e-04  -1.348  0.17767
## pitch_forearm         1.855e-02  4.633e-04  40.032  < 2e-16 ***
## yaw_belt             -6.957e-03  5.272e-04 -13.195  < 2e-16 ***
## yaw_arm               7.767e-04  1.752e-04   4.433 9.36e-06 ***
## yaw_dumbbell         -2.753e-03  2.288e-04 -12.031  < 2e-16 ***
## yaw_forearm          -8.186e-04  1.291e-04  -6.343 2.33e-10 ***
## total_accel_arm      -5.045e-03  1.131e-03  -4.461 8.21e-06 ***
## total_accel_belt      1.071e-01  8.423e-03  12.712  < 2e-16 ***
## total_accel_dumbbell  7.404e-05  1.537e-03   0.048  0.96158
## total_accel_forearm   3.604e-02  1.233e-03  29.225  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.299 on 13720 degrees of freedom
## Multiple R-squared:  0.2293, Adjusted R-squared:  0.2284
## F-statistic: 255.2 on 16 and 13720 DF,  p-value: < 2.2e-16
```

```
  lm_predict <- predict(lm_model,testing_data)
  lm_prediction <- chartr("12345", "ABCDE", round(lm_predict, digits=0))

  summary(rf_model$finalModel)
```

```
##                 Length Class      Mode
## call               6  -none-     call
## type               1  -none-     character
## predicted      13737  -none-     numeric
## mse               10  -none-     numeric
## rsq               10  -none-     numeric
## oob.times      13737  -none-     numeric
## importance        16  -none-     numeric
## importanceSD       0  -none-     NULL
## localImportance    0  -none-     NULL
## proximity          0  -none-     NULL
## ntree              1  -none-     numeric
## mtry               1  -none-     numeric
## forest            11  -none-     list
## coefs              0  -none-     NULL
## y              13737  -none-     numeric
## test               0  -none-     NULL
## inbag              0  -none-     NULL
## xNames            16  -none-     character
## problemType        1  -none-     character
## tuneValue          1  data.frame list
## obsLevels          1  -none-     logical
## param              2  -none-     list
```

```
  rf_predict <-  predict(rf_model,testing_data)
  rf_prediction <- chartr("12345", "ABCDE", round(rf_predict, digits=0))

  ## Comparisons
  conf.matrix <- round(prop.table(table(lm_prediction, gbm_prediction, rf_prediction), 3), 3)
  conf.matrix
```

```
## , , rf_prediction = A
##
##              gbm_prediction
## lm_prediction     A     B     C     D     E
##             A 0.143 0.000 0.000 0.000 0.000
##             B 0.286 0.286 0.000 0.000 0.000
##             C 0.143 0.143 0.000 0.000 0.000
##             D 0.000 0.000 0.000 0.000 0.000
##
## , , rf_prediction = B
##
##              gbm_prediction
## lm_prediction     A     B     C     D     E
##             A 0.000 0.000 0.000 0.000 0.000
##             B 0.000 0.250 0.000 0.000 0.000
##             C 0.000 0.125 0.375 0.000 0.000
##             D 0.000 0.000 0.250 0.000 0.000
##
## , , rf_prediction = C
##
##              gbm_prediction
## lm_prediction     A     B     C     D     E
##             A 0.000 0.000 0.000 0.000 0.000
##             B 0.000 0.000 0.000 0.000 0.000
##             C 0.000 0.000 1.000 0.000 0.000
##             D 0.000 0.000 0.000 0.000 0.000
##
## , , rf_prediction = D
##
##              gbm_prediction
## lm_prediction     A     B     C     D     E
##             A 0.000 0.000 0.000 0.000 0.000
##             B 0.000 0.000 0.000 0.000 0.000
##             C 0.000 0.000 0.000 1.000 0.000
##             D 0.000 0.000 0.000 0.000 0.000
##
## , , rf_prediction = E
##
##              gbm_prediction
## lm_prediction     A     B     C     D     E
##             A 0.000 0.000 0.000 0.000 0.000
##             B 0.000 0.000 0.333 0.000 0.000
##             C 0.000 0.000 0.000 0.333 0.333
##             D 0.000 0.000 0.000 0.000 0.000
```

CONCLUSIONS

The predicted cases for all three models is found below. The random forest model performed well and was used to predict the 20 cases for the quiz.

```
  prediction_results <- cbind(gbm_prediction, lm_prediction, rf_prediction)
  kable(prediction_results) %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

gbm_prediction	lm_prediction	rf_prediction
C	D	B
A	C	A
B	B	B
B	B	A
B	B	A
E	C	E
D	C	D
C	D	B
A	A	A
B	C	A
C	C	B
C	C	C
C	C	B
A	B	A
D	C	E
C	B	E
A	B	A
C	C	B
B	C	B
B	B	B