Practical Machine Learning - Write Up

Nicholas Wright

29 September 2017

Summary

Random forest analysis was carried out on the training set to predict the ‘classe’ variable in the test set.

```
#The following code brings in the relevant packages and reduces the 'training' dataset by rem
oving non-numeric data and variables which are unlikely to be relevant.

setwd("~/R/R - Practical Machine Learning")
library(caret)
```

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

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

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
originaltraining = read.csv("pml-training.csv", na.strings=c("", "NA", "NULL"))
originaltesting = read.csv("pml-testing.csv", na.strings=c("", "NA", "NULL"))
dim(originaltraining)
```

```
## [1] 19622   160
```

```
dim(originaltesting)
```

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

```
training2 <- originaltraining[ , colSums(is.na(originaltraining)) == 0]
dim(training2)
```

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

```
head(training2)
```

```
##   X user_name raw_timestamp_part_1 raw_timestamp_part_2   cvtd_timestamp
## 1 1  carlitos           1323084231               788290 05/12/2011 11:23
## 2 2  carlitos           1323084231               808298 05/12/2011 11:23
## 3 3  carlitos           1323084231               820366 05/12/2011 11:23
## 4 4  carlitos           1323084232               120339 05/12/2011 11:23
## 5 5  carlitos           1323084232               196328 05/12/2011 11:23
## 6 6  carlitos           1323084232               304277 05/12/2011 11:23
##   new_window num_window roll_belt pitch_belt yaw_belt total_accel_belt
## 1         no         11      1.41       8.07    -94.4                3
## 2         no         11      1.41       8.07    -94.4                3
## 3         no         11      1.42       8.07    -94.4                3
## 4         no         12      1.48       8.05    -94.4                3
## 5         no         12      1.48       8.07    -94.4                3
## 6         no         12      1.45       8.06    -94.4                3
##   gyros_belt_x gyros_belt_y gyros_belt_z accel_belt_x accel_belt_y
## 1         0.00         0.00        -0.02          -21            4
## 2         0.02         0.00        -0.02          -22            4
## 3         0.00         0.00        -0.02          -20            5
## 4         0.02         0.00        -0.03          -22            3
## 5         0.02         0.02        -0.02          -21            2
## 6         0.02         0.00        -0.02          -21            4
##   accel_belt_z magnet_belt_x magnet_belt_y magnet_belt_z roll_arm
## 1           22            -3           599          -313     -128
## 2           22            -7           608          -311     -128
## 3           23            -2           600          -305     -128
## 4           21            -6           604          -310     -128
## 5           24            -6           600          -302     -128
## 6           21             0           603          -312     -128
##   pitch_arm yaw_arm total_accel_arm gyros_arm_x gyros_arm_y gyros_arm_z
## 1      22.5    -161              34        0.00        0.00       -0.02
## 2      22.5    -161              34        0.02       -0.02       -0.02
## 3      22.5    -161              34        0.02       -0.02       -0.02
## 4      22.1    -161              34        0.02       -0.03        0.02
## 5      22.1    -161              34        0.00       -0.03        0.00
## 6      22.0    -161              34        0.02       -0.03        0.00
##   accel_arm_x accel_arm_y accel_arm_z magnet_arm_x magnet_arm_y
## 1        -288         109        -123         -368          337
## 2        -290         110        -125         -369          337
## 3        -289         110        -126         -368          344
## 4        -289         111        -123         -372          344
## 5        -289         111        -123         -374          337
## 6        -289         111        -122         -369          342
##   magnet_arm_z roll_dumbbell pitch_dumbbell yaw_dumbbell
## 1          516      13.05217      -70.49400    -84.87394
## 2          513      13.13074      -70.63751    -84.71065
## 3          513      12.85075      -70.27812    -85.14078
## 4          512      13.43120      -70.39379    -84.87363
## 5          506      13.37872      -70.42856    -84.85306
## 6          513      13.38246      -70.81759    -84.46500
##   total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y gyros_dumbbell_z
## 1                   37                0            -0.02             0.00
## 2                   37                0            -0.02             0.00
## 3                   37                0            -0.02             0.00
## 4                   37                0            -0.02            -0.02
## 5                   37                0            -0.02             0.00
## 6                   37                0            -0.02             0.00
##   accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z magnet_dumbbell_x
```

```
## 1             -234               47             -271             -559
## 2             -233               47             -269             -555
## 3             -232               46             -270             -561
## 4             -232               48             -269             -552
## 5             -233               48             -270             -554
## 6             -234               48             -269             -558
##   magnet_dumbbell_y magnet_dumbbell_z roll_forearm pitch_forearm
## 1               293               -65         28.4         -63.9
## 2               296               -64         28.3         -63.9
## 3               298               -63         28.3         -63.9
## 4               303               -60         28.1         -63.9
## 5               292               -68         28.0         -63.9
## 6               294               -66         27.9         -63.9
##   yaw_forearm total_accel_forearm gyros_forearm_x gyros_forearm_y
## 1        -153                  36            0.03            0.00
## 2        -153                  36            0.02            0.00
## 3        -152                  36            0.03           -0.02
## 4        -152                  36            0.02           -0.02
## 5        -152                  36            0.02            0.00
## 6        -152                  36            0.02           -0.02
##   gyros_forearm_z accel_forearm_x accel_forearm_y accel_forearm_z
## 1           -0.02             192             203            -215
## 2           -0.02             192             203            -216
## 3            0.00             196             204            -213
## 4            0.00             189             206            -214
## 5           -0.02             189             206            -214
## 6           -0.03             193             203            -215
##   magnet_forearm_x magnet_forearm_y magnet_forearm_z classe
## 1              -17              654              476      A
## 2              -18              661              473      A
## 3              -18              658              469      A
## 4              -16              658              469      A
## 5              -17              655              473      A
## 6               -9              660              478      A
```

```
filtered <- c('X', 'user_name', 'raw_timestamp_part_1', 'raw_timestamp_part_2', 'cvtd_timesta
mp', 'new_window', 'num_window')
training3 <- training2[, -which(names(training2) %in% filtered)]
dim(training3)
```

```
## [1] 19622    53
```

R Markdown

Inspection of the training set shows a number of variables that have very low variance (so are unlikely to be effective predictors for ‘classe’) and a number of correlated variables (that are unlikely to increase the predictive power of any model). These are removed.The training set is split 80:20 into two sets - one for building the model (80% of the data) and the other for validating the model.

```
#Variables with low variance and highly correlated variables are removed.
#The original training set of data is split 80:20 into a training set and a new testing set.
LowVariance= nearZeroVar(training3[sapply(training3, is.numeric)], saveMetrics = TRUE)
training4 = training3[,LowVariance[, 'nzv']==0]
dim(training4)
```

```
## [1] 19622    53
```

```
corrMatrix <- cor(na.omit(training4[sapply(training4, is.numeric)]))
dim(corrMatrix)
```

```
## [1] 52 52
```

```
removecor = findCorrelation(corrMatrix, cutoff = .90, verbose = TRUE)
```

```
## Compare row 10  and column  1 with corr  0.992
##   Means:  0.27 vs 0.168 so flagging column 10
## Compare row 1  and column  9 with corr  0.925
##   Means:  0.25 vs 0.164 so flagging column 1
## Compare row 9  and column  4 with corr  0.928
##   Means:  0.233 vs 0.161 so flagging column 9
## Compare row 8  and column  2 with corr  0.966
##   Means:  0.245 vs 0.157 so flagging column 8
## Compare row 19  and column  18 with corr  0.918
##   Means:  0.091 vs 0.158 so flagging column 18
## Compare row 46  and column  31 with corr  0.914
##   Means:  0.101 vs 0.161 so flagging column 31
## Compare row 46  and column  33 with corr  0.933
##   Means:  0.083 vs 0.164 so flagging column 33
## All correlations <= 0.9
```

```
training5 = training4[,-removecor]
dim(training5)
```

```
## [1] 19622    46
```

```
inTrain <- createDataPartition(y=training5$classe, p=0.8, list=FALSE)
trainingset <- training5[inTrain,]; validationset <- training5[-inTrain,]
dim(trainingset);dim(validationset)
```

```
## [1] 15699    46
```

```
## [1] 3923   46
```

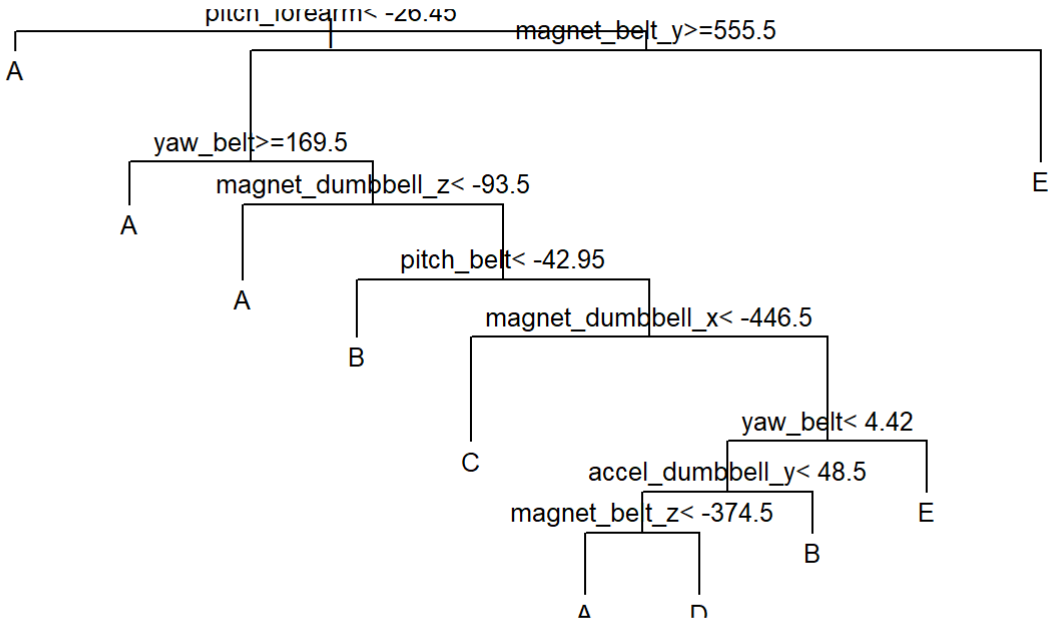
Caret

Using the ‘Caret’ package, a ‘decision tree’ model was constructed.

```
set.seed(22222)
model1 <- train(classe ~ .,method="rpart",data=trainingset)
print(model1$finalModel)
```

```
## n= 15699
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 15699 11235 A (0.28 0.19 0.17 0.16 0.18)
##     2) pitch_forearm< -26.45 1388    69 A (0.95 0.05 0 0 0) *
##     3) pitch_forearm>=-26.45 14311 11166 A (0.22 0.21 0.19 0.18 0.2)
##       6) magnet_belt_y>=555.5 13116  9973 A (0.24 0.23 0.21 0.18 0.15)
##        12) yaw_belt>=169.5 651    71 A (0.89 0.055 0 0.054 0) *
##        13) yaw_belt< 169.5 12465  9536 B (0.21 0.23 0.22 0.19 0.15)
##          26) magnet_dumbbell_z< -93.5 1548   631 A (0.59 0.28 0.047 0.055 0.03) *
##          27) magnet_dumbbell_z>=-93.5 10917  8253 C (0.15 0.23 0.24 0.2 0.17)
##            54) pitch_belt< -42.95 661   104 B (0.021 0.84 0.092 0.026 0.018) *
##            55) pitch_belt>=-42.95 10256  7653 C (0.16 0.19 0.25 0.22 0.18)
##             110) magnet_dumbbell_x< -446.5 5674  3490 C (0.15 0.11 0.38 0.21 0.14) *
##             111) magnet_dumbbell_x>=-446.5 4582  3261 B (0.17 0.29 0.091 0.22 0.24)
##               222) yaw_belt< 4.42 4116  2796 B (0.18 0.32 0.095 0.24 0.16)
##                 444) accel_dumbbell_y< 48.5 2428  1591 D (0.28 0.2 0.14 0.34 0.042)
##                   888) magnet_belt_z< -374.5 1003   422 A (0.58 0.22 0.056 0.051 0.094) *
##                   889) magnet_belt_z>=-374.5 1425   639 D (0.062 0.18 0.2 0.55 0.0056) *
##                 445) accel_dumbbell_y>=48.5 1688   851 B (0.052 0.5 0.033 0.097 0.32) *
##               223) yaw_belt>=4.42 466    28 E (0 0.0021 0.058 0 0.94) *
##       7) magnet_belt_y< 555.5 1195   227 E (0.0017 0.0033 0.0017 0.18 0.81) *
```

```
plot(model1$finalModel)
text(model1$finalModel,pretty=0, cex =.8)
```



```
#This builds a decision tree model, which we can use on the validation data.

prediction1=predict(model1,validationset)
prediction2 = with(validationset,table(prediction1,classe))
sum(diag(prediction2))/sum(as.vector(prediction2))
```

```
## [1] 0.5745603
```

```
#The last line calculates the error rate.
```

As we can see, this is a relatively large error rate.

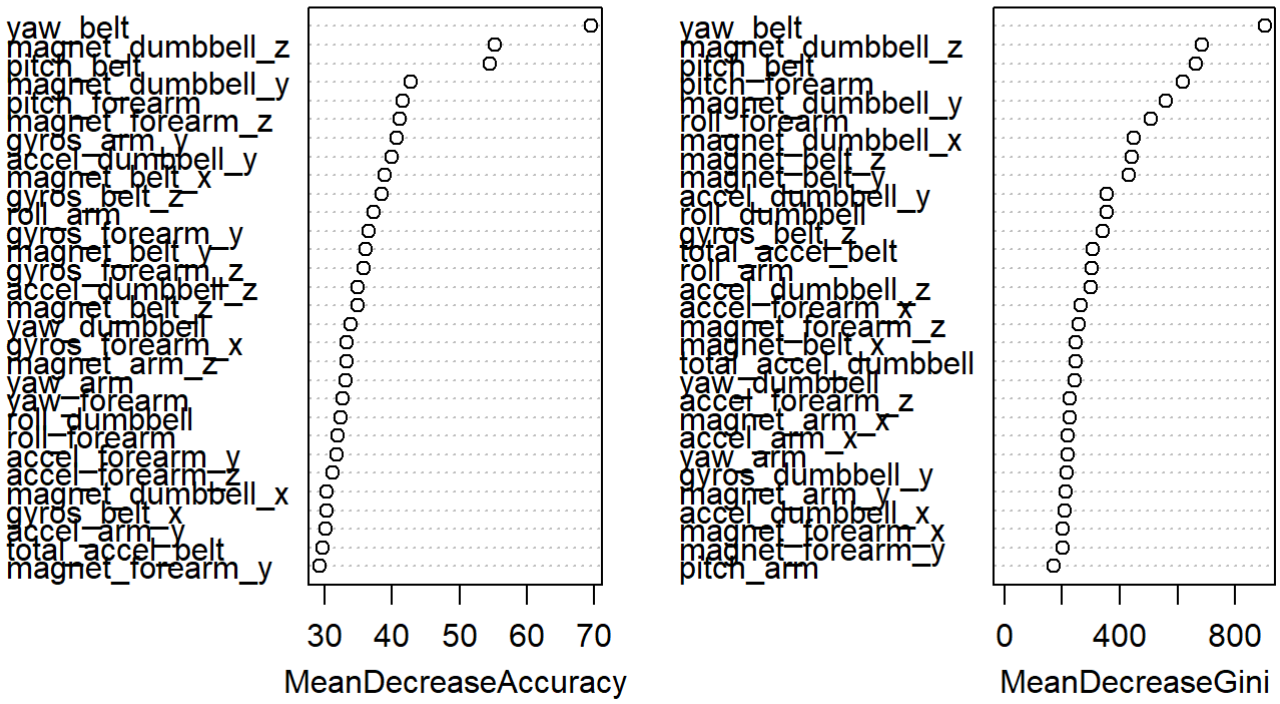
RandomForest

Using the 'RandomForest' package, a 'random forest' model was constructed.

```
set.seed(22222)
randomforest1=randomForest(classe~.,data=trainingset,ntree=500, importance=TRUE)
randomforest1
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trainingset, ntree = 500,      importance = TRU
E)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.48%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4459    4    0    0    1 0.001120072
## B   11 3020    7    0    0 0.005924951
## C    0   19 2718    1    0 0.007304602
## D    0    0   23 2547    3 0.010104936
## E    0    0    1    5 2880 0.002079002
```

```
varImpPlot(randomforest1,)
```




```
rfpredict=predict(randomforest1,validationset,type="class")
preddata = with(validationset,table(rfpredict,classe))
sum(diag(preddata))/sum(as.vector(preddata))
```

```
## [1] 0.9951568
```

```
#This builds a decision tree model, which we can use on the validation data.
```

As the error rate is much lower, we can use the random forest model on the original testing dataset.

```
newprediction <- predict(randomforest1, originaltesting)
newprediction
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```

This provides us with the predicted classes for each case (and the answers to the quiz!)