CourseRA- Practical Machine Learning

DFeron

December 1, 2017

Background

Background

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. In this project, your goal will be to use 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. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

Read Data

```
## Warning: package 'caret' was built under R version 3.4.2
```

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

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

```
## Warning: package 'rpart.plot' was built under R version 3.4.2
```

```
## Warning: package 'rattle' was built under R version 3.4.2
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
## Warning: package 'randomForest' was built under R version 3.4.2
```

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

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

```
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:rattle':
##
##     importance
```

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

```
## Warning: package 'corrplot' was built under R version 3.4.3
```

```
## corrplot 0.84 loaded
```

Loading Data and Cleaning Data

```
#Set Training Data Set to 70% of the data
in_train  <- createDataPartition(train$classe, p=0.7, list=FALSE)
train_data <- train[in_train, ]
test_data  <- train[-in_train, ]
dim(train_data)
```

```
## [1] 13737   160
```

```
dim(test_data)
```

```
## [1] 5885  160
```

```
#Remove variables w/ ~0 variance
near_zero <- nearZeroVar(train_data)
train_data <- train_data[, -near_zero]
test_data  <- test_data[, -near_zero]
dim(train_data)
```

```
## [1] 13737   106
```

```
dim(test_data)
```

```
## [1] 5885  106
```

```
#Remove variables that are more than 95% NA
mostly_NA    <- sapply(train_data, function(x) mean(is.na(x))) > 0.95
train_data <- train_data[, mostly_NA==FALSE]
test_data  <- test_data[, mostly_NA==FALSE]
dim(train_data)
```

```
## [1] 13737    59
```

```
dim(test_data)
```

```
## [1] 5885   59
```

```
#remove variables that are for identification only, not informative
train_data <- train_data[, -(1:5)]
test_data  <- test_data[, -(1:5)]
dim(train_data)
```

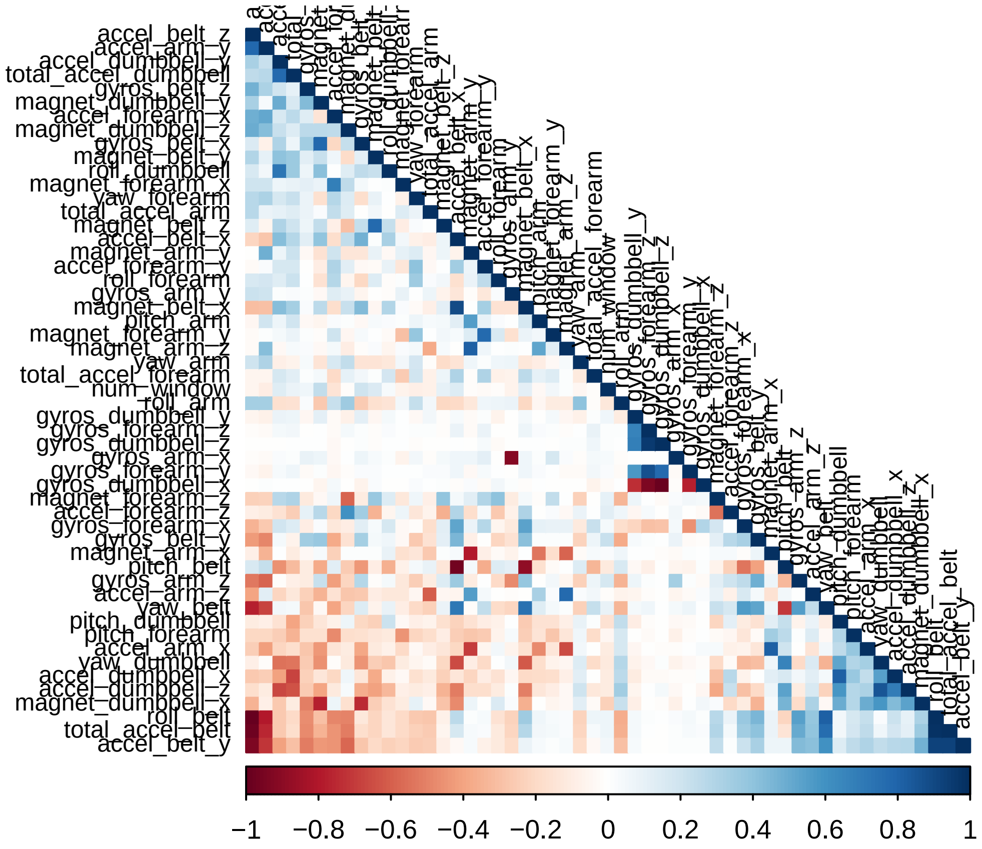
```
## [1] 13737    54
```

```
dim(test_data)
```

```
## [1] 5885   54
```

Correlation Analysis: performed before modeling

```
cor_matrix <- cor(train_data[, -54])
corrplot(cor_matrix, order = "FPC", method = "color", type = "lower",
         tl.cex = 0.75, tl.col = "black")
```



```
print("As the key indicates, there are only a few highly correlated variables in the data.  Therefore we
```

```
## [1] "As the key indicates, there are only a few highly correlated variables in the data.  Therefore w
```

Prediction Modeling- Random Forest

```
#Check model fit
set.seed(12345)
control_randfor <- trainControl(method="cv", number=3, verboseIter=FALSE)
modfit_rf <- train(classe ~ ., data=train_data, method="rf",
                          trControl=control_randfor)
modfit_rf$finalModel
```

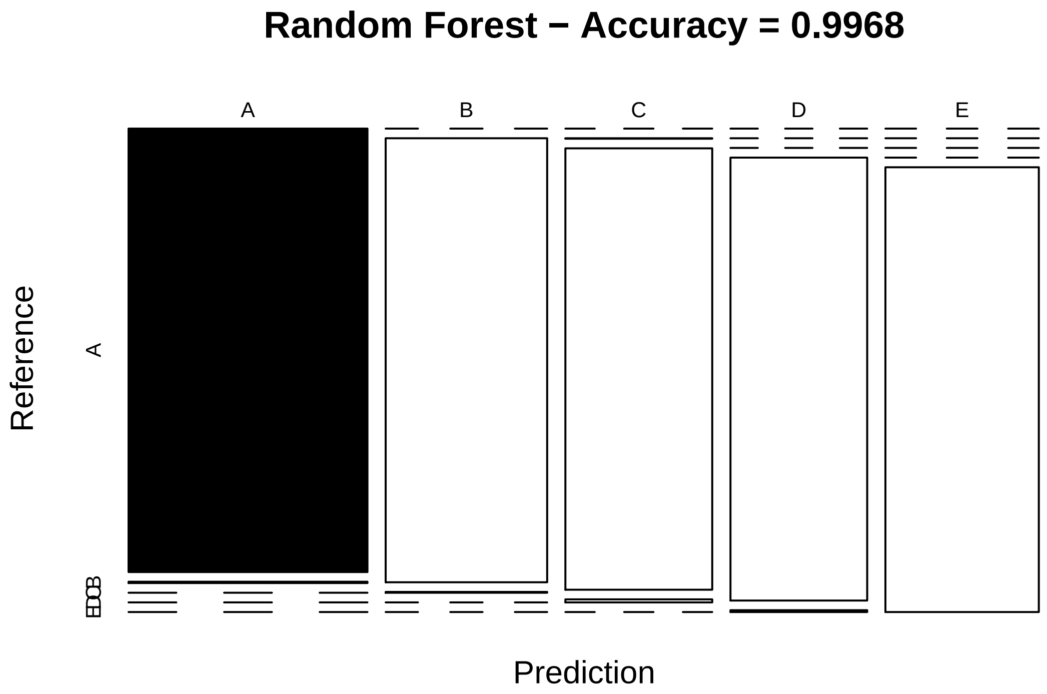
```
##
## Call:
```

```
##  randomForest(x = x, y = y, mtry = param$mtry)
##                 Type of random forest: classification
##                       Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.19%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    1    0    0    1 0.0005120328
## B    6 2651    1    0    0 0.0026335591
## C    0    6 2390    0    0 0.0025041736
## D    0    0    8 2244    0 0.0035523979
## E    0    0    0    3 2522 0.0011881188
```

```r
#Run prediction on Test data
predict_randfor <- predict(modfit_rf, newdata=test_data)
conf_mat_rf <- confusionMatrix(predict_randfor, test_data$classe)
conf_mat_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    2    0    0
##          C    0    1 1024    7    0
##          D    0    0    0  957    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9968
##                  95% CI : (0.995, 0.9981)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9959
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9981   0.9927   0.9963
## Specificity            0.9988   0.9996   0.9984   0.9992   1.0000
## Pos Pred Value         0.9970   0.9982   0.9922   0.9958   1.0000
## Neg Pred Value         1.0000   0.9987   0.9996   0.9986   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1740   0.1626   0.1832
## Detection Prevalence   0.2853   0.1929   0.1754   0.1633   0.1832
## Balanced Accuracy      0.9994   0.9972   0.9982   0.9960   0.9982
```

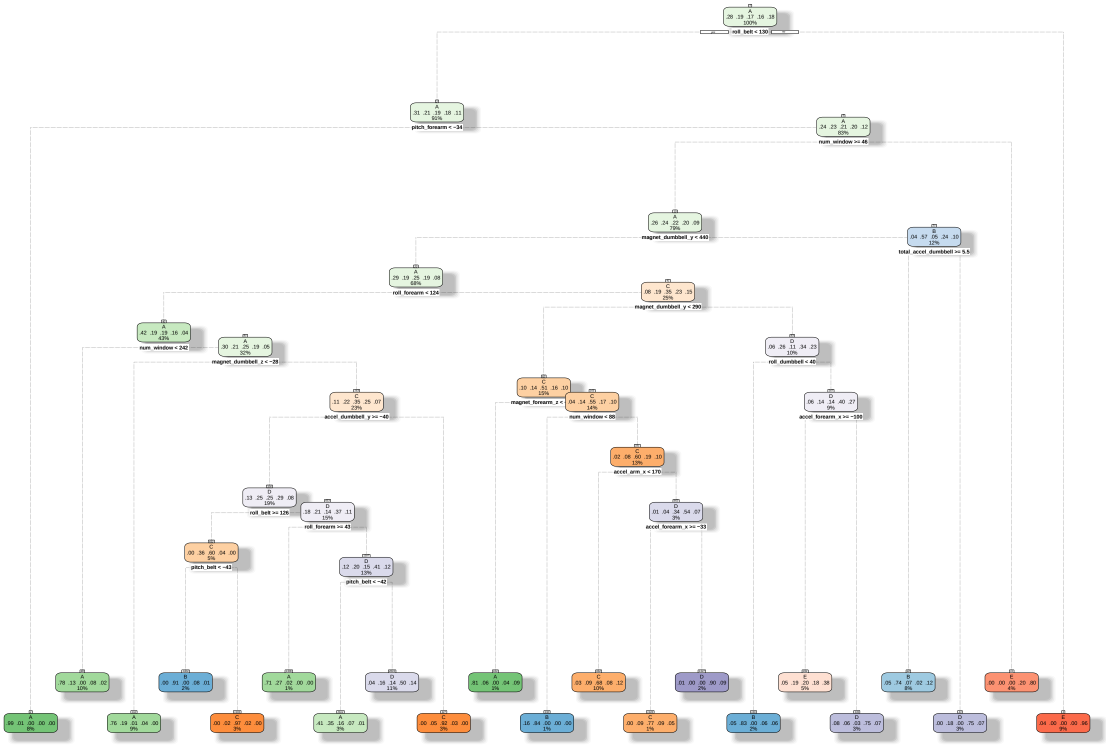
```r
#Plot results of test prediction
plot(conf_mat_rf$table, col = conf_mat_rf$byClass,
     main = paste("Random Forest - Accuracy =",
                  round(conf_mat_rf$overall['Accuracy'], 4)))
```



Prediction Modeling- Decision Trees

```
#Check model fit
set.seed(12345)
modfit_dt <- rpart(classe ~ ., data=train_data, method="class")
fancyRpartPlot(modfit_dt)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



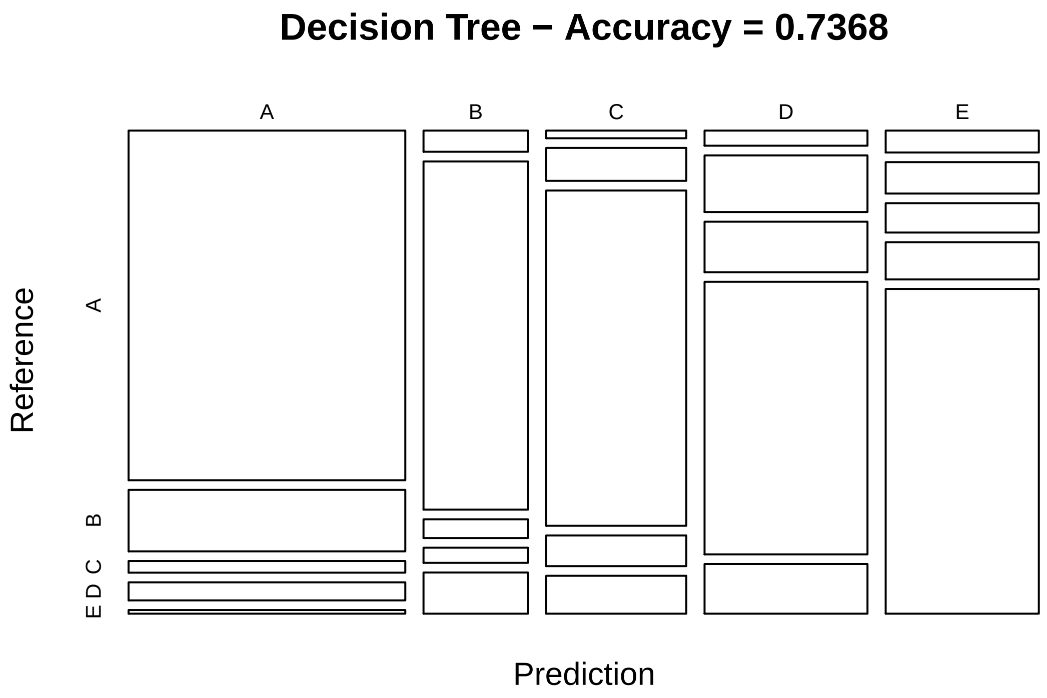
Rattle 2017−Dec−06 09:50:02 ferond

```
#Run prediction on Test data
predict_dt <- predict(modfit_dt, newdata=test_data, type="class")
conf_mat_dt <- confusionMatrix(predict_dt, test_data$classe)
conf_mat_dt
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1530  269   51   79   16
##          B   35  575   31   25   68
##          C   17   73  743   68   84
##          D   39  146  130  702  128
##          E   53   76   71   90  786
##
## Overall Statistics
##
##                Accuracy : 0.7368
##                  95% CI : (0.7253, 0.748)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6656
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
##
##                        Class: A Class: B Class: C Class: D Class: E
## Sensitivity              0.9140  0.50483   0.7242   0.7282   0.7264
## Specificity              0.9014  0.96650   0.9502   0.9100   0.9396
## Pos Pred Value           0.7866  0.78338   0.7543   0.6131   0.7305
## Neg Pred Value           0.9635  0.89051   0.9422   0.9447   0.9384
## Prevalence               0.2845  0.19354   0.1743   0.1638   0.1839
## Detection Rate           0.2600  0.09771   0.1263   0.1193   0.1336
## Detection Prevalence     0.3305  0.12472   0.1674   0.1946   0.1828
## Balanced Accuracy        0.9077  0.73566   0.8372   0.8191   0.8330
```

```
plot(conf_mat_dt$table, col = conf_mat_dt$byClass,
     main = paste("Decision Tree - Accuracy =",
                  round(conf_mat_dt$overall['Accuracy'], 4)))
```



Summary

```
## [1] "Of the two modeling techniques used above (Random Forest, Decision Tree), the accuracy of each
```

Predict Test

```
predict_test <- predict(modfit_rf, newdata=test)
predict_test
```

```
##  [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
```