ML\_Project

Swetha Thagella

March 7, 2018

## Project Goal

The goal of the project is to predict the manner in which the 6 participants did the exercise which is identified by the “classe” variable in the training set.

# Input data and initial exploration

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rattle)

## 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.

training <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"))  
test <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"))  
#colnames(training) - not printing the result

There are 159 variables in the dataset excluding the independent variable “classe”. Based on the cursory look, not all variables would be useful in predicting the “classe”. The following steps involve cleaning the data in order to only retain relevant variables for the prediction models.

# Cleaning data

Removing NAs from training and test data sets

trn <- training[, colSums(is.na(training)) == 0]  
tst <- test[, colSums(is.na(test)) == 0]

Removing Zero Covariates so only relevant variables/predictors can be used

newtraining <- nearZeroVar(trn, saveMetrics = TRUE)  
nztraining <- trn[, newtraining$nzv == FALSE]

# Crossvalidation

Splitting the training data into training and test so that the model that is developed can be tested prior to testing it with the final testing data set

subtrain <- createDataPartition(y=nztraining$classe, p = 0.7, list = FALSE)  
train1 <- nztraining[subtrain, ] [c(-1,-2,-3,-4,-5)]  
test1 <- nztraining[-subtrain, ] [c(-1,-2,-3,-4,-5)]

# Algorithm Selection

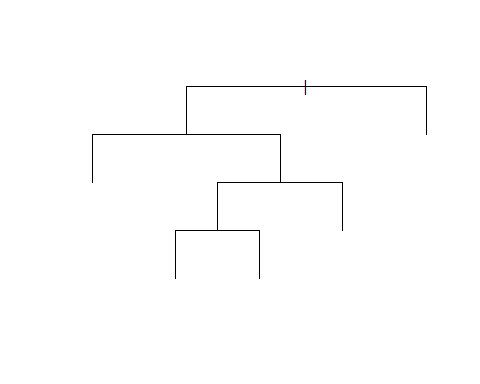
The data definitely seems to contain non-linear relationships. Since prediction with trees is more robust for regression as well as non-linear variable relationships, I tried the following three algorithms to evaluate the accuracy of each and then choose the best one.

# 1. Decision Trees Model (rpart)

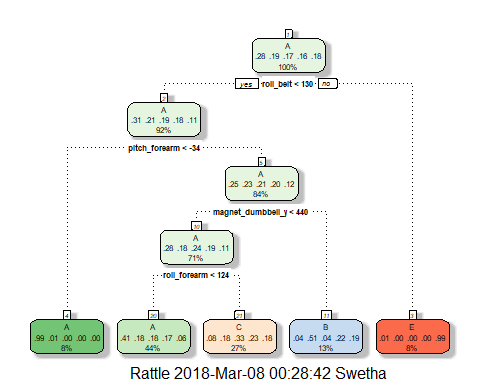
set.seed(100)  
modfit\_rp <- train(classe ~ ., data = train1, method = "rpart")  
print(modfit\_rp$finalModel)

## n= 13737   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 13737 9831 A (0.28 0.19 0.17 0.16 0.18)   
## 2) roll\_belt< 130.5 12573 8676 A (0.31 0.21 0.19 0.18 0.11)   
## 4) pitch\_forearm< -33.65 1091 8 A (0.99 0.0073 0 0 0) \*  
## 5) pitch\_forearm>=-33.65 11482 8668 A (0.25 0.23 0.21 0.2 0.12)   
## 10) magnet\_dumbbell\_y< 439.5 9711 6959 A (0.28 0.18 0.24 0.19 0.11)   
## 20) roll\_forearm< 123.5 6033 3566 A (0.41 0.18 0.18 0.17 0.061) \*  
## 21) roll\_forearm>=123.5 3678 2459 C (0.077 0.18 0.33 0.23 0.18) \*  
## 11) magnet\_dumbbell\_y>=439.5 1771 862 B (0.035 0.51 0.04 0.22 0.19) \*  
## 3) roll\_belt>=130.5 1164 9 E (0.0077 0 0 0 0.99) \*

plot(modfit\_rp$finalModel, uniform = TRUE)



rattle::fancyRpartPlot(modfit\_rp$finalModel)



pred\_rp <- predict(modfit\_rp, newdata = test1)  
conmatrix\_rp <- confusionMatrix(pred\_rp, test1$classe)  
conmatrix\_rp

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1532 497 481 435 159  
## B 19 377 37 178 147  
## C 118 265 508 351 300  
## D 0 0 0 0 0  
## E 5 0 0 0 476  
##   
## Overall Statistics  
##   
## Accuracy : 0.4916   
## 95% CI : (0.4787, 0.5044)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.3348   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9152 0.33099 0.49513 0.0000 0.43993  
## Specificity 0.6267 0.91972 0.78720 1.0000 0.99896  
## Pos Pred Value 0.4936 0.49736 0.32944 NaN 0.98960  
## Neg Pred Value 0.9489 0.85138 0.88073 0.8362 0.88786  
## Prevalence 0.2845 0.19354 0.17434 0.1638 0.18386  
## Detection Rate 0.2603 0.06406 0.08632 0.0000 0.08088  
## Detection Prevalence 0.5274 0.12880 0.26202 0.0000 0.08173  
## Balanced Accuracy 0.7709 0.62536 0.64116 0.5000 0.71944

# 2. Random Forest Model

This includes 10 fold cross validation (as higher as per literature was not anymore significant)

set.seed(100)  
modfit\_rf <- train(classe ~ ., data = train1, method = "rf", trControl = trainControl(method = "cv", 10), ntree = 100)  
pred\_rf <- predict(modfit\_rf, newdata = test1)  
conmatrix\_rf <- confusionMatrix(pred\_rf, test1$classe)  
conmatrix\_rf

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1674 0 0 0 0  
## B 0 1139 2 0 0  
## C 0 0 1021 4 0  
## D 0 0 3 960 1  
## E 0 0 0 0 1081  
##   
## Overall Statistics  
##   
## Accuracy : 0.9983   
## 95% CI : (0.9969, 0.9992)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9979   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.0000 1.0000 0.9951 0.9959 0.9991  
## Specificity 1.0000 0.9996 0.9992 0.9992 1.0000  
## Pos Pred Value 1.0000 0.9982 0.9961 0.9959 1.0000  
## Neg Pred Value 1.0000 1.0000 0.9990 0.9992 0.9998  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2845 0.1935 0.1735 0.1631 0.1837  
## Detection Prevalence 0.2845 0.1939 0.1742 0.1638 0.1837  
## Balanced Accuracy 1.0000 0.9998 0.9972 0.9975 0.9995

# 3. Gradient Boosting Model including cross validation

set.seed(100)  
modfit\_gbm <- train(classe ~ ., data = train1, method = "gbm", verbose = FALSE, trControl = trainControl(method = "repeatedcv", 10, repeats = 1))  
print(modfit\_gbm$finalModel)

## A gradient boosted model with multinomial loss function.  
## 150 iterations were performed.  
## There were 53 predictors of which 45 had non-zero influence.

pred\_gbm <- predict(modfit\_gbm, newdata = test1)  
conmatrix\_gbm <- confusionMatrix(pred\_gbm, test1$classe)  
conmatrix\_gbm

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1671 12 0 0 0  
## B 3 1124 17 7 9  
## C 0 3 1008 13 3  
## D 0 0 1 944 15  
## E 0 0 0 0 1055  
##   
## Overall Statistics  
##   
## Accuracy : 0.9859   
## 95% CI : (0.9825, 0.9888)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9822   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9982 0.9868 0.9825 0.9793 0.9750  
## Specificity 0.9972 0.9924 0.9961 0.9967 1.0000  
## Pos Pred Value 0.9929 0.9690 0.9815 0.9833 1.0000  
## Neg Pred Value 0.9993 0.9968 0.9963 0.9959 0.9944  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2839 0.1910 0.1713 0.1604 0.1793  
## Detection Prevalence 0.2860 0.1971 0.1745 0.1631 0.1793  
## Balanced Accuracy 0.9977 0.9896 0.9893 0.9880 0.9875

# Final Prediction

Based on the highest prediction Accuracy of 0.9969, I chose to apply the Random Forest algorithm to the Test data set.The Accuracy of the Decision Tree Model was 0.4975 while the GBM model had an accuracy of 0.9869.The out of sample error of the Random Forest model is estimated to be 0.31% (1-0.9969).

pred\_rf\_test <- predict(modfit\_rf, newdata=tst)  
pred\_rf\_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