Peer review assignment Practical machine learning

The goal of your project is to predict the manner in which they did the exercise. This is the “classe” variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

# Get the data

# library packages

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(randomForest)

## randomForest 4.6-14

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

##   
## Attaching package: 'randomForest'

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

library(rpart)   
library(rpart.plot)  
library(RColorBrewer)  
library(rattle)

## Rattle: A free graphical interface for data science with R.  
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

##   
## Attaching package: 'rattle'

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

library(ggplot2)

# exploratory analysis

#dim(training); dim(testing); summary(training); summary(testing); str(training); str(testing); head(training); head(testing)

# Clean the data

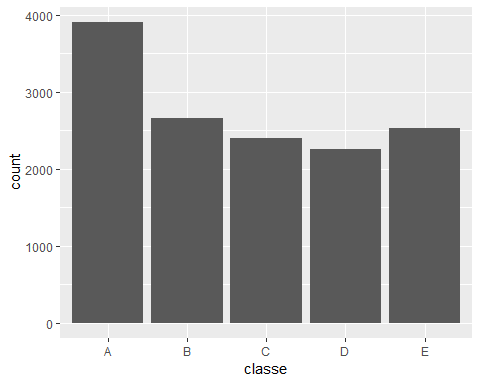
## Devide the training set into two

with a 70/30 chance

# let’s have a look at classe

ggplot(myTraining, aes(classe)) +  
 geom\_histogram(stat="count", position = "dodge")

## Warning: Ignoring unknown parameters: binwidth, bins, pad

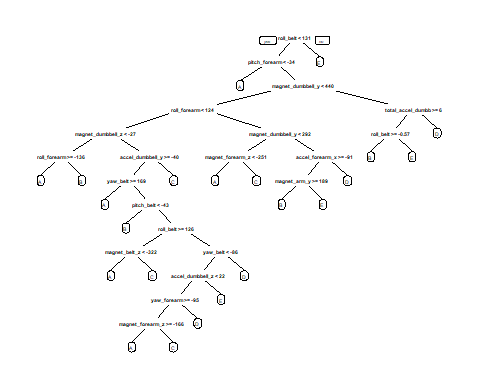


In the plot we can see that the level A has the most counts and level D the least.

# Decision Tree

We fit a predictive model for activity recognition using Decision Tree algorithm.

model1 <- rpart(classe ~ ., data=myTraining, method="class")  
prp(model1)



Now, we estimate the performance of the model on the testing data set.

# Predicting dt:  
predictiontree <- predict(model1, myTesting, type = "class")  
confusionMatrix(myTesting$classe, predictiontree)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1492 48 38 59 37  
## B 191 605 152 80 111  
## C 21 78 833 62 32  
## D 47 88 129 614 86  
## E 22 94 128 60 778  
##   
## Overall Statistics  
##   
## Accuracy : 0.7344   
## 95% CI : (0.7229, 0.7457)  
## No Information Rate : 0.3013   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6635   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.8415 0.6627 0.6508 0.7017 0.7452  
## Specificity 0.9557 0.8926 0.9581 0.9301 0.9372  
## Pos Pred Value 0.8913 0.5312 0.8119 0.6369 0.7190  
## Neg Pred Value 0.9333 0.9351 0.9080 0.9470 0.9446  
## Prevalence 0.3013 0.1551 0.2175 0.1487 0.1774  
## Detection Rate 0.2535 0.1028 0.1415 0.1043 0.1322  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.8986 0.7776 0.8044 0.8159 0.8412

accuracy1 <- postResample(predictiontree, myTesting$classe)  
ose1 <- 1 - as.numeric(confusionMatrix(myTesting$classe, predictiontree)$overall[1])

The Accuracy of the prediction tree is 72.9%, not terribly good. And the Estimated Out-of-Sample Error 27.1%. Let’s have a look at a random forest prediction.

## Random forest prediction

Now we will fit a predictive model for classe using Random Forest algorithm. It automatically selects important variables and is robust to correlated covariates & outliers in general. We will use 5-fold cross validation when applying the algorithm.

#fitting  
Randomfrst <- train(classe ~ ., data = myTraining, method = "rf", trControl = trainControl(method = "cv", 5), ntree = 250)  
Randomfrst

## Random Forest   
##   
## 13737 samples  
## 52 predictor  
## 5 classes: 'A', 'B', 'C', 'D', 'E'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 10989, 10989, 10989, 10990, 10991   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 2 0.9909730 0.9885802  
## 27 0.9898085 0.9871059  
## 52 0.9807090 0.9755901  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 2.

Now, we estimate the performance of the model on the testing set.

# Predicting rf:  
predictRF <- predict(Randomfrst, myTesting)  
confusionMatrix(myTesting$classe, predictRF)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1673 1 0 0 0  
## B 8 1131 0 0 0  
## C 0 9 1015 2 0  
## D 0 0 18 946 0  
## E 0 0 0 1 1081  
##   
## Overall Statistics  
##   
## Accuracy : 0.9934   
## 95% CI : (0.991, 0.9953)  
## No Information Rate : 0.2856   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9916   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9952 0.9912 0.9826 0.9968 1.0000  
## Specificity 0.9998 0.9983 0.9977 0.9964 0.9998  
## Pos Pred Value 0.9994 0.9930 0.9893 0.9813 0.9991  
## Neg Pred Value 0.9981 0.9979 0.9963 0.9994 1.0000  
## Prevalence 0.2856 0.1939 0.1755 0.1613 0.1837  
## Detection Rate 0.2843 0.1922 0.1725 0.1607 0.1837  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.9975 0.9948 0.9902 0.9966 0.9999

accuracy2 <- postResample(predictRF, myTesting$classe)  
ose2 <- 1 - as.numeric(confusionMatrix(myTesting$classe, predictRF)$overall[1])

The Accuracy of the prediction tree is 99.18%, not terribly good. And the Estimated Out-of-Sample Error 0.8%. As expected the random forest technique yielded much beter results.

# Predicting the testing data set

predict(Randomfrst, testing[, -length(names(testing))])

## [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

This shows the predicted outcomes based on our fitted random forest model. With an accuracy above 99% on our cross-validation data, we can expect that very few, or none, of the test samples will be missclassified.