PMLProjectWriteUp

Nasir Jamal Khan

Saturday, January 24, 2015

Description:

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.
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).

Objective:

The goal of your project is to:
- Predict the manner in which they did the exercise. - Create a report describing how you built your model, how you used cross validation, You may use variable "classe" or any of the other variables to predict with, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did

Data Source:

The training and test data for this project are available here:

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

The data for this project come from this source: link

Processing and Analysis

Data Loading:

The source training and testing datasets from above links were downloaded and loaded into respective data frame as following.

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

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

Data Cleaning:

The both datasets have identical 160 variables. There is a high number of variables which are either irrelevent or not valid i.e. high percent of values such as **NA**. Also the first seven variables are removed as those are not revelent for preparing the predicting model.

```
set.seed(78662)     # A random seed number which will be used throughout this project

trainingCols <- colnames(training)
CountbyColumns <- as.vector(apply(training, 2,
                            function(training) length(which(!is.na(training)))))
Var4removal <- c()
for (cnt in 1:length(CountbyColumns)) {
    if (CountbyColumns[cnt] < nrow(training)) {
        Var4removal <- c(Var4removal, trainingCols[cnt])
    }
}
Cleantraining <- training[,!(names(training) %in% Var4removal)]
Cleantraining <- Cleantraining[,8:length(colnames(Cleantraining))]
#
Cleantesting <- testing[,!(names(testing) %in% Var4removal)]
testing <- Cleantesting[,8:length(colnames(Cleantesting))]
dim(Cleantraining); dim(testing)
```

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

```
## [1] 20 53
```

Partitioning of Training Data for processing and cross validating:

```
inTrain <- createDataPartition(y=Cleantraining$classe, p=0.6, list=FALSE)
newtraining <- Cleantraining[inTrain, ]; validation <- Cleantraining[-inTrain, ]
dim(newtraining); dim(validation)
```

```
## [1] 11776    53
```

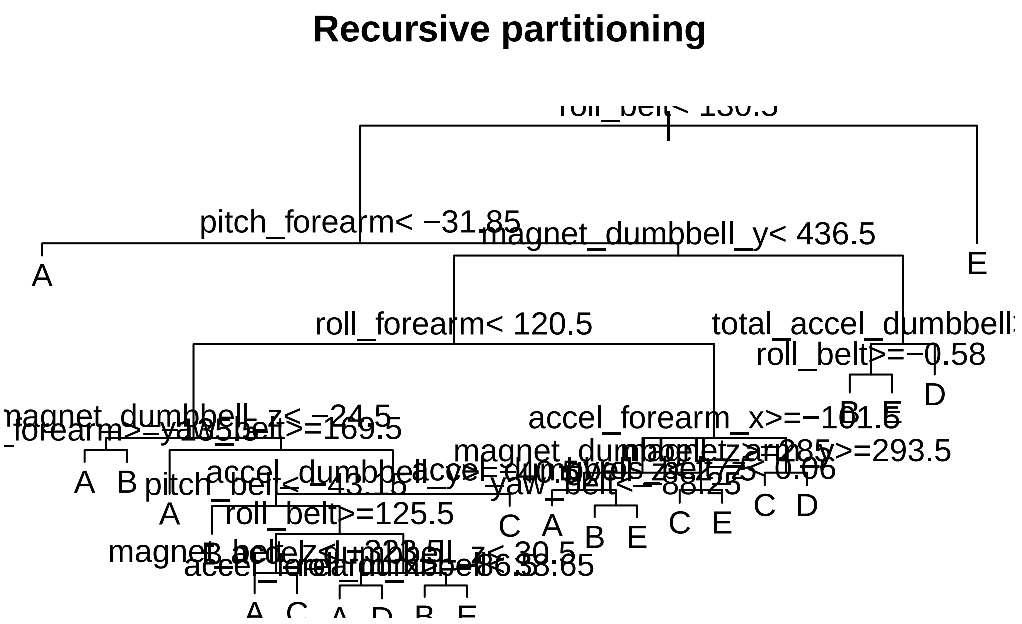
```
## [1] 7846   53
```

Model fitting using rpart:

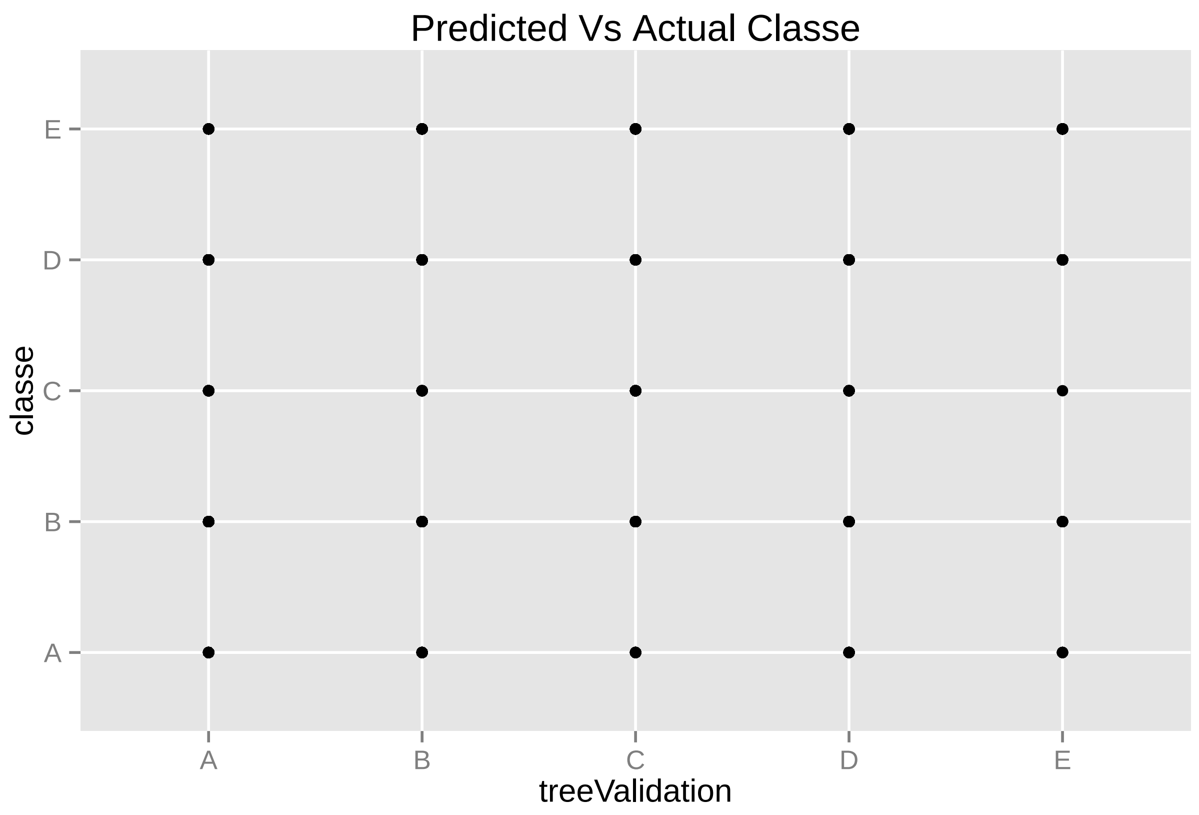
```
set.seed(78662)
library(rpart.plot)
```

```
## Loading required package: rpart
```

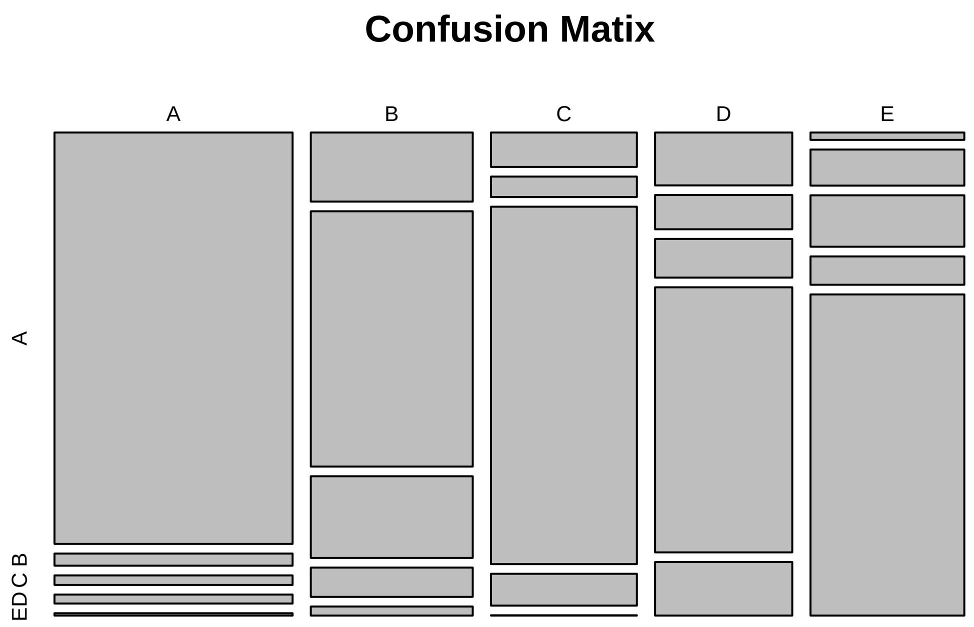
```
tree1 <- rpart(validation$classe ~ ., data = validation)
plot(tree1,main="Recursive partitioning")
text(tree1)
```



```
treeValidation <- predict(tree1, validation, type = "class")
qplot(treeValidation,classe,data=validation, main="Predicted Vs Actual Classe")
```



```
CMat<-confusionMatrix(treeValidation, validation$classe)
ctable <- as.table(matrix(c(CMat$table), nrow = 5, byrow = TRUE))
#plot(ctable)
plot(ctable, type = "barplot",main="Confusion Matix")
```



```r
print(CMat)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2066  236  106  153   24
##          B   61  872   63   99  117
##          C   48  279 1100  112  167
##          D   45  100   98  767   92
##          E   12   31    1  155 1042
##
## Overall Statistics
##
##                Accuracy : 0.7452
##                  95% CI : (0.7354, 0.7548)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.676
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9256   0.5744   0.8041  0.59642   0.7226
## Specificity            0.9076   0.9463   0.9065  0.94893   0.9689
## Pos Pred Value         0.7992   0.7195   0.6448  0.69601   0.8396
## Neg Pred Value         0.9684   0.9026   0.9564  0.92304   0.9394
## Prevalence             0.2845   0.1935   0.1744  0.16391   0.1838
## Detection Rate         0.2633   0.1111   0.1402  0.09776   0.1328
## Detection Prevalence   0.3295   0.1545   0.2174  0.14045   0.1582
## Balanced Accuracy      0.9166   0.7604   0.8553  0.77268   0.8458
```

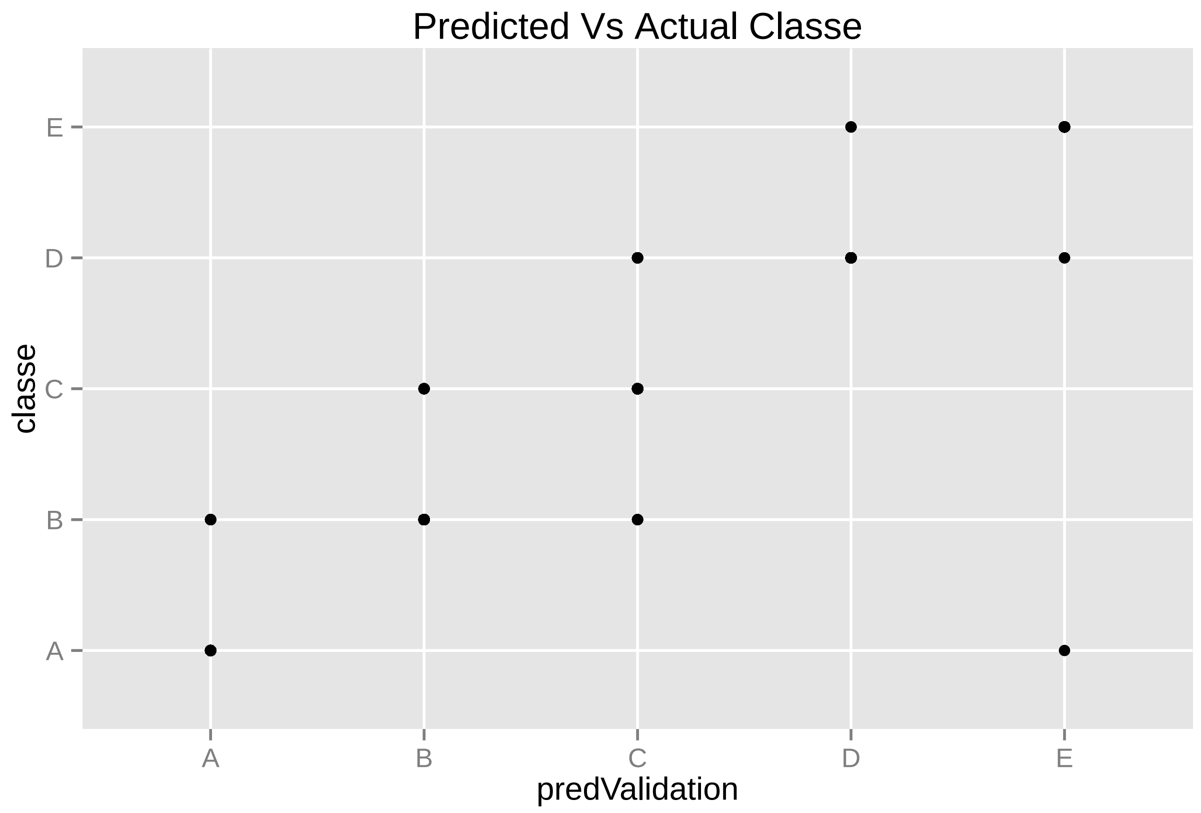
Result:

Based on the above following results from predicted model and confusion matrix data we conclude that this model is not accurate and fit enough for testing against **testing** dataset.

- Accuracy : 0.7452
- 95% CI : (0.7354, 0.7548)
- Kappa : 0.676

Model fitting using randomForest method and preparing prediction data:

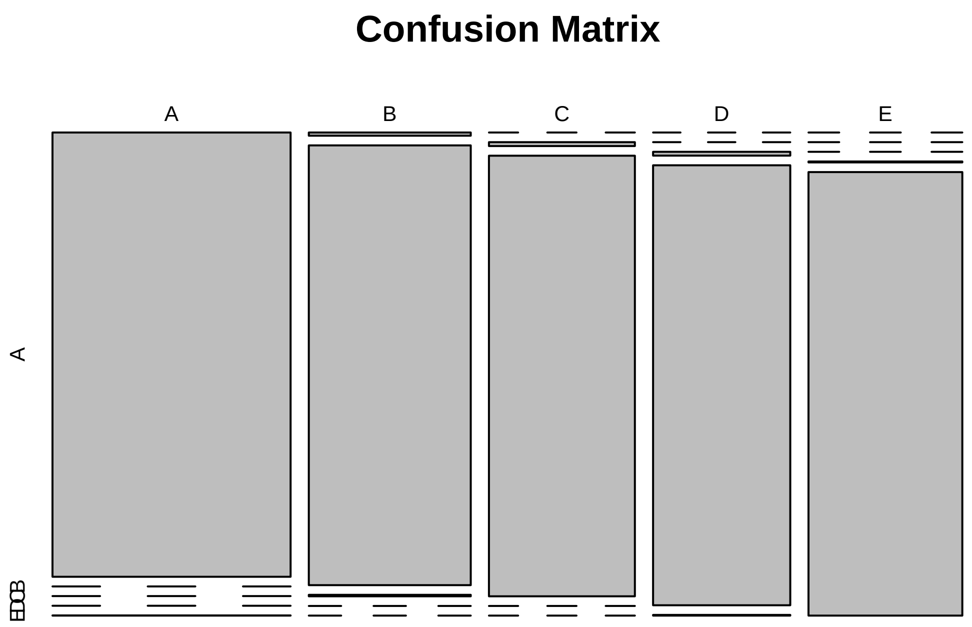
```
set.seed(78662)
modelFit <- randomForest(classe ~. , data=newtraining)
predValidation <- predict(modelFit, validation, type = "class")
qplot(predValidation,classe,data=validation,main="Predicted Vs Actual Classe")
```



```r
print(modelFit)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = newtraining)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.72%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3344    3    0    1    0 0.001194743
## B   18 2253    8    0    0 0.011408513
## C    0   17 2034    3    0 0.009737098
## D    0    0   31 1899    0 0.016062176
## E    0    0    1    3 2161 0.001847575
```

```r
CMat<-confusionMatrix(predValidation, validation$classe)
ctable <- as.table(matrix(c(CMat$table), nrow = 5, byrow = TRUE))
plot(ctable, type = "barplot", main="Confusion Matrix")
```



```r
print(CMat)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2231   11    0    0    0
##          B    0 1502   12    0    0
##          C    0    5 1356   11    0
##          D    0    0    0 1273    3
##          E    1    0    0    2 1439
##
## Overall Statistics
##
##                Accuracy : 0.9943
##                  95% CI : (0.9923, 0.9958)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9927
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9895   0.9912   0.9899   0.9979
## Specificity            0.9980   0.9981   0.9975   0.9995   0.9995
## Pos Pred Value         0.9951   0.9921   0.9883   0.9976   0.9979
## Neg Pred Value         0.9998   0.9975   0.9981   0.9980   0.9995
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1914   0.1728   0.1622   0.1834
## Detection Prevalence   0.2858   0.1930   0.1749   0.1626   0.1838
## Balanced Accuracy      0.9988   0.9938   0.9944   0.9947   0.9987
```

Result:

Based on following result from predicted model and confusion matrix data we conclude that this model is highly accurate and fit for testing against **testing** dataset.

- OOB estimate of error rate: 0.72%
- Accuracy : 0.9943
- 95% CI : (0.9923, 0.9958)
- Kappa : 0.9927

Preparing files for submitting for test:

The prediction model was tested against the testing dataset and using the new prediction model 20 new files were created which were all correct. The following chunck shows the commented out r codes to produce those files so that it does not produce those files again.

```r
set.seed(78662)
# predtest <- predict(modelFit, testing, type = "class")
# pml_write_files = function(x){
#   n = length(x)
#   for(i in 1:n){
#     filename = paste0("problem_id_",i,".txt")
#     write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
#   }
# }
#
# pml_write_files(predtest)
```