Practical Machine Learning Assignment

Introduction to the project

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, we will use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants.The goal of your project is to predict the manner in which they did the exercise.

Data Sources

The training Datset set is available at:
Training Dataset

The testing Dataset is available at:
Testing Dataset

Data Exploration

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.1
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.5.1
```

```
library(ggplot2)
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.5.1
```

```
## 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.
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.1
```

```
## randomForest 4.6-14
```

```
## 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
```

Reading the training and testing Dataset:

```
training<-read.csv("training.csv")[,c(-1)]#as they are just row numbers
testing<-read.csv("testing.csv")[,c(-1)]# as they are just row numbers
dim(training)
```

```
## [1] 19622   159
```

```
dim(testing)
```

```
## [1]  20 159
```

As it can be seen that the data is multidimensional and hence the columns with approximately zero variance should be removed.Approaches such as Principal component analysis could also be used.
Rows with NA values should also be removed.

Data preprocessing

The focus of preprocessing will be remove zero variance variables and the rows with NA values. The exact preprocessing will be done on the testing dataset as well.

```
nzv<-nearZeroVar(training)
#this variable contains the columns with almost zero variance
training<-training[,-nzv]
testing<-testing[,-nzv]
dim(training)
```

```
## [1] 19622    99
```

```
dim(testing)
```

```
## [1] 20 99
```

Now removing the variables with more than 60% NA values,as they wont be much impactful for prediction model.

```
NaValues<-sapply(training ,function(x) mean(is.na(x))>0.6)
training<-training[,NaValues==FALSE]
testing<-testing[,NaValues==FALSE]
# removing the id and time variables as they wont effect the model
training <- training[,-c(1:5)]
testing <- testing[,-c(1:5)]

dim(training)
```

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

```
dim(testing)
```

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

By removing Zero variance predictors and majorly NA values predictors,along with principal component analysis,potential predictors are reduced to 28 from 160.

Training Models

Creating validation dataset:

```
ind<-createDataPartition(training$classe,p=0.75,list=FALSE)
training<-training[ind,]
validation<-training[-ind,]
dim(training)
```

```
## [1] 14718    53
```

```
dim(validation)
```

```
## [1] 3679   53
```

As this is classification problem,we will try using decision trees,random forests,boosting an d combining models using Model Stacking.

Prediction using Decision trees

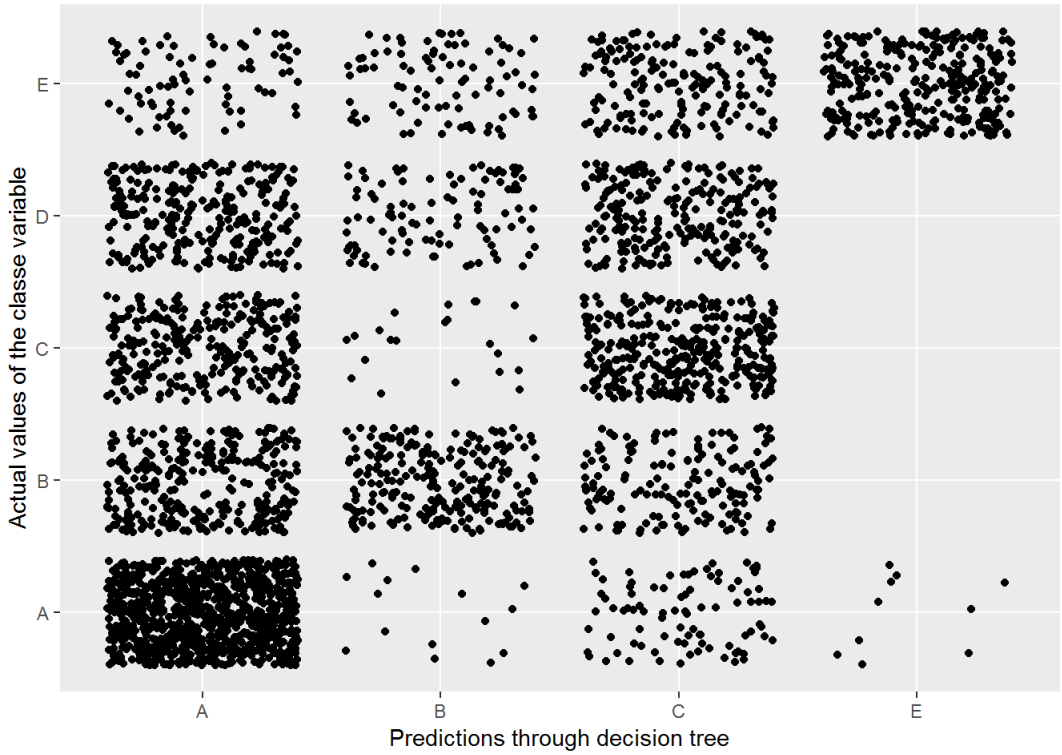
```
model1<-train(classe~.,data=training,method="rpart",trControl=trainControl(method="cv"))
pred1<-predict(model1,validation)
confusionMatrix(factor(pred1),factor(validation$classe))
```

```
## Warning in levels(reference) != levels(data): longer object length is not a
## multiple of shorter object length
```

```
## Warning in confusionMatrix.default(factor(pred1), factor(validation
## $classe)): Levels are not in the same order for reference and data.
## Refactoring data to match.
```

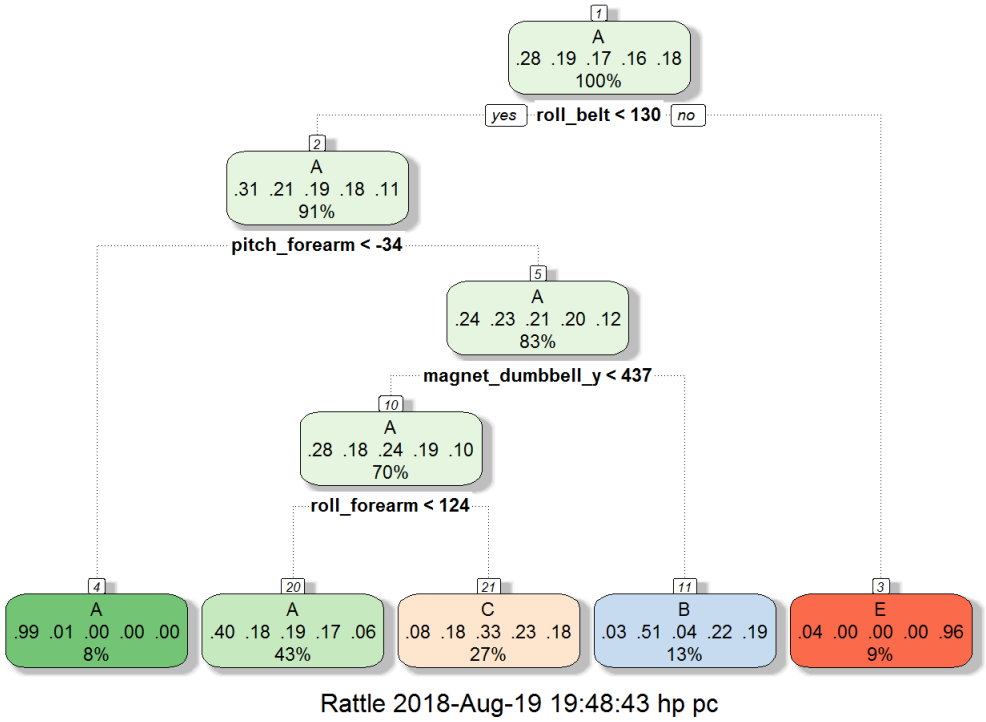
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 939 302 284 277  81
##          B  15 238  22 103  89
##          C  89 157 344 239 179
##          D   0   0   0   0   0
##          E  10   0   0   0 313
##
## Overall Statistics
##
##                Accuracy : 0.4982
##                  95% CI : (0.482, 0.5145)
##     No Information Rate : 0.2861
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3445
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8917  0.34146  0.52923   0.0000  0.47281
## Specificity            0.6408  0.92326  0.78093   1.0000  0.99669
## Pos Pred Value         0.4987  0.50964  0.34127      NaN  0.96904
## Neg Pred Value         0.9366  0.85719  0.88552   0.8318  0.89607
## Prevalence             0.2861  0.18935  0.17658   0.1682  0.17984
## Detection Rate         0.2551  0.06466  0.09345   0.0000  0.08503
## Detection Prevalence   0.5115  0.12687  0.27384   0.0000  0.08775
## Balanced Accuracy      0.7663  0.63236  0.65508   0.5000  0.73475
```

```
qplot(pred1,validation$classe,xlab="Predictions through decision tree",ylab="Actual values of the classe var
iable",geom=c("jitter"))
```




```
fancyRpartPlot(model1$finalModel)
```

```
## Warning: Bad 'data' field in model 'call'.
## To silence this warning:
##     Call prp with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```



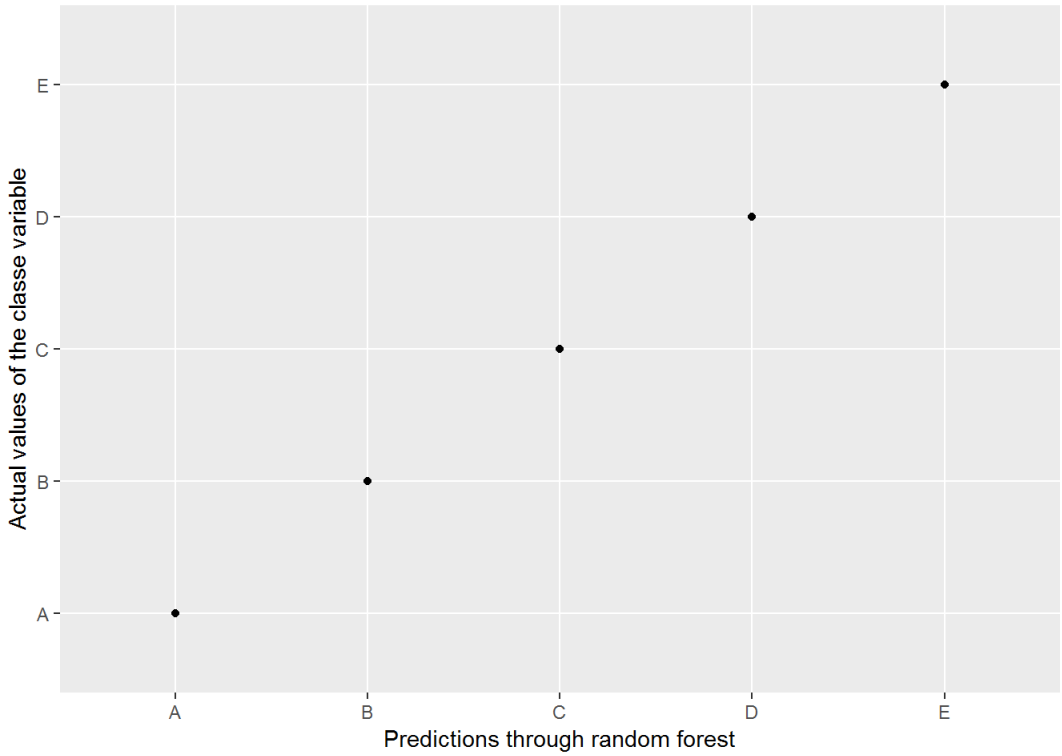
The above plot shows the decision tree used for prediction.

Prediction using random forest

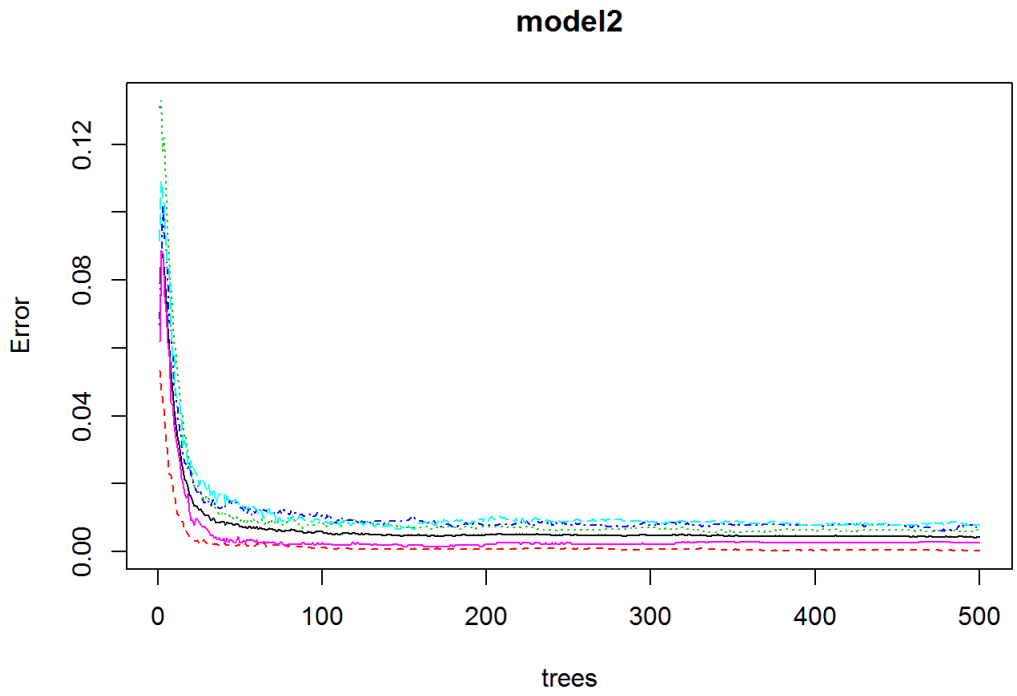
```
model2<-randomForest(classe~.,data=training)
pred2<-predict(model2,validation)
confusionMatrix(factor(pred2),factor(validation$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1048    0    0    0    0
##          B    0  712    0    0    0
##          C    0    0  642    0    0
##          D    0    0    0  598    0
##          E    0    0    0    0  679
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.999, 1)
##     No Information Rate : 0.2849
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2849   0.1935   0.1745   0.1625   0.1846
## Detection Rate         0.2849   0.1935   0.1745   0.1625   0.1846
## Detection Prevalence   0.2849   0.1935   0.1745   0.1625   0.1846
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
qplot(pred2,validation$classe,xlab="Predictions through random forest",ylab="Actual values of the classe var
iable")
```




```
plot(model2)
```



It can be seen that random forest technique has a very high accuracy.Hence it will be our final model for prediction.The error rate of the model is very low,as is apparent by the diagnostic plot

Prediction for the test set

```
final_pred<-predict(model2,testing)
final_pred
```

```
##  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
```