Assignment_ML

Downloading and Reading the Data

```
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
```

```
train = read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"))
test = read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"))
```

Data Prepocessing

```
# Selecting data that contains less than 85% NA values and 90% "" values
ind=c()
for(i in 1:dim(train)[2]){
  if(sum(is.na(train[,i]))>(0.85*dim(train)[2]) || sum(train[,i]=="")>(0.9*dim(train)[2])){
    ind=append(i,ind)
  }
}

# Removing columns containing zero variance and removing the first 7 columns as it contains irrelevant
final = train[,-ind]
final = final[,-nearZeroVar(final)]
final = final[,-(1:6)]
test=test[,-ind]
test = test[,-nearZeroVar(test)]
test = test[,-(1:6)]
```

Partitioning of Data

The data is divided into a training set and cross-validation set. The split percentage is 80% and 20% respectively.

```
set.seed(12312)
inbuild = createDataPartition(y=final$classe,p=0.8,list = FALSE)
validation=final[-inbuild,]
training = final[inbuild,]
training$classe = as.factor(training$classe)
validation$classe = as.factor(validation$classe)
```

Running a Random Forest Model

The coefficients are determined using the random forest model

```
modnew = randomForest(classe ~ ., data = training, importance = TRUE, ntrees = 8)
```

Prediction of Classes using the Cross Validation set

```
cross_par = predict(modnew,validation)
confusionMatrix(validation$classe,cross_par)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    0    0    0    0
##          B    3  755    1    0    0
##          C    0    0  684    0    0
##          D    0    0    4  639    0
##          E    0    0    0    0  721
##
## Overall Statistics
##
##                Accuracy : 0.998
##                  95% CI : (0.996, 0.9991)
##     No Information Rate : 0.2852
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9974
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9973   1.0000   0.9927   1.0000   1.0000
## Specificity            1.0000   0.9987   1.0000   0.9988   1.0000
## Pos Pred Value         1.0000   0.9947   1.0000   0.9938   1.0000
```

```
## Neg Pred Value         0.9989   1.0000   0.9985   1.0000   1.0000
## Prevalence             0.2852   0.1925   0.1756   0.1629   0.1838
## Detection Rate         0.2845   0.1925   0.1744   0.1629   0.1838
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9987   0.9994   0.9964   0.9994   1.0000
```

The accuracy of the model on the cross-validation set is 0.998 and as a result the out-of-sample error is 0.002.

Identifying the Classes of the Test set

Here the parameters obtained from the training set using the random forest model is used

```
test_par = predict(modnew,test)
test_par
```

```
##  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
```

Conclusion:

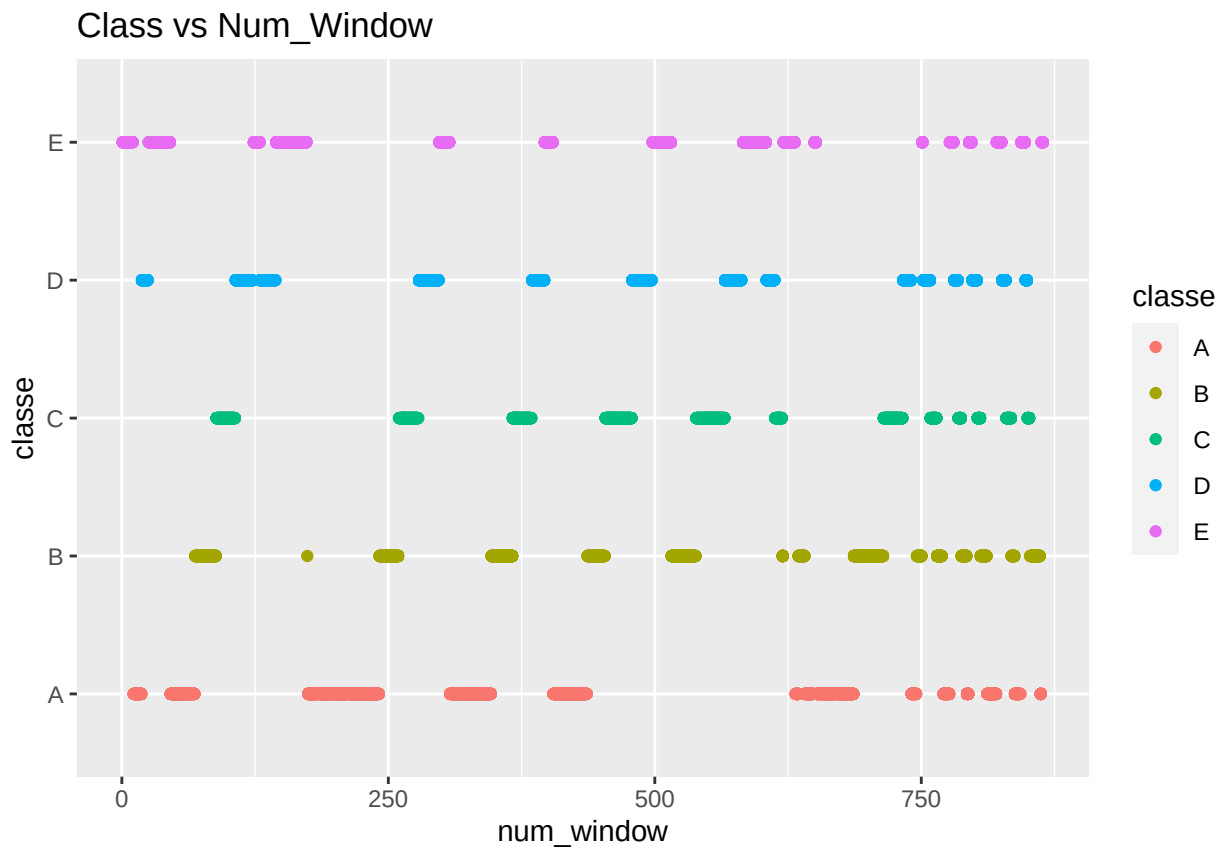
1) Here Random forest model is used because it gives a high accuracy at classification.In addition, it is less affected by the outliers present in the data.

2) The accuracy of the model on the cross-validation is 0.998 and the out-of-sample error is 0.002 which is approximately 0. This shows that the random forest model for this dataset is very efficient and accurate.

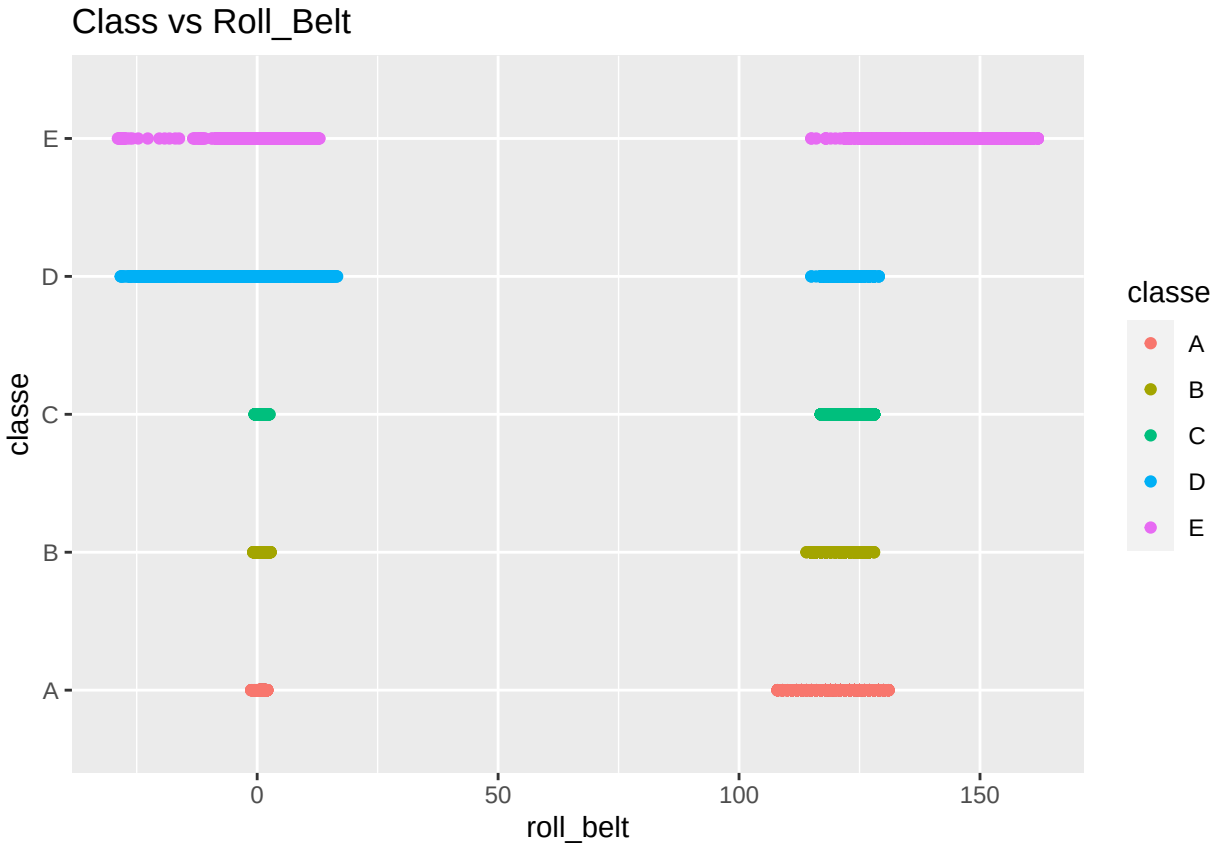
Appendix:

Visualizing the first 2 parameters - num_window and roll_belt for each class - A,B,C,D,E

```
qplot(num_window,classe,col = classe,data = train,main = "Class vs Num_Window")
```




```
qplot(roll_belt,classe,col = classe,data = train,main = "Class vs Roll_Belt")
```


Class vs Roll_Belt
classe
E
D
C
B
A
0
50
100
150
roll_belt
classe
A
B
C
D
E