Project_MachineLearning

Loading required library for this project.

```
library(caret)
```

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

```
library(e1071)
```

```
library(rpart)
library(rpart.plot)
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

I am going to choose the seed 11111 here.

```
set.seed(11111)
```

Getting the Data

I already downloaded data to my current directory. So data will be readed from the current directory.

```
trainingset <- read.table("pml-training.csv",sep=",",header= T,na.strings=c("NA","#DIV/0!",""))
testingset <- read.table("pml-testing.csv",sep=",",header= T,na.strings=c("NA","#DIV/0!",""))
```

Checking the dimension of our data set

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

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

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

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

Partioning the training data into two sets

The training data set contains 53 variables and 19622 obs. The testing data set contains 53 variables and 20 obs. In order to perform cross-validation, the training data set is partionned into 2 sets: subTraining (75%) and subTest (25%). This will be performed using random subsampling without replacement.

```
subsamples <- createDataPartition(y=trainingset$classe, p=0.75, list=FALSE)
subTraining <- trainingset[subsamples, ]
subTesting <- trainingset[-subsamples, ]
dim(subTraining)
```

[1] 14718 53

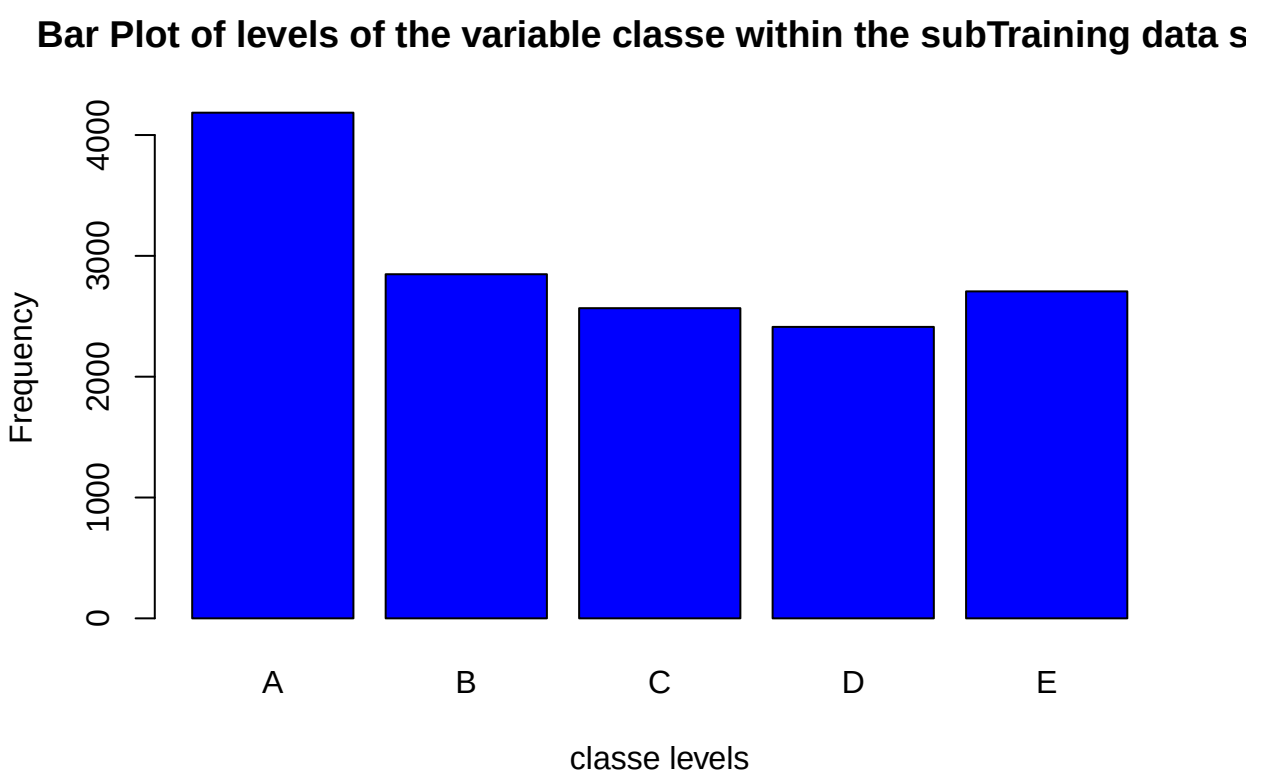
```
dim(subTesting)
```

[1] 4904 53

Exploratory data analysis

The variable "classe" contains 5 levels: A, B, C, D and E. A plot of the outcome variable will allow us to see the frequency of each levels in the subTraining data set and compare one another.

```
plot(subTraining$classe, col="blue", main="Bar Plot of levels of the variable classe within the subTrai
```



From the graph above, we can see that each level frequency is within the same order of magnitude of each other. Level A is the most frequent with more than 4000 occurrences while level D is the least frequent with about 2500 occurrences.

** prediction model:Random Forest**

```
model <- randomForest(classe ~. , data=subTraining, method="class")

# Predicting:
prediction <- predict(model, subTesting, type = "class")
# Test results on subTesting data set:
```

```
confusionMatrix(prediction, subTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    4    0    0    0
##          B    0  940    2    0    0
##          C    0    5  853    5    0
##          D    0    0    0  796    0
##          E    0    0    0    3  901
##
## Overall Statistics
##
##                Accuracy : 0.9961
##                  95% CI : (0.994, 0.9977)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9951
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9905   0.9977   0.9900   1.0000
## Specificity            0.9989   0.9995   0.9975   1.0000   0.9993
## Pos Pred Value         0.9971   0.9979   0.9884   1.0000   0.9967
## Neg Pred Value         1.0000   0.9977   0.9995   0.9981   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1917   0.1739   0.1623   0.1837
## Detection Prevalence   0.2853   0.1921   0.1760   0.1623   0.1843
## Balanced Accuracy      0.9994   0.9950   0.9976   0.9950   0.9996
```

Decision

The accuracy of the random Forest model is 0.995. The expected out-of-sample error is estimated at 0.005, or 0.5%. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set. Our Test data set comprises 20 cases. 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.

Submission

```
# predict outcome levels on the original Testing data set using Random Forest algorithm
predictfinal <- predict(model, testingset, type="class")
predictfinal
```

```
##  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
```

```r
# Write files for submission
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(predictfinal)
```