Machinelearning

```
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
library(lubridate)

##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date

library(caret)

## Loading required package: lattice

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

## The following object is masked from 'package:dplyr':
##
##     combine

library(rpart)
library(rpart.plot)
library(corrplot)

## corrplot 0.84 loaded
```

```
data.train<- read.csv("C:\\Users\\CT\\Desktop\\Machine Learning\\pml-
training.csv", na.strings = c("NA", "#DIV/0!", ""))

data.test<- read.csv("C:\\Users\\CT\\Desktop\\Machine Learning\\pml-
testing.csv", na.strings = c("NA", "#DIV/0!", ""))
```

Data Understandaing:

```
dim(data.train)

## [1] 19622   160
```

Data Transformation : Convert date and add new variable (Day)

```
data.train$cvtd_timestamp<- as.Date(data.train$cvtd_timestamp, format =
"%m/%d/%Y %H:%M")
data.train$Day<-factor(weekdays(data.train$cvtd_timestamp)) #Add day variable
```

Exploratory Data Analysis

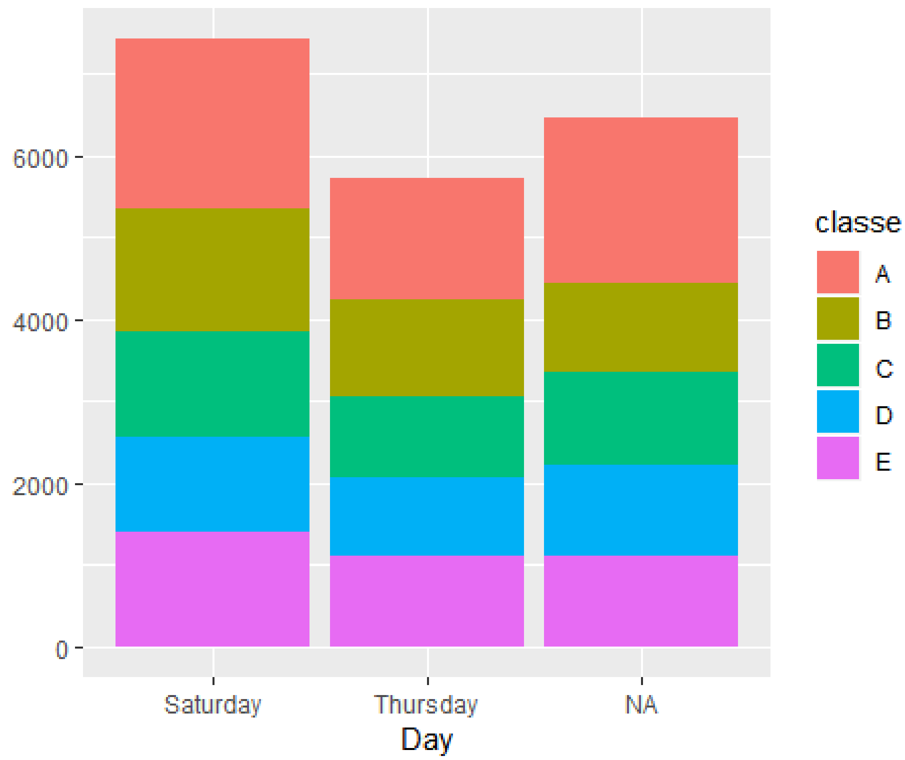
```
table(data.train$classe)

##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607

prop.table(table(data.train$classe))

##
##         A         B         C         D         E
## 0.2843747 0.1935073 0.1743961 0.1638977 0.1838243

prop.table(table(data.train$user_name))

##
##    adelmo  carlitos   charles    eurico    jeremy     pedro
## 0.1983488 0.1585975 0.1802059 0.1564570 0.1733768 0.1330140

prop.table(table(data.train$user_name,data.train$classe),1)

##
##                    A         B         C         D         E
##   adelmo   0.2993320 0.1993834 0.1927030 0.1323227 0.1762590
##   carlitos 0.2679949 0.2217224 0.1584190 0.1561697 0.1956941
##   charles  0.2542421 0.2106900 0.1524321 0.1815611 0.2010747
##   eurico   0.2817590 0.1928339 0.1592834 0.1895765 0.1765472
##   jeremy   0.3459730 0.1437390 0.1916520 0.1534392 0.1651969
##   pedro    0.2452107 0.1934866 0.1911877 0.1796935 0.1904215

prop.table(table(data.train$user_name,data.train$classe),2)

##
##                    A         B         C         D         E
```

```
##    adelmo   0.2087814 0.2043719 0.2191701 0.1601368 0.1901857
##    carlitos 0.1494624 0.1817224 0.1440678 0.1511194 0.1688384
##    charles  0.1611111 0.1962075 0.1575102 0.1996269 0.1971167
##    eurico   0.1550179 0.1559126 0.1428989 0.1809701 0.1502634
##    jeremy   0.2109319 0.1287859 0.1905319 0.1623134 0.1558082
##    pedro    0.1146953 0.1329997 0.1458212 0.1458333 0.1377876
```

```
prop.table(table(data.train$classe, data.train$Day),1)
```

```
##
##       Saturday  Thursday
##    A 0.5833804 0.4166196
##    B 0.5600147 0.4399853
##    C 0.5651030 0.4348970
##    D 0.5478220 0.4521780
##    E 0.5581302 0.4418698
```

```
qplot(x=Day, fill=classe, data = data.train)
```



Data Cleaning:

```
#### Remove columns with NA missing values
data.train <- data.train[, colSums(is.na(data.train)) == 0]
data.test <- data.test[, colSums(is.na(data.test)) == 0]

#### Remove columns that are not relevant to accelerometer measurements.
classe<- data.train$classe
trainRemove<- grepl("^X|timestamp|window", names(data.train))
```

```
data.train<- data.train[, !trainRemove]
trainCleaned<- data.train[, sapply(data.train, is.numeric)]
trainCleaned$classe<- classe
testRemove<- grepl("^X|timestamp|window", names(data.test))
data.test<- data.test[, !testRemove]
testCleaned<- data.test[, sapply(data.test, is.numeric)]
```

Now, the cleaned data contains 19622 observations and 53 variables for both train and test datasets

Create Train and Test data sets:

```
set.seed(22519)
inTrain <- createDataPartition(trainCleaned$classe, p=0.70, list=F)
trainData <- trainCleaned[inTrain, ]
testData <- trainCleaned[-inTrain, ]
```

Data Modelling:

```
controlRf <- trainControl(method="cv", 5)
rfmod<- train(classe ~., data=trainData, method="rf", trControl=controlRf,
importance=TRUE, ntree=100)
rfmod

## 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, 10991, 10988, 10989, 10991
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9902446  0.9876590
##   27    0.9911181  0.9887647
##   52    0.9852942  0.9813962
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

Accuacy of the model on Validation data set:

```
predictRfmod<- predict(rfmod, testData)
confusionMatrix(testData$classe, predictRfmod)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction    A    B    C    D    E
```

```
##          A 1673    0    0    0    1
##          B    7 1128    4    0    0
##          C    0    0 1021    5    0
##          D    0    0   13  950    1
##          E    0    0    1    7 1074
##
## Overall Statistics
##
##                Accuracy : 0.9934
##                  95% CI : (0.991, 0.9953)
##     No Information Rate : 0.2855
##     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            0.9958   1.0000   0.9827   0.9875   0.9981
## Specificity            0.9998   0.9977   0.9990   0.9972   0.9983
## Pos Pred Value         0.9994   0.9903   0.9951   0.9855   0.9926
## Neg Pred Value         0.9983   1.0000   0.9963   0.9976   0.9996
## Prevalence             0.2855   0.1917   0.1766   0.1635   0.1828
## Detection Rate         0.2843   0.1917   0.1735   0.1614   0.1825
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9978   0.9988   0.9908   0.9923   0.9982
```

```
accuracy <- postResample(predictRfmod, testData$classe)
accuracy
```

```
## Accuracy    Kappa
## 0.993373 0.991617
```

```
Error <- 1 - as.numeric(confusionMatrix(testData$classe,
predictRfmod)$overall[1])
Error
```

```
## [1] 0.006627018
```

So, the estimated accuracy of the model is 99.32% and the estimated out-of-sample error is 0.68%.

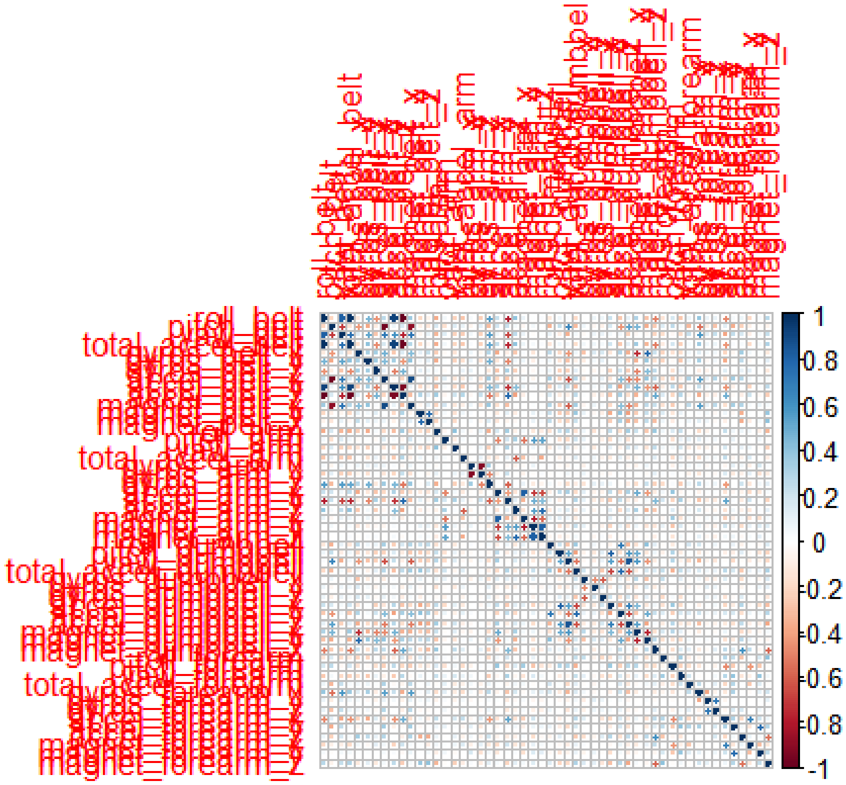
Predicting on Test Data Set

```
result <- predict(rfmod, testCleaned[, -length(names(testCleaned))])
result
```

```
##  [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
```

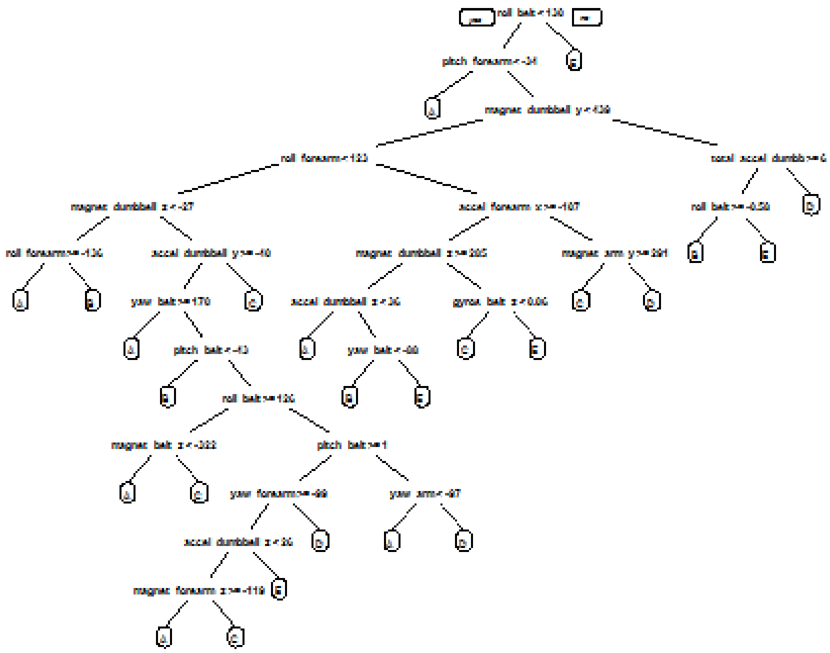
Correlation Matrix

```
corrPlot <- cor(trainData[, -length(names(trainData))])
corrplot(corrPlot, method="circle")
```



Tree Visualization

```
rtree<- rpart(classe ~ ., data=trainData, method="class")
prp(rtree)
```



roll belt < 130
pitch forearm < -34
magnet dumbbell y < 439
roll forearm < 123
total accel dumbb >= 6
magnet dumbbell z < -27
accel forearm x >= -107
roll belt >= -0.58
roll forearm >= -136
accel dumbbell y >= -40
magnet dumbbell z >= 285
magnet arm y >= 291
yaw belt >= 170
accel dumbbell z < 36
gyros belt z < 0.06
pitch belt < -43
yaw belt < -88
roll belt >= 126
magnet belt z < -322
pitch belt >= 1
yaw forearm >= -99
yaw arm < -87
accel dumbbell z < 36
magnet forearm z >= -119