Summary

The project is a classification problem and one of the best methods in machine learning is Random Forest. That is why I choose this method. Cross validation was included through the function “trainControl(method = "cv", number = 4)”. I printed out most of the information from the simulations to learn more about the processes.

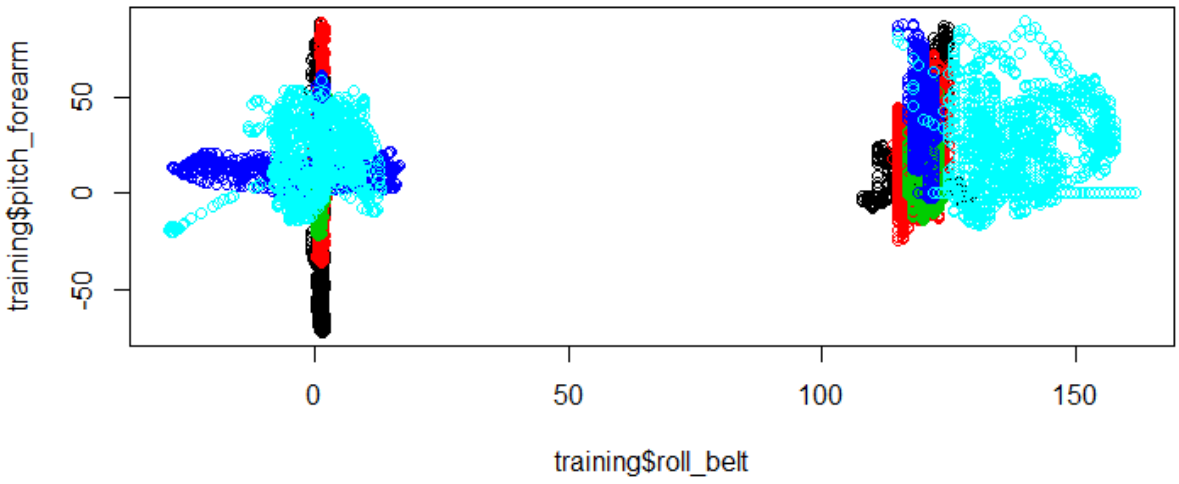
The overall logic of the program is as follows:

- read the training and testing data from the files given
- delete the columns that are empty, NA, or otherwise useless
- create new training and testing files
- check out the data by plotting pairs of variables. Two examples are shown in Fig.1 and 2. There are some clustering and dependence on the outcome but the relationships are complex
- to check expected out of sample error, divide the training data into sub-training and testing by a commonly used ratio 60% to 40%
- train the data and predict. The results reveal the needed error as error=1-accuracy=1-(sum(correctly guessed testing data)/total number of testing data)
- look at the variable importance (for general purposes)
- rerun the training on the whole training data set (the more data the better the prediction)
- make prediction on whole original test data and save the results of testing

Code and outputs

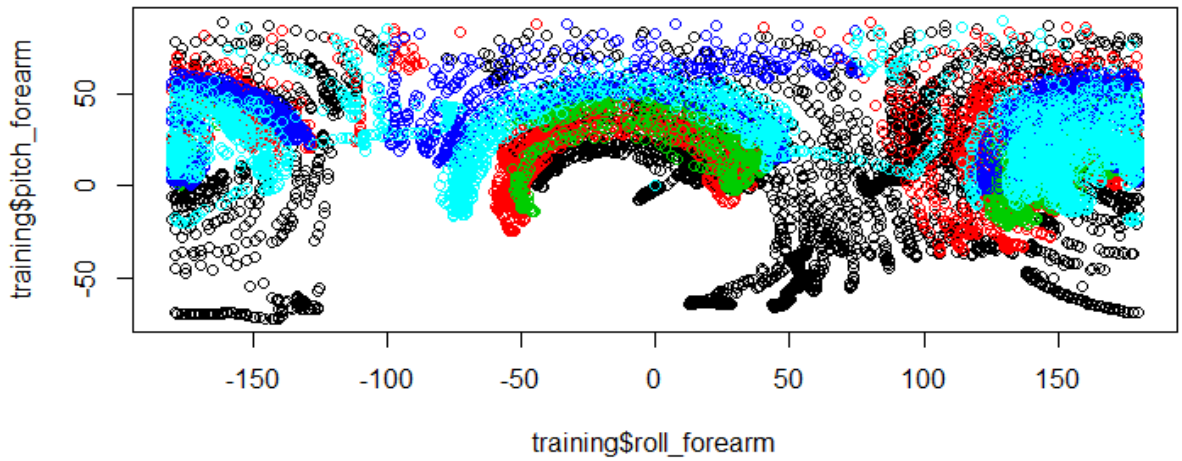
```
##Loading original training data
data_train<-read.csv("pml-training.csv")

##Loading original testing data
data_test<-read.csv("pml-testing.csv")

## deleting unnecessary columns manually
dcol<-c(1, 3,5,12:36,50:59,69:83,87:101,103:112,125:139,141:150)

training<-data_train[,-dcol]
testing<-data_test[,-dcol]

## testing by plotting
plot(training$roll_belt,training$pitch_forearm,col=training$classe)
```

Figs. 1



```
plot(training$roll_forearm,training$pitch_forearm,col=training$classe)
```

Figs. 2



There are clusters and division by “classe” as seen from Figs. 1 and 2

```
##  ----------------Out of sample error estimate----------------
datSet<-training

## -----------------Trial RF training--------------------------
library(caret)

## Breaking the original test file into training (train0) and testing (tes0) parts
inTrain0 <- createDataPartition(y=datSet$classe,p=0.6, list=FALSE)
train0 <- datSet[inTrain0,]
test0 <- datSet[-inTrain0,]

## determining number of cross validation option
trControl <- trainControl(method = "cv", number = 4) ## number of cross validations

## Training the data on the subset
modFit0 <- train(train0$classe~ .,data=train0, method="rf", trControl = trControl)
```

Displaying results of training

modFit0

```
Random Forest

11776 samples
   56 predictor
    5 classes: 'A', 'B', 'C', 'D', 'E'

No pre-processing
Resampling: Cross-Validated (4 fold)

Summary of sample sizes: 8833, 8832, 8831, 8832

Resampling results across tuning parameters:

  mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
   2    0.991     0.989  0.00198      0.00251
  31    0.996     0.994  0.00141      0.00179
  60    0.991     0.989  0.00442      0.00559

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 31.
```

summary(modFit0)

```
Length Class      Mode
call                4  -none-     call
type                1  -none-     character
predicted       11776  factor     numeric
err.rate         3000  -none-     numeric
confusion          30  -none-     numeric
votes           58880  matrix     numeric
oob.times       11776  -none-     numeric
classes             5  -none-     character
importance         60  -none-     numeric
importanceSD        0  -none-     NULL
localImportance     0  -none-     NULL
proximity           0  -none-     NULL
ntree               1  -none-     numeric
mtry                1  -none-     numeric
forest             14  -none-     list
y               11776  factor     numeric
test                0  -none-     NULL
inbag               0  -none-     NULL
xNames             60  -none-     character
problemType         1  -none-     character
tuneValue           1  data.frame list
obsLevels           5  -none-     character
```

modFit0$finalModel

```
Call:
 randomForest(x = x, y = y, mtry = param$mtry)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 31

        OOB estimate of  error rate: 0.32%
Confusion matrix:
```

```
     A    B    C    D    E  class.error
A 3346    1    0    0    1 0.0005973716
B    9 2267    2    1    0 0.0052654673
C    0    5 2046    3    0 0.0038948393
D    0    0   11 1919    0 0.0056994819
E    0    1    0    4 2160 0.0023094688
```

Random forest variable importance

varImp(modFit0)

```
rf variable importance

  only 20 most important variables shown (out of 60)

                     Overall
num_window           100.000
roll_belt             65.525
pitch_forearm         40.602
yaw_belt              31.655
magnet_dumbbell_y     30.975
magnet_dumbbell_z     29.967
pitch_belt            27.380
roll_forearm          23.292
accel_dumbbell_y      13.455
accel_forearm_x       11.696
magnet_dumbbell_x     11.511
roll_dumbbell         10.834
accel_belt_z           9.892
total_accel_dumbbell   9.046
accel_dumbbell_z       8.878
magnet_belt_y          8.063
magnet_forearm_z       7.749
magnet_belt_z          7.593
magnet_belt_x          6.457
gyros_belt_z           6.175
```

Checking predictions on test0 for out of sample error

pred0<-predict(modFit0,test0);

summary(pred0)

```
   A    B    C    D    E
2243 1506 1375 1279 1443
```

table(pred0,test0$classe)

```
pred0    A    B    C    D    E
    A 2232   11    0    0    0
    B    0 1504    2    0    0
    C    0    3 1366    6    0
    D    0    0    0 1279    0
    E    0    0    0    1 1442
```

Out of sample error estimate

```
out_error <- 1 -  sum(pred0 == test0$classe)/length(pred0)

out_error*100 ##in %
```

```
[1] 0.293143
```

Final random forest on full training data set

```
modFit<- train(training $classe~ .,data=training, method="rf", trControl = trControl)

modFit
```

```
Random Forest

19622 samples
   56 predictor
    5 classes: 'A', 'B', 'C', 'D', 'E'

No pre-processing
Resampling: Cross-Validated (4 fold)

Summary of sample sizes: 14715, 14717, 14716, 14718

Resampling results across tuning parameters:

  mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
   2    0.995     0.994  0.001304     0.001650
  31    0.998     0.998  0.000645     0.000816
  60    0.996     0.995  0.001342     0.001698

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 31.
```

```
pred <- predict(modFit, testing)

pred
```

```
[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
```