# Course Project applied machine learning

Required libraries: library(caret), library(xgboost) and register multicore support with library(doMC) and register...

First step is reading the training and testing set.

```
train = read.csv("pml-training.csv")
test = read.csv("pml-testing.csv")
```

I do not split the data into a train and cross validation set, since the xgboost algorithm, which I will use, has a built in cross validation function.

Next, Remove columns that have empty factor, NULL or NA to have a dense dataset and remove columns with little information.

```
hasNoNA = complete.cases(t(train))

trainComplete = train[,hasNoNA]
testComplete = test[,hasNoNA]

toRemove = apply(trainComplete, 2, function(col) ifelse(any(col == ''), TRUE, FALSE))

trainComplete = trainComplete[,!toRemove]
testComplete = testComplete[,!toRemove]

rm(hasNoNA)
rm(toRemove)
```

Next, I convert names into numbers. This is necessary for xgboost, since it can deal only with numeric data. I also delete all the nun-numeric columns such as timestamp, new window etc...

```
convertNamesToNumbers <- function(x){
    if (x == "carlitos"){
        return(1)
    }
    else if (x == "pedro"){
        return(2)
    }
    else if (x == "adelmo"){
        return(3)
    }
    else if (x == "charles"){
        return(4)
    }
    else if (x == "eurico"){
        return(5)
    }
    else if (x == "jeremy"){
        return(6)
    }
}

trainComplete$user_name = as.character(trainComplete$user_name)
trainComplete$user_name = sapply(trainComplete$user_name, convertNamesToNumbers)
```

```
testComplete$user_name = as.character(testComplete$user_name)
testComplete$user_name = sapply(testComplete$user_name, convertNamesToNumbers)
```

Train xgboost model, first do cross validation and then feed best parameters into model for prediction.

```
train.model <- xgb.DMatrix(data = as.matrix(train.data), label = train.label)

num_class = 5
nthread = 4
nfold = 5
nround = 20
max_depth = 5
eta = 1

params <- list(objective = "multi:softprob",
      num_class = num_class,
      max_depth = max_depth,
      eta = eta
      )

cv <- xgb.cv(train.model, params = params, nthread = nthread, nfold = nfold, nround = nround)
```

```
## [1]  train-merror:0.178078+0.009554  test-merror:0.185455+0.006959
## [2]  train-merror:0.078496+0.005568  test-merror:0.087860+0.009672
## [3]  train-merror:0.040045+0.003343  test-merror:0.049027+0.004840
## [4]  train-merror:0.022424+0.003851  test-merror:0.028540+0.004239
## [5]  train-merror:0.009760+0.001709  test-merror:0.015646+0.003464
## [6]  train-merror:0.004536+0.000577  test-merror:0.008868+0.001189
## [7]  train-merror:0.002523+0.000682  test-merror:0.006472+0.001411
## [8]  train-merror:0.001185+0.000368  test-merror:0.004740+0.001212
## [9]  train-merror:0.000522+0.000215  test-merror:0.004128+0.000931
## [10] train-merror:0.000293+0.000143  test-merror:0.002905+0.000799
## [11] train-merror:0.000166+0.000137  test-merror:0.002089+0.000691
## [12] train-merror:0.000115+0.000094  test-merror:0.001886+0.000766
## [13] train-merror:0.000038+0.000031  test-merror:0.001529+0.000702
## [14] train-merror:0.000026+0.000031  test-merror:0.001274+0.000664
## [15] train-merror:0.000013+0.000026  test-merror:0.001121+0.000572
## [16] train-merror:0.000000+0.000000  test-merror:0.000917+0.000444
## [17] train-merror:0.000000+0.000000  test-merror:0.000815+0.000374
## [18] train-merror:0.000000+0.000000  test-merror:0.000561+0.000338
## [19] train-merror:0.000000+0.000000  test-merror:0.000561+0.000338
## [20] train-merror:0.000000+0.000000  test-merror:0.000561+0.000338
```

I stop the iteration after 20 rounds, because the test error for a 5-fold cross validation does not improve any more. The parameters used for xgboost are

- num_class = 5, nthread = 4, nfold = 5, nround = 20, max_depth = 5, eta = 1
- params <- list(objective = "multi:softprob", num_class = num_class, max_depth = max_depth, eta = eta )

Establish model for prediction with these parameters

```
model <- xgboost(train.model, max.depth = max_depth, eta = eta, nthread = nthread, nround = nround, num
```

```
## [1]  train-merror:0.191061
## [2]  train-merror:0.085771
```

```
## [3]  train-merror:0.037509
## [4]  train-merror:0.021659
## [5]  train-merror:0.011874
## [6]  train-merror:0.006370
## [7]  train-merror:0.002854
## [8]  train-merror:0.001835
## [9]  train-merror:0.000815
## [10] train-merror:0.000561
## [11] train-merror:0.000306
## [12] train-merror:0.000204
## [13] train-merror:0.000051
## [14] train-merror:0.000000
## [15] train-merror:0.000000
## [16] train-merror:0.000000
## [17] train-merror:0.000000
## [18] train-merror:0.000000
## [19] train-merror:0.000000
## [20] train-merror:0.000000
```

```
# prediction on test data
predictions <- predict(model, newdata = as.matrix(test.data))
predictions
```

```
##  [1] 1 0 1 0 0 4 3 1 0 0 1 2 1 0 4 4 0 1 1 1
```

These are the predictions on the test set. The equivalences are: A=0, B=1 etc. . . .The accuracy on the test set is 100%.