Prediction Assignment Writeup

Briefly, using data gathered by tracking activity devices, such as accelerometers on the belt, forearm, arm, and dumbell of 6 participants, the goal of this project is to predict how well these devices determine what type of exercise users did.

Loading data

In this section, I downloaded the two data sets: training and testing.

```
download.file(url = "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",destfile = "
training<-read.csv("./pml-training.csv", na.strings=c("NA","#DIV/0!",""))
download.file(url = "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",destfile = "./
testing<-read.csv("./pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
```

Data Processing

The training data set has many columns containing only NAs. So, I removed this columns and, then, excluded the columns that are time-series or are non numeric. Next, I did the same for the testing data set.

```
remove.na<-sapply(training,function(x)any(is.na(x)|x == ""))
training.new<-training[,!remove.na]
training.new<-training.new[,-c(1:7)]

remove.na1<-sapply(testing,function(x)any(is.na(x)|x == ""))
testing.new<-testing[,!remove.na1]
testing.new<-testing.new[,-c(1:7)]
```

Creating cross validation data set

Here, I split the "training.new" data set into 60% training and 40% probing test.

```
library(caret)
```

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

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

```
## Warning in as.POSIXlt.POSIXct(Sys.time()): unknown timezone 'zone/tz/2017c.
## 1.0/zoneinfo/America/Los_Angeles'
```

```
set.seed(12345)
partition<-createDataPartition(training.new$classe,p=0.6,list=FALSE)
training2<-training.new[partition,]
training2$classe<-as.factor(training2$classe)
probe<-training.new[-partition,]
probe$classe<-as.factor(probe$classe)
```

Training the model

Here, I am using random forest from "caret" package. Accuracy is 98% and out of bag error is less than 1%.

```
set<-trainControl(method="cv", 5)
mod.fit<-train(classe~., data=training2,method="rf",trControl=set,
               ntree=250)
mod.fit
```

```
## Random Forest
##
## 11776 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 9421, 9420, 9420, 9421, 9422
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9893005  0.9864640
##   27    0.9900648  0.9874314
##   52    0.9865827  0.9830254
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

```
varImp(mod.fit)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 52)
##
##                      Overall
## roll_belt             100.00
## pitch_forearm          58.29
## yaw_belt               54.04
## pitch_belt             44.88
## magnet_dumbbell_z      44.19
## magnet_dumbbell_y      40.57
## roll_forearm           38.70
## accel_dumbbell_y       21.29
## magnet_dumbbell_x      19.34
## roll_dumbbell          17.79
## accel_forearm_x        17.77
## magnet_belt_z          15.00
## accel_belt_z           14.47
## accel_dumbbell_z       14.43
## magnet_forearm_z       13.19
## magnet_belt_y          11.63
## total_accel_dumbbell   11.31
## yaw_arm                10.44
## gyros_belt_z           10.11
## magnet_belt_x          10.09
```

```
mod.fit$finalModel
```

```
##
```

```
## Call:
##  randomForest(x = x, y = y, ntree = 250, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 250
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.87%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3341    5    0    0    2 0.002090800
## B   18 2246   14    0    1 0.014480035
## C    0    6 2038   10    0 0.007789679
## D    0    0   27 1900    3 0.015544041
## E    0    1    6    9 2149 0.007390300
```

Cross validation

Accuracy is 99%.

```
pred<-predict(mod.fit,probe)
confusionMatrix(pred,probe[,"classe"])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2227   11    0    0    0
##          B    5 1502    7    0    2
##          C    0    5 1357   18    3
##          D    0    0    4 1265    5
##          E    0    0    0    3 1432
##
## Overall Statistics
##
##                Accuracy : 0.992
##                  95% CI : (0.9897, 0.9938)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9898
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9978   0.9895   0.9920   0.9837   0.9931
## Specificity            0.9980   0.9978   0.9960   0.9986   0.9995
## Pos Pred Value         0.9951   0.9908   0.9812   0.9929   0.9979
## Neg Pred Value         0.9991   0.9975   0.9983   0.9968   0.9984
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2838   0.1914   0.1730   0.1612   0.1825
## Detection Prevalence   0.2852   0.1932   0.1763   0.1624   0.1829
## Balanced Accuracy      0.9979   0.9936   0.9940   0.9911   0.9963
```

Predicting with testing data set

```
pred.f<-predict(mod.fit,testing.new)
pred.f
```

```
##  [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
```