Practical Machine Learning Project : Prediction Assignment Writeup

***Varsha Shah***

***Monday, January 09, 2017***

Step 1 : Environment Preparation

Step 2 : Data Loading and Cleaning

The next step is loading the dataset from the URL provided above. The training dataset is then partinioned in 2 to create a Training set (70% of the data) for the modeling process and a Test set (with the remaining 30%) for the validations. The testing dataset is not changed and will only be used for the quiz results generation.

```
# download the datasets
training <-
read.csv("C:/Users/CICAdmin/Desktop/varsha/pml-training.csv")
testing  <-
read.csv("C:/Users/CICAdmin/Desktop/varsha/pml-testing.csv")

# create a partition with the training dataset
inTrain  <- createDataPartition(training$classe, p=0.7, list=FALSE)
TrainSet <- training[inTrain, ]
TestSet  <- training[-inTrain, ]
dim(TrainSet)
## [1] 13737   160
dim(TestSet)
## [1] 5885  160
```

Both created datasets have 160 variables. Those variables have plenty of NA, that can be removed with the cleaning procedures below. The Near Zero variance (NZV) variables are also removed and the ID variables as well.

```
NZV <- nearZeroVar(TrainSet)
TrainSet <- TrainSet[, -NZV]
TestSet  <- TestSet[, -NZV]
dim(TrainSet)
## [1] 13737   106
dim(TestSet)
## [1] 5885  106
```

Step 3 : Remove Variables that are almost NA

```
na.data<- sapply(TrainSet , function(x)  mean(is.na(x)) > 0.95 )
TrainSet<-TrainSet[,na.data==FALSE]
TestSet<-TestSet[,na.data==FALSE]
dim(TrainSet)
## [1] 13737    59
dim(TestSet)
## [1] 5885   59
```

Remove Identification VAribale 1 to 5

```
TrainSet<-TrainSet[,-c(1:5)]
TestSet<-TestSet[,-c(1:5)]
dim(TrainSet)
## [1] 13737    54
dim(TestSet)
## [1] 5885   54
```

With the cleaning process above, the number of variables for the analysis has been reduced to 54 only.

Step 4 : Correlation Analysis

A correlation among variables is analysed before proceeding to the modeling procedures.

```
corMatrix<- cor (TrainSet[,-54])
corrplot(corMatrix,order="FPC",method="color",type="lower",tl.cex=0.8
,tl.col=rgb(0,0,0))
```

accel_belt_y
total_accel_belt
roll_belt
magnet_dumbbell_x
accel_dumbbell_z
accel_dumbbell_x
yaw_dumbbell
accel_arm_x
pitch_forearm
pitch_dumbbell
yaw_belt
accel_arm_z
gyros_arm_z
pitch_belt
magnet_arm_x
gyros_belt_y
gyros_forearm_x
accel_forearm_z
magnet_forearm_z
gyros_dumbbell_x
gyros_forearm_y
gyros_arm_x
gyros_dumbbell_z
gyros_forearm_z
gyros_dumbbell_y
roll_arm
num_window
total_accel_forearm
yaw_arm
magnet_arm_z
magnet_forearm_y
pitch_arm
magnet_belt_x
gyros_arm_y
roll_forearm
accel_forearm_y
magnet_arm_y
accel_belt_x
magnet_belt_z
total_accel_arm
yaw_forearm
magnet_forearm_x
roll_dumbbell
magnet_belt_y
gyros_belt_x
magnet_dumbbell_z
accel_forearm_x
magnet_dumbbell_y
gyros_belt_z
total_accel_dumbbell
accel_dumbbell_y
accel_arm_y
accel_belt_z
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6

```
graphics.off()
```
The highly correlated variables are shown in dark colors in the graph above. To make an evem more compact analysis, a PCA (Principal Components Analysis) could be performed as pre-processing step to the datasets. Nevertheless, as the correlations are quite few, this step will not be applied for this assignment.

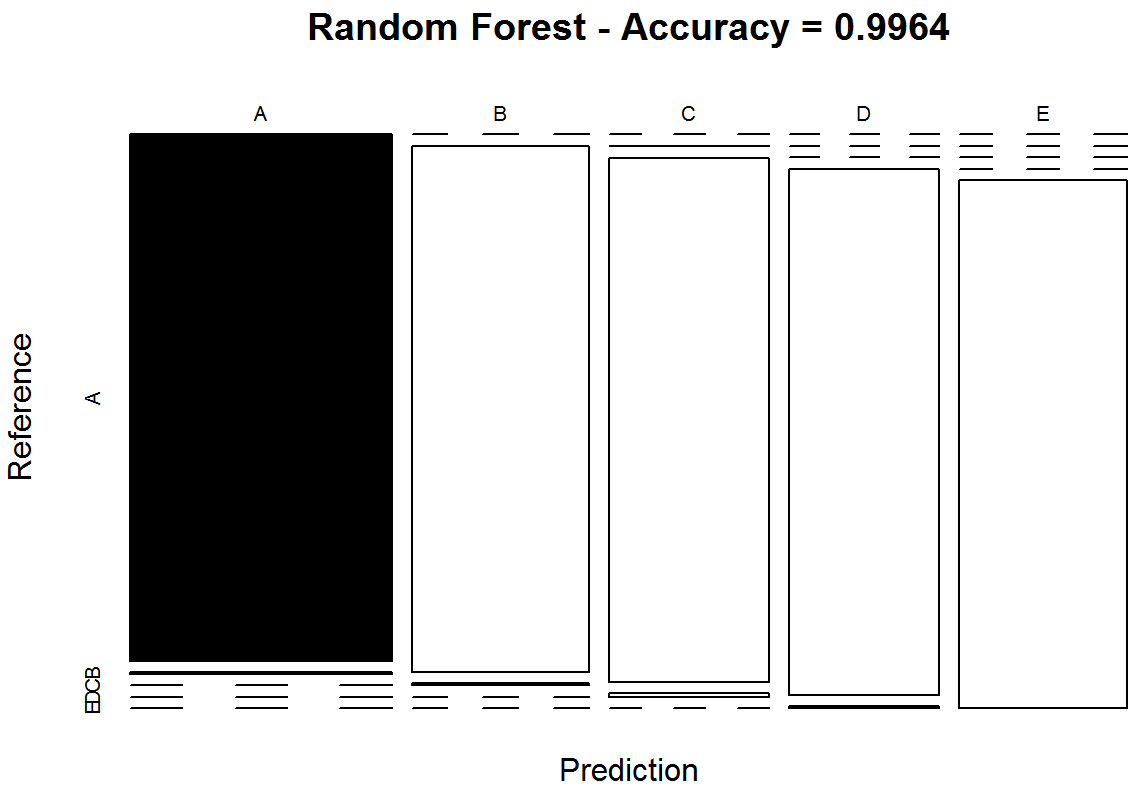
Step : 5 Prediction Model Building

Three methods will be applied to model the regressions (in the Train dataset) and the best one (with higher accuracy when applied to the Test dataset) will be used for the quiz predictions. The methods are: Random Forests, Decision Tree and Generalized Boosted Model, as described below.
A Confusion Matrix is plotted at the end of each analysis to better visualize the accuracy of the models.

Method: Random Forest

```
set.seed(12345)
controlRF <- trainControl(method="cv", number=3, verboseIter=FALSE)
modFitRandForest <- train(classe ~ ., data=TrainSet, method="rf",
                          trControl=controlRF)
modFitRandForest$finalModel
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.2%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3904    1    0    0    1    0.000512
## B    8 2647    2    1    0    0.004138
## C    0    4 2391    1    0    0.002087
## D    0    0    5 2247    0    0.002220
## E    0    0    0    4 2521    0.001584
# prediction on Test dataset
predictRandForest <- predict(modFitRandForest, newdata=TestSet)
confMatRandForest <- confusionMatrix(predictRandForest,
TestSet$classe)
confMatRandForest
## Confusion Matrix and Statistics
##
```

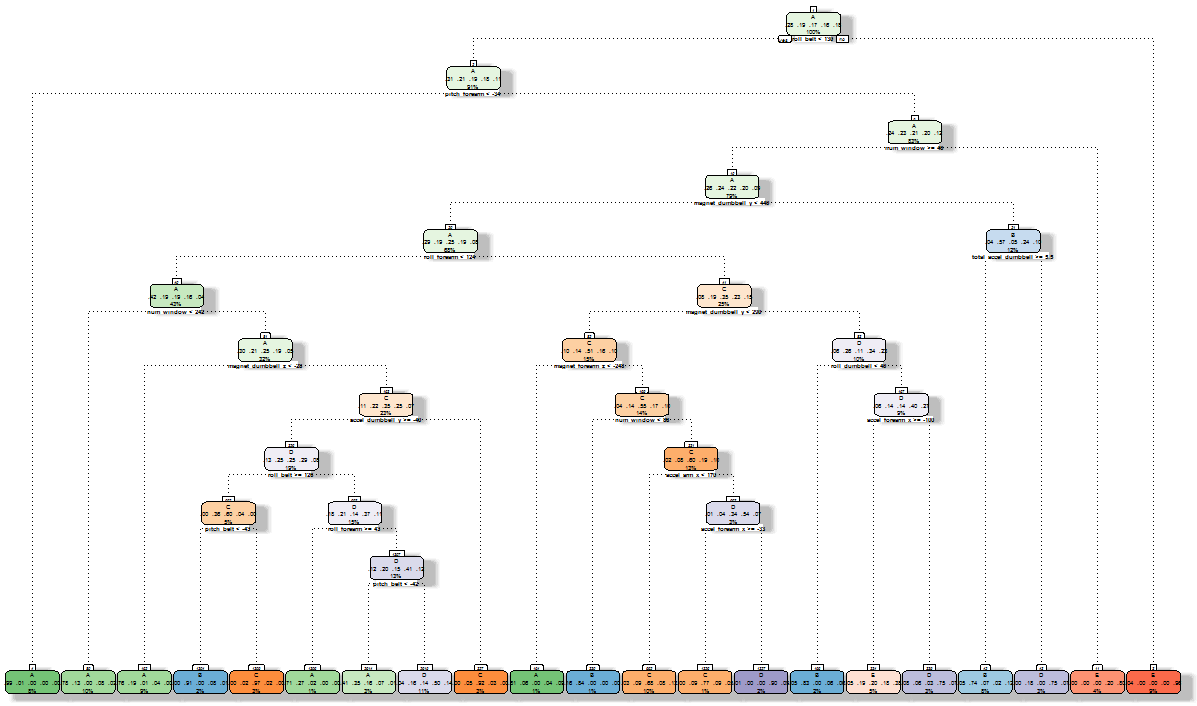
```
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    4    0    0
##          C    0    1 1022    7    0
##          D    0    0    0  957    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.996
##                  95% CI : (0.995, 0.998)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.995
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    0.995    0.996    0.993    0.996
## Specificity             0.999    0.999    0.998    0.999    1.000
## Pos Pred Value          0.997    0.996    0.992    0.996    1.000
## Neg Pred Value          1.000    0.999    0.999    0.999    0.999
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.193    0.174    0.163    0.183
## Detection Prevalence    0.285    0.193    0.175    0.163    0.183
## Balanced Accuracy       0.999    0.997    0.997    0.996    0.998
# plot matrix results
plot(confMatRandForest$table, col = confMatRandForest$byClass,
     main = paste("Random Forest - Accuracy =",
                  round(confMatRandForest$overall['Accuracy'], 4)))
```




```
graphics.off()
```

Method: Decision Trees

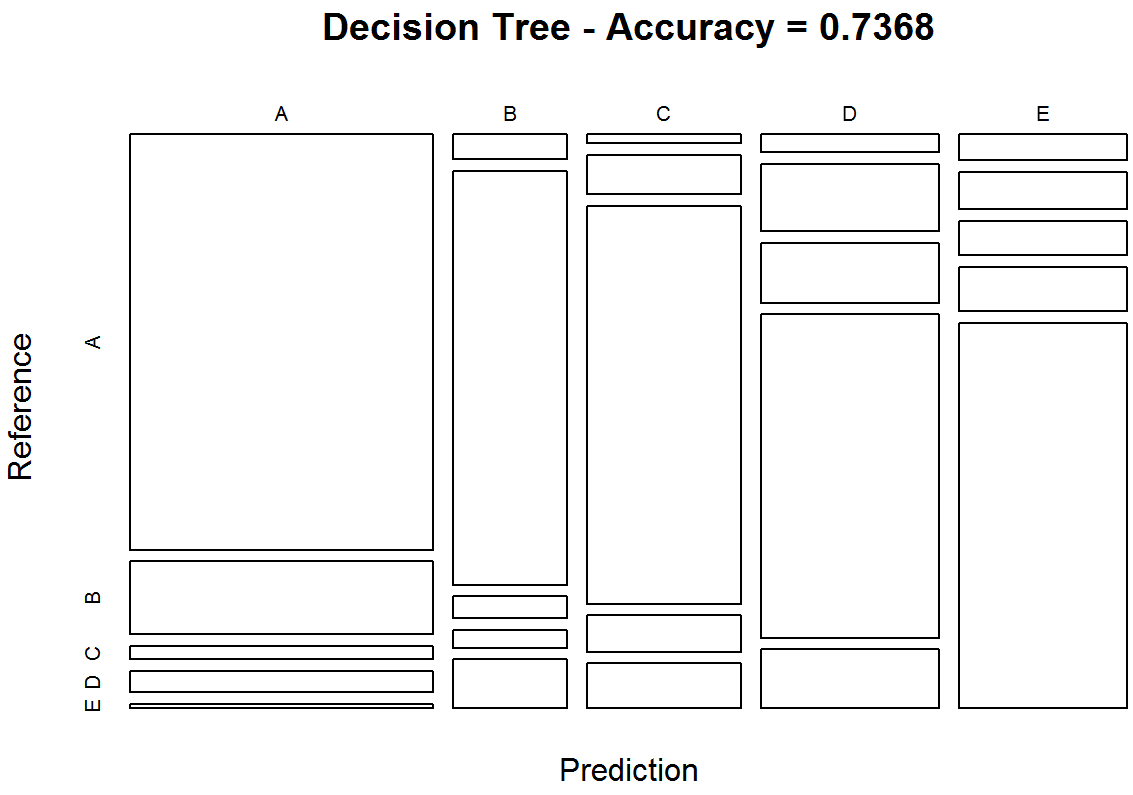
```
set.seed(12345)
modFitDecTree <- rpart(classe ~ ., data=TrainSet, method="class")
fancyRpartPlot(modFitDecTree)
```



Rattle 2017-Jan-09 14:45:54 CICAdmin

```
predictDecTree <- predict(modFitDecTree, newdata=TestSet,
type="class")
confMatDecTree <- confusionMatrix(predictDecTree, TestSet$classe)
confMatDecTree
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1530  269   51   79   16
##          B   35  575   31   25   68
##          C   17   73  743   68   84
##          D   39  146  130  702  128
##          E   53   76   71   90  786
##
## Overall Statistics
##
##                Accuracy : 0.737
##                  95% CI : (0.725, 0.748)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
```

```
##                   Kappa : 0.666
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.914   0.5048    0.724    0.728    0.726
## Specificity             0.901   0.9665    0.950    0.910    0.940
## Pos Pred Value          0.787   0.7834    0.754    0.613    0.730
## Neg Pred Value          0.963   0.8905    0.942    0.945    0.938
## Prevalence              0.284   0.1935    0.174    0.164    0.184
## Detection Rate          0.260   0.0977    0.126    0.119    0.134
## Detection Prevalence    0.331   0.1247    0.167    0.195    0.183
## Balanced Accuracy       0.908   0.7357    0.837    0.819    0.833
plot(confMatDecTree$table, col = confMatDecTree$byClass,
     main = paste("Decision Tree - Accuracy =",
                  round(confMatDecTree$overall['Accuracy'], 4)))
```

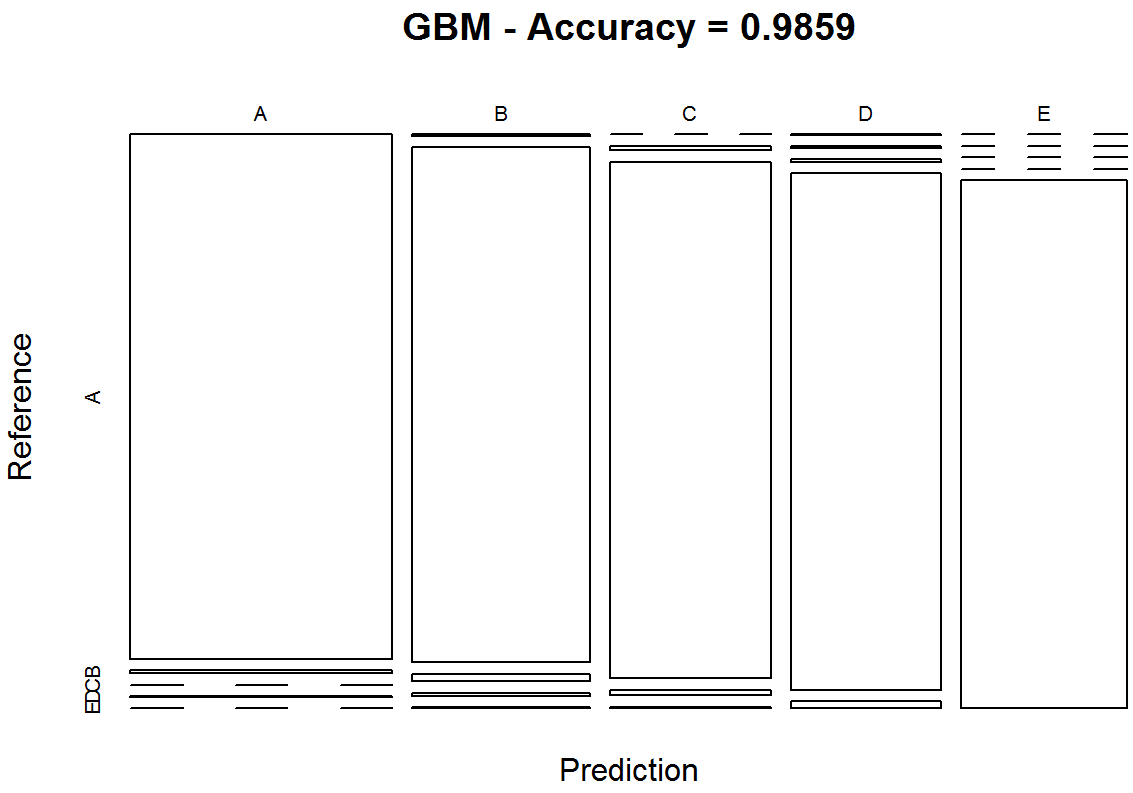



```
graphics.off()
```

Method: Generalized Boosted Model

```
set.seed(12345)
controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats
= 1)
modFitGBM  <- train(classe ~ ., data=TrainSet, method = "gbm",
                    trControl = controlGBM, verbose = FALSE)
## Loading required package: gbm
## Warning: package 'gbm' was built under R version 3.2.5
## Loading required package: survival
## Loading required package: splines
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:caret':
##
##     cluster
##
## Loading required package: parallel
## Loaded gbm 2.1.1
## Loading required package: plyr
modFitGBM$finalModel
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 42 had non-zero influence.
predictGBM <- predict(modFitGBM, newdata=TestSet)
confMatGBM <- confusionMatrix(predictGBM, TestSet$classe)
confMatGBM
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670    9    0    2    0
##          B    3 1118   16    7    2
##          C    0    9 1006   11    3
##          D    1    3    4  944   13
##          E    0    0    0    0 1064
##
## Overall Statistics
##
##                Accuracy : 0.986
##                  95% CI : (0.983, 0.989)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.982
##  Mcnemar's Test P-Value : NA
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.998    0.982    0.981    0.979    0.983
## Specificity             0.997    0.994    0.995    0.996    1.000
## Pos Pred Value          0.993    0.976    0.978    0.978    1.000
## Neg Pred Value          0.999    0.996    0.996    0.996    0.996
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.190    0.171    0.160    0.181
## Detection Prevalence    0.286    0.195    0.175    0.164    0.181
## Balanced Accuracy       0.997    0.988    0.988    0.987    0.992
plot(confMatGBM$table, col = confMatGBM$byClass,
     main = paste("GBM - Accuracy =",
round(confMatGBM$overall['Accuracy'], 4)))
```




```
graphics.off()
```

Applying the Selected Model to the Test Data

The accuracy of the 3 regression modeling methods above are:

- Random Forest : 0.9963
- Decision Tree : 0.7368
- GBM : 0.9839

In that case, the Random Forest model will be applied to predict the 20 quiz results (testing dataset) as shown below.

```
predictTEST <- predict(modFitRandForest, newdata=testing)
predictTEST
##  [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
```