Machine Learning

Vinay Sri Harsha

8/19/2019

Data Source

The training data for this project are available in the below link:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available below link:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

Data Loading and Cleaning

```
library(caret)
```

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

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

```
library(lattice)
library(ggplot2)
library(rpart)
library(rpart.plot)

training <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testing1  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

training_Data <- read.csv(url(training), strip.white = TRUE, na.strings =c("NA",""))
Test_Data <- read.csv(url(testing1),  strip.white = TRUE,na.strings=c("NA",""))

dim(training_Data)
```

```
## [1] 19622   160
```

```
dim(Test_Data)
```

```
## [1]  20 160
```

Create two partitions (75% and 25%) within the original training dataset

```
in_train <- createDataPartition(training_Data$classe,p=0.75,list = FALSE)
train_set <- training_Data[in_train,]
test_set <- training_Data[-in_train,]

dim(train_set)
```

```
## [1] 14718   160
```

```
dim(test_set)
```

```
## [1] 4904  160
```

The two datasets (train_set and test_set) have large number of NA values as well as non-zero variance variables. Both will be removed.

```
nzv_var <-nearZeroVar(train_set)
train_set <- train_set[,-nzv_var]
test_set <- test_set[,-nzv_var]

dim(train_set)
```

```
## [1] 14718   118
```

```
dim(test_set)
```

```
## [1] 4904  118
```

Remove variables that are mostly NA. A threshold of 75% selected

```
na_var <- sapply(train_set,function(x) mean(is.na(x)))>0.75
train_set <- train_set [,na_var==FALSE]
test_set <- test_set [,na_var==FALSE]

dim(train_set)
```

```
## [1] 14718    59
```

```
dim(test_set)
```

```
## [1] 4904   59
```

Since columns 1 to 5 are identification variables only, removing these as well.

```
train_set <- train_set[,-(1:5)]
test_set <- test_set [,-(1:5)]

dim(train_set)
```

```
## [1] 14718    54
```

```
dim(test_set)
```

```
## [1] 4904   54
```

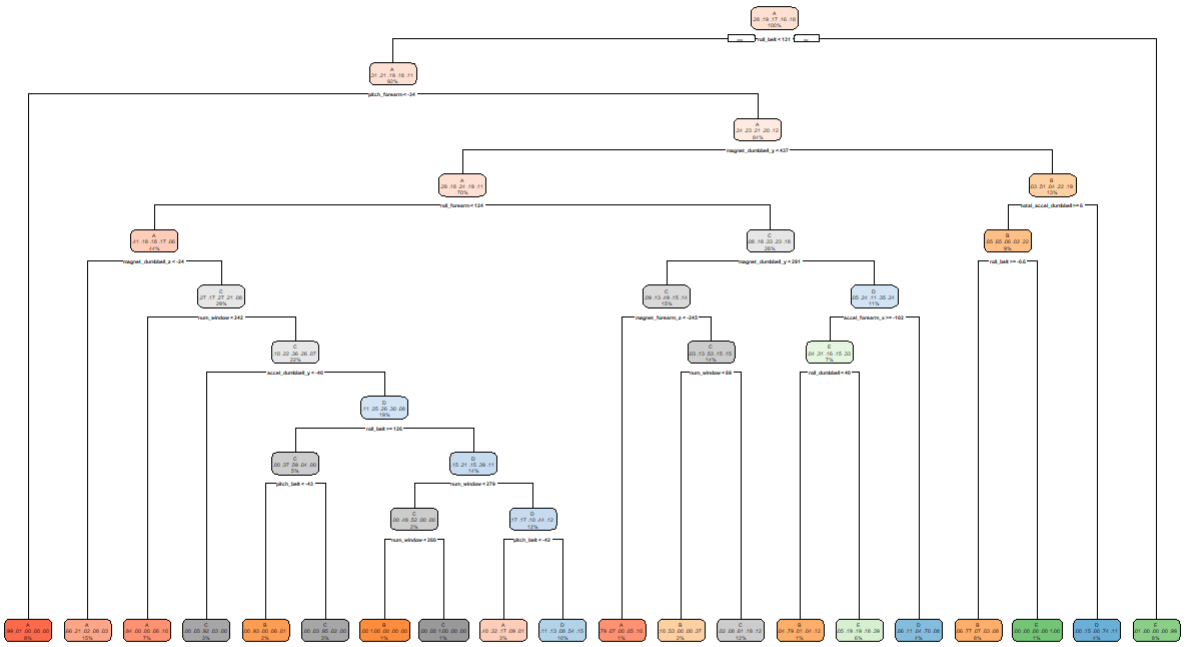
The number of variables now reduced to 54 from 160.

Prediction Models

Decision Tree Model:

```
set.seed(1813)
fit_decision_tree <-rpart(classe ~ .,data=train_set,method = "class")

rpart.plot(fit_decision_tree)
```



Predictions of decision tree model on test_set

```
predict_decision_tree <- predict(fit_decision_tree,newdata = test_set,type = "class")
conf_decision_tree <- confusionMatrix(predict_decision_tree,test_set$classe)
conf_decision_tree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1235  216   45   90   62
##          B   45  493   26   27   66
##          C   14   69  678  122   77
##          D   78  117   48  523  100
##          E   23   54   58   42  596
##
## Overall Statistics
##
##                Accuracy : 0.7188
##                  95% CI : (0.706, 0.7314)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6424
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8853   0.5195   0.7930   0.6505   0.6615
## Specificity            0.8823   0.9585   0.9304   0.9163   0.9558
## Pos Pred Value         0.7494   0.7504   0.7063   0.6039   0.7710
## Neg Pred Value         0.9509   0.8926   0.9551   0.9304   0.9262
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2518   0.1005   0.1383   0.1066   0.1215
## Detection Prevalence   0.3361   0.1340   0.1958   0.1766   0.1576
## Balanced Accuracy      0.8838   0.7390   0.8617   0.7834   0.8086
```

The predictive accuracy of decision tree model is 75%

Random Forest Model

```
ctrl_RF <-trainControl(method="repeatedcv",number=5,repeats=2)
fit_RF <- train(classe ~ .,data =train_set,method="rf",trControl=ctrl_RF,verbose=FALSE)
fit_RF$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, verbose = FALSE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.19%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4184    1    0    0    0 0.0002389486
## B    2 2843    2    1    0 0.0017556180
## C    0    6 2561    0    0 0.0023373588
## D    0    0    7 2404    1 0.0033167496
## E    0    0    0    8 2698 0.0029563932
```

Predictions of Random forest model on test_set data

```
predict_RF<-predict(fit_RF,newdata=test_set)
conf_RF <- confusionMatrix(predict_RF,test_set$classe)
conf_RF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    2    0    0    0
##          B    0  945    4    0    0
##          C    0    2  851    5    0
##          D    0    0    0  799    1
##          E    1    0    0    0  900
##
## Overall Statistics
##
##                Accuracy : 0.9969
##                  95% CI : (0.995, 0.9983)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9961
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9958   0.9953   0.9938   0.9989
## Specificity            0.9994   0.9990   0.9983   0.9998   0.9998
## Pos Pred Value         0.9986   0.9958   0.9918   0.9988   0.9989
## Neg Pred Value         0.9997   0.9990   0.9990   0.9988   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1927   0.1735   0.1629   0.1835
## Detection Prevalence   0.2847   0.1935   0.1750   0.1631   0.1837
## Balanced Accuracy      0.9994   0.9974   0.9968   0.9968   0.9993
```

The predictive accuracy of Random Forest Model is 99.8%

Random Forest Model is selected and applied to make predictions on the 20 data points on the original test data (Test_Data)

```
predict_test <- predict(fit_RF,newdata = Test_Data)
predict_test
```

```
##  [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
```