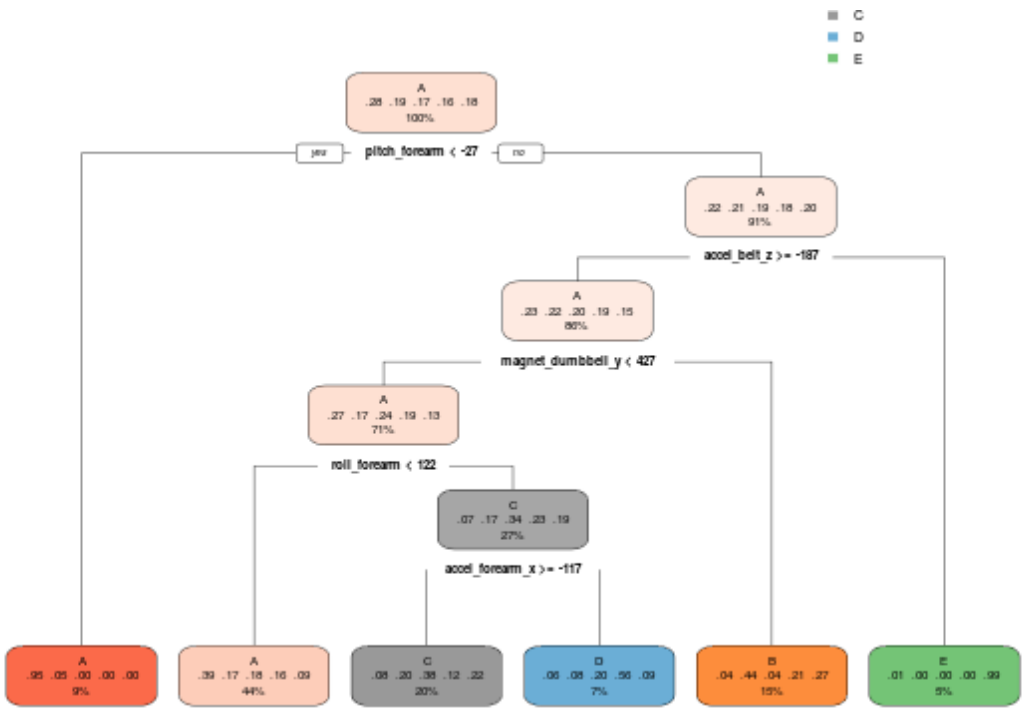
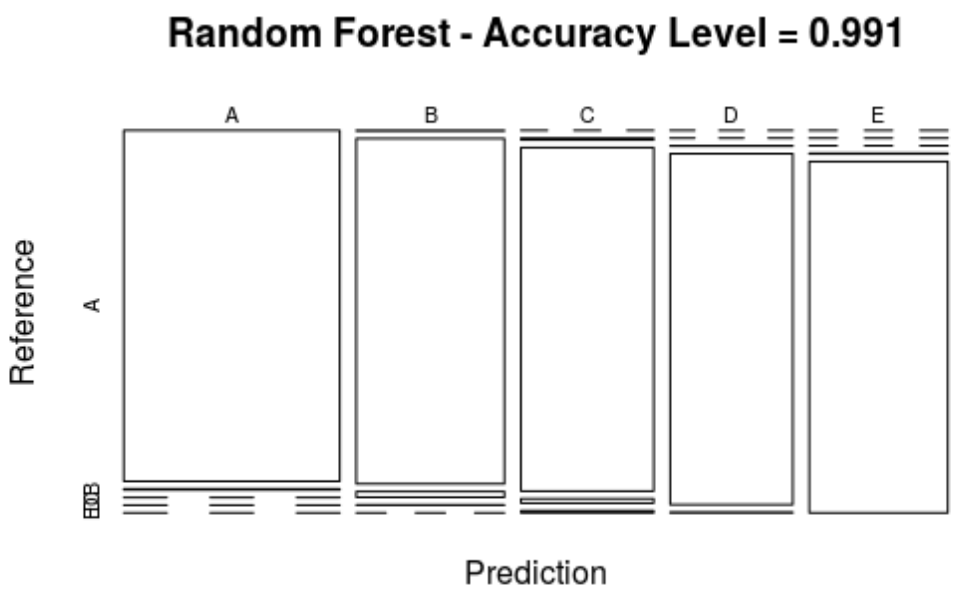
```
# Loading the data

train <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
test  <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training_dat2 <- read.csv(url(train))
testing_dat2 <- read.csv(url(test))

dim(training_dat2)
[1] 19622   160
dim(testing_dat2)
[1]  20 160


# Cleaning the data
# Removing variables that are having nearly zero variance

non_zero_variance <- nearZeroVar(training_dat2)
training_dat2 <- training_dat2[,-non_zero_variance]
testing_dat2 <- testing_dat2[,-non_zero_variance]

dim(training_dat2)
[1] 19622   100
dim(testing_dat2)
[1]  20 100

# Removing variables that are having NA values, threshold is 95%

na_val <- sapply(training_dat2, function(x) mean(is.na(x))) > 0.95
training_dat2 <- training_dat2[,na_val == FALSE]
testing_dat2 <- testing_dat2[,na_val == FALSE]

dim(training_dat2)
[1] 19622    59
dim(testing_dat2)
[1] 20 59

# Removing non-numeric variables which will not contribute into model

training_dat2 <- training_dat2[,8:59]
testing_dat2 <- testing_dat2[,8:59]

dim(training_dat2)
[1] 19622    52
dim(testing_dat2)
[1] 20 52

# Partitioning the data

partition <- createDataPartition(training_dat2$classe, p=0.6, list=FALSE)
training5 <- training_dat2[partition,]
testing5 <- training_dat2[-partition,]

dim(training5)
[1] 11776    52
dim(testing5)
[1] 7846   52

# Decision tree model

DT_model_fit <- train(classe ~ ., data = training5, method="rpart")
DT_prediction <- predict(DT_model_fit, testing5)
DT_pred_confusion_matrix <- confusionMatrix(DT_prediction, testing5$classe)
DT_pred_confusion_matrix
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 2032  633  635  557  340
         B   42  536   58  249  299
         C  126  304  532  192  342
         D   31   45  143  288   67
         E    1    0    0    0  394

Overall Statistics

               Accuracy : 0.482
                 95% CI : (0.4709, 0.4932)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16
                  Kappa : 0.3222
 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9104  0.35310  0.38889  0.22395  0.27323
Specificity            0.6144  0.89760  0.85119  0.95640  0.99984
Pos Pred Value         0.4842  0.45270  0.35561  0.50174  0.99747
Neg Pred Value         0.9452  0.85260  0.86835  0.86276  0.85935
Prevalence             0.2845  0.19347  0.17436  0.16391  0.18379
Detection Rate         0.2590  0.06832  0.06781  0.03671  0.05022
Detection Prevalence   0.5349  0.15090  0.19067  0.07316  0.05034
Balanced Accuracy      0.7624  0.62535  0.62004  0.59018  0.63654

rpart.plot(DT_model_fit$finalModel, roundint=FALSE)
```



```
# Random forest model

RF_model_fit <- train(classe ~ ., data = training5, method = "rf", ntree = 100)
RF_prediction <- predict(RF_model_fit, testing5)
RF_pred_confusion_matrix <- confusionMatrix(RF_prediction, testing5$classe)
RF_pred_confusion_matrix

Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 2230    6    0    0    0
         B    2 1505   25    1    0
         C    0    7 1342   17    8
         D    0    0    1 1266    2
         E    0    0    0    2 1432

Overall Statistics

               Accuracy : 0.991
                 95% CI : (0.9886, 0.9929)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16
                  Kappa : 0.9886
Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9991   0.9914   0.9810   0.9844   0.9931
Specificity            0.9989   0.9956   0.9951   0.9995   0.9997
Pos Pred Value         0.9973   0.9817   0.9767   0.9976   0.9986
Neg Pred Value         0.9996   0.9979   0.9960   0.9970   0.9984
Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2842   0.1918   0.1710   0.1614   0.1825
Detection Prevalence   0.2850   0.1954   0.1751   0.1617   0.1828
Balanced Accuracy      0.9990   0.9935   0.9880   0.9920   0.9964

plot(RF_pred_confusion_matrix$table, col = RF_pred_confusion_matrix$byClass,
     main = paste("Random Forest - Accuracy Level =",
                  round(RF_pred_confusion_matrix$overall['Accuracy'], 4)))
```




```
# Final prediction

Final_RF_prediction <- predict(RF_model_fit, testing_dat2)
Final_RF_prediction

[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
```