Machine learning course project

Limine.S
5/24/2020

Overview

Firstly the raw training data is splited into training and testing dataset. Then the training dataset is used to build predict models. Finally we found that random forest is better model of this case.

Clean and explore data

```
library(caret)
## Loading required package: lattice
## Loading required package: ggplot2
trainingData <- read.csv("pml-training.csv")
testingData <- read.csv("pml-testing.csv")
trainingData <- as.data.frame(trainingData)
testingData <- as.data.frame(testingData)
trainingData$classe <- factor(trainingData$classe)
NArat <- function(x){
   apply(x, 2, function(y) sum(is.na(y)))/nrow(x)
}
trainingData <- trainingData[,NArat(trainingData) < .7]
nearZero <- nearZeroVar(trainingData, saveMetrics = TRUE)
trainingData <- trainingData[, !nearZero$nzv]
trainingData <- trainingData[,-
grep("name|timestamp|window|^X",names(trainingData))]
trainingNN <- trainingData[, -length(names(trainingData))]
findCor <- findCorrelation(cor(trainingNN,use="complete.obs"), cutoff =
.7)
trainingFil <- trainingData[,-findCor]
inTrain <- createDataPartition(y = trainingFil$classe, p = 0.6,
                               list = FALSE)
training <- trainingFil[inTrain,]
testing <- trainingFil[-inTrain,]
```

rpart model

```
library(rpart)
rpartMod <- rpart(formula = classe ~ ., data = training)
rpartPred <- predict(rpartMod,newdata = testing,
                     type = "class")
confusionMatrix(rpartPred,testing$classe)
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1846  360  147  201  159
##          B   66  692   95  114  283
##          C   69  255  996  145  195
##          D  249  209  130  823  218
##          E    2    2    0    3  587
##
## Overall Statistics
##
##                Accuracy : 0.6301
##                  95% CI : (0.6193, 0.6408)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5294
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8271   0.4559   0.7281   0.6400  0.40707
```

```
## Specificity            0.8456   0.9118   0.8975   0.8771  0.99891
## Pos Pred Value         0.6804   0.5536   0.6000   0.5052  0.98822
## Neg Pred Value         0.9248   0.8748   0.9399   0.9255  0.88210
## Prevalence             0.2845   0.1935   0.1744   0.1639  0.18379
## Detection Rate         0.2353   0.0882   0.1269   0.1049  0.07482
## Detection Prevalence   0.3458   0.1593   0.2116   0.2076  0.07571
## Balanced Accuracy      0.8363   0.6838   0.8128   0.7586  0.70299
```

random forest model

```
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2228   26    3    0    0
##          B    3 1484   15    0    0
##          C    0    4 1335   33    0
##          D    0    1   15 1244    8
##          E    1    3    0    9 1434
##
## Overall Statistics
##
##                Accuracy : 0.9846
##                  95% CI : (0.9816, 0.9872)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9805
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9776   0.9759   0.9673   0.9945
## Specificity            0.9948   0.9972   0.9943   0.9963   0.9980
## Pos Pred Value         0.9872   0.9880   0.9730   0.9811   0.9910
## Neg Pred Value         0.9993   0.9946   0.9949   0.9936   0.9987
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2840   0.1891   0.1702   0.1586   0.1828
## Detection Prevalence   0.2877   0.1914   0.1749   0.1616   0.1844
## Balanced Accuracy      0.9965   0.9874   0.9851   0.9818   0.9962
```

Conclusion

Ramdon forest model is better fit model with an accuracy 0.9927, so it's preferable to choose ramdon forest to predict 20 cases.