pmlcz-1.R

changlin

Fri Sep 1 17:26:10 2017

```
# Prepare the R environment by loading required packages
library(plyr)
library(caret)
```

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

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

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(gbm)
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.3
```

```
library(survival)
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(splines)
library(parallel)
library(e1071)
# set seed for reproducibility
set.seed(425)

#options("repos")[[1]][1]
#options(repos="https://cran.cnr.berkeley.edu/")
#install.packages("gbm")

# Loading Data from url
urlTraining <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(urlTraining, destfile="pml-training.csv")
dataTraining1 <- read.csv("pml-training.csv", header=TRUE)


# use "dim" to give brief overveiw of loaded dataset structure
dim(dataTraining1)
```

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

```
# remove number column - write to "dataTraining" dataframe for further processing
dataTraining <- dataTraining1[,-1]
# Remove varables with NA greater than 90%. "dim" to review datset structure
naVar <- sapply(dataTraining, function(x) mean(is.na(x))) > 0.90
dataTraining <- dataTraining[, naVar==FALSE]
dim(dataTraining)
```

```
## [1] 19622    92
```

```
# remove variables with Near Zero Variance
nearZero <- nearZeroVar(dataTraining)
dataTraining <- dataTraining[, -nearZero]
dim(dataTraining)
```

```
## [1] 19622    58
```

```
# split dataframe training and for cross-validation.
dataTraining2 <- createDataPartition(y=dataTraining$classe,p=0.75, list=FALSE)
tidyTraining <- dataTraining[dataTraining2,]
tidyCrossVal <- dataTraining[-dataTraining2,]
# check structure of the partitioned datesets.
dim(tidyTraining)
```

```
## [1] 14718    58
```

```
dim(tidyCrossVal)
```

```
## [1] 4904   58
```

```
# set a control group - use 6 'k' fold
rfControl <- trainControl(method = "oob", number = 6)
# Random forest Model - use 300 trees
modelRandomF <- train(classe ~., data=tidyTraining, method="rf", ntree=300, metric="Kappa", trControl=rfControl)
modelRandomF$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 300, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 40
##
##         OOB estimate of  error rate: 0.05%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4185    0    0    0    0 0.0000000000
## B    1 2847    0    0    0 0.0003511236
## C    0    2 2563    2    0 0.0015582392
## D    0    0    1 2410    1 0.0008291874
## E    0    0    0    0 2706 0.0000000000
```

```
# cross validation of random forest model
predictRandomF <- predict(modelRandomF, tidyCrossVal)
confMatxRandomF <-confusionMatrix(predictRandomF, tidyCrossVal$classe)
confMatxRandomF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    1    0    0    0
##          B    0  947    1    0    0
##          C    0    1  854    0    0
##          D    0    0    0  804    1
##          E    0    0    0    0  900
##
## Overall Statistics
##
##                Accuracy : 0.9992
##                  95% CI : (0.9979, 0.9998)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.999
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9979   0.9988   1.0000   0.9989
## Specificity            0.9997   0.9997   0.9998   0.9998   1.0000
## Pos Pred Value         0.9993   0.9989   0.9988   0.9988   1.0000
## Neg Pred Value         1.0000   0.9995   0.9998   1.0000   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1931   0.1741   0.1639   0.1835
## Detection Prevalence   0.2847   0.1933   0.1743   0.1642   0.1835
## Balanced Accuracy      0.9999   0.9988   0.9993   0.9999   0.9994
```

```
# train SVMlinear
modelSVM = train(classe ~., data=tidyTraining, method="svmLinear", metric="Kappa")
modelSVM$finalModel
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 3887
##
## Objective Function Value : -1062.338 -181.6431 -4.5 -0.478 -1044.128 -50.4787 -1.1213 -608.2038 -38.0151 -814.
8714
## Training error : 0.082416
```

```
# SVM cross validation
predictSVM <- predict(modelSVM, tidyCrossVal)
confMatxSVM <- confusionMatrix(predictSVM, tidyCrossVal$classe)
confMatxSVM
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1357  115    4    0    0
##          B   30  770   58    4    1
##          C    8   64  782   61    1
##          D    0    0   11  690   41
##          E    0    0    0   49  858
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9088          
##                  95% CI : (0.9004, 0.9168)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.8845          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9728   0.8114   0.9146   0.8582   0.9523
## Specificity            0.9661   0.9765   0.9669   0.9873   0.9878
## Pos Pred Value         0.9194   0.8922   0.8537   0.9299   0.9460
## Neg Pred Value         0.9889   0.9557   0.9817   0.9726   0.9892
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2767   0.1570   0.1595   0.1407   0.1750
## Detection Prevalence   0.3010   0.1760   0.1868   0.1513   0.1850
## Balanced Accuracy      0.9694   0.8939   0.9408   0.9228   0.9700
```

```
# set control
gbmControl <- trainControl(method = "repeatedcv")
# train GBM
modelGBM <- train(classe ~., data=tidyTraining, method="gbm", metric="Kappa", trControl=gbmControl,
verbose=FALSE)
modelGBM$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 79 predictors of which 44 had non-zero influence.
```

```
# cross validation of GBM
predictGBM <- predict(modelGBM, tidyCrossVal)
confMatxGBM <-confusionMatrix(predictGBM, tidyCrossVal$classe)
confMatxGBM
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    6    0    0    0
##          B    0  942    1    0    0
##          C    0    1  847    0    0
##          D    0    0    7  804    2
##          E    0    0    0    0  899
## 
## Overall Statistics
## 
##                Accuracy : 0.9965
##                  95% CI : (0.9945, 0.998)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9956
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9926   0.9906   1.0000   0.9978
## Specificity            0.9983   0.9997   0.9998   0.9978   1.0000
## Pos Pred Value         0.9957   0.9989   0.9988   0.9889   1.0000
## Neg Pred Value         1.0000   0.9982   0.9980   1.0000   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1921   0.1727   0.1639   0.1833
## Detection Prevalence   0.2857   0.1923   0.1729   0.1658   0.1833
## Balanced Accuracy      0.9991   0.9962   0.9952   0.9989   0.9989
```

```
# Load test data from URL
urlTesting <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(urlTesting, destfile="pml-testing.csv")
testing <- read.csv("pml-testing.csv", header=TRUE)
# Check dimension of testing dataset
dim(testing)
```

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

```
# apply Random Forest Model model to testing dataset
predictTest <- predict(modelRandomF, newdata=testing)
# print out prediction
print(as.data.frame(predictTest))
```

```
##    predictTest
## 1            B
## 2            A
## 3            B
## 4            A
## 5            A
## 6            E
## 7            D
## 8            B
## 9            A
## 10           A
## 11           B
## 12           C
## 13           B
## 14           A
## 15           E
## 16           E
## 17           A
## 18           B
## 19           B
## 20           B
```