machine learning

Julian Zamudio

12/22/2020

Summary

Colecting the data

Preliminary data analysis

cleaning Data

All the incomplete data as NA values and non- defined values were excluded from the training data set. The variables to be used are the variables with values in the testing data set, therefore the columns with NA values are removed from the testing and training data set.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

```
set.seed(34567)
# removing the columns with NA's in the quiz test
testing.1 <-testing[,colSums(!is.na(testing)) > 0]
testing.1 <- testing.1[,-c(1:7)]
name.1 <- c(names(testing.1), "classe")

# cleaning the training set

training[training == ""] <- NA

# creating the training set with NA's columns

name.2 <- names(training)
training.1 <- training[,which(name.2 %in% name.1)]
name.3 <- names(training.1)
miss.1 <- name.1[-which(name.1 %in% name.3)]
testing.1 <- testing.1[,-which(name.1 %in% miss.1)]

# Creating the training and test set

inTrain  <- createDataPartition(training.1$classe, p=0.7, list=FALSE)
Train <- training.1[inTrain, ]
Test  <- training.1[-inTrain, ]
```

The data clean reduce the NA to less than 5 %. Additionally most of the data was keep it. To define with columns to keep the quiz data set was used. The Columns with NA values was removed from it. Then, the same columns were removed from the data set. Afterwards, the Train and Test set were created in from the training data set.

Defining the model

testing random model fit

```
# creating the model

modcontrol <- trainControl(method= "cv")
modrandom <- train(classe ~ ., data=Train, method="rf",
                 trControl= modcontrol, prox = TRUE)

# getting the results from the model

print(modrandom)
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12364, 12362, 12363, 12364, 12364, 12362, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9922840  0.9902388
##   27    0.9922109  0.9901468
##   52    0.9850032  0.9810302
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
modrandom$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, proximity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 0.72%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    2    0    0    0 0.0005120328
## B   15 2636    7    0    0 0.0082768999
## C    0   20 2372    4    0 0.0100166945
## D    0    0   41 2208    3 0.0195381883
## E    0    0    0    7 2518 0.0027722772
```

The accuracy of the method for is above 0.99. The kappa coefficients as well. The error per classe is in general below 0.02 and the overall estimated error is 0.72%.

Testing the Random forest model

```
predrandom <- predict( modrandom,newdata = Test)
confusionMatrix(predrandom,Test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1136    7    0    0
##          C    0    3 1018   10    0
##          D    0    0    1  954    6
##          E    0    0    0    0 1076
##
## Overall Statistics
##
##                Accuracy : 0.9954
##                  95% CI : (0.9933, 0.997)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9942
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9974   0.9922   0.9896   0.9945
## Specificity            1.0000   0.9985   0.9973   0.9986   1.0000
## Pos Pred Value         1.0000   0.9939   0.9874   0.9927   1.0000
## Neg Pred Value         1.0000   0.9994   0.9984   0.9980   0.9988
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1930   0.1730   0.1621   0.1828
## Detection Prevalence   0.2845   0.1942   0.1752   0.1633   0.1828
## Balanced Accuracy      1.0000   0.9979   0.9948   0.9941   0.9972
```

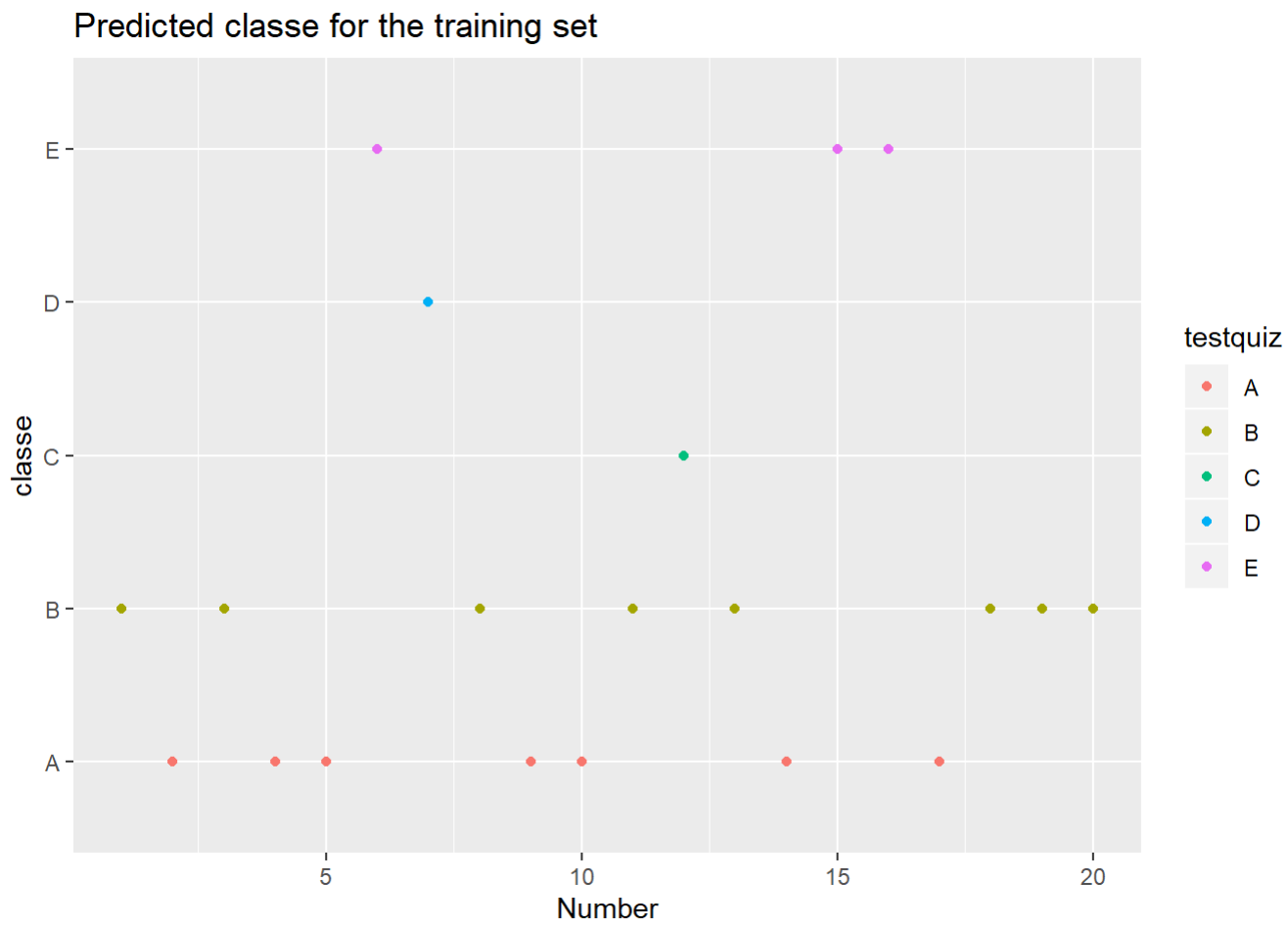
Discussion

No forecast was done in the data sets. it is defined not relevant in this study. The results are not shown, but unsupervised method and linear regression have low accuracy. Indeed, those models were discarded. Then the focus was in models to predict factor outcome. The boosting model(also results are not shown) was tested first because it was found high variance in 0.55 of the predictors in the training data base. It was expected to have compensation for the predictors with large variance using that model.The results seems to have acceptable accuracy at 150 number of trees with 0.70.

The random forest method shows high specificity and sensitivity.

Predicting the testing data set

```
testquiz <- predict(modrandom,testing.1)
qplot(x = c(1:20), y = testquiz, geom = "point", color = testquiz, main = "Predicted classe for
 the training set", xlab = "Number", ylab = "classe")
```


Predicted classe for the training set
classe
E
D
C
B
A
5
10
15
20
Number
testquiz
A
B
C
D
E