Machine Learning Lab 02

Maximilian Pfundstein (maxpf364)

2018-11-29

Contents

1 Assignment 2: Analysis of Credit Scoring

1.1 Import `creditscoring.xls`

Let's import the data and have a look at it.

Table 1: creditscoring.xls

resident	property	age	other	housing	existcr	job	depends	telephon	foreign	good_bad
4	1	67	3	2	2	3	1	2	1	good
2	1	22	3	2	1	3	1	1	1	bad
3	1	49	3	2	1	2	2	1	1	good
4	2	45	3	3	1	3	2	1	1	good
4	4	53	3	3	2	3	2	1	1	bad
4	4	35	3	3	1	2	2	2	1	good

1.2 Decision Tree Fitting

Task: Fit a decision tree to the training data by using the following measures of impurity:

a. Deviance
b. Gini index

1.2.1 Deviance

The model for the decision tree using deviance.

```
##
## Classification tree:
## tree(formula = good_bad ~ ., data = train, split = "deviance")
## Variables actually used in tree construction:
##  [1] "duration" "history"  "marital"  "existcr"  "amount"   "purpose"
##  [7] "savings"  "resident" "age"      "other"
## Number of terminal nodes:  22
## Residual mean deviance:  0.7423 = 277.6 / 374
## Misclassification error rate: 0.1869 = 74 / 396
```

The confusion matrix looks as follows:

	bad	good
bad	49	56
good	43	152

Therefore the error rate is:

```
## [1] 0.33
```

1.2.2 Gini

The model for the decision tree using gini

```
##
## Classification tree:
## tree(formula = good_bad ~ ., data = train, split = "gini")
## Variables actually used in tree construction:
##  [1] "foreign"  "coapp"    "depends"  "telephon" "existcr"  "savings"
##  [7] "history"  "property" "amount"   "marital"  "duration" "resident"
## [13] "job"      "installp" "purpose"  "employed" "housing"
## Number of terminal nodes:  53
## Residual mean deviance:  0.9468 = 324.7 / 343
## Misclassification error rate: 0.2247 = 89 / 396
```

The confusion matrix looks as follows:

	bad	good
bad	25	43
good	67	165

Therefore the error rate is:

```
## [1] 0.3666667
```

1.2.3 Conclusions

Question: Report the misclassification rates for the training and test data. Choose the measure providing the better results for the following steps.

Answer: The misqualification rate for the decision tree with deviance is `0.33` compared to the decision tree with gini as the classifier which has a misqualification rate of `0.3666667`. Therefor we will continue with using the decision tree that uses **deviance** as the classifier.

1.3 Finding the Optimal Tree

Task:

1. Use training and validation sets to choose the optimal tree depth.
2. Present the graphs of the dependence of deviances for the training and the validation data on the number of leaves.
3. Report the optimal tree, report it's depth and the variables used by the tree.
4. Interpret the information provided by the tree structure.
5. Estimate the misclassification rate for the test data.

1.3.1 Optimal Tree Depth

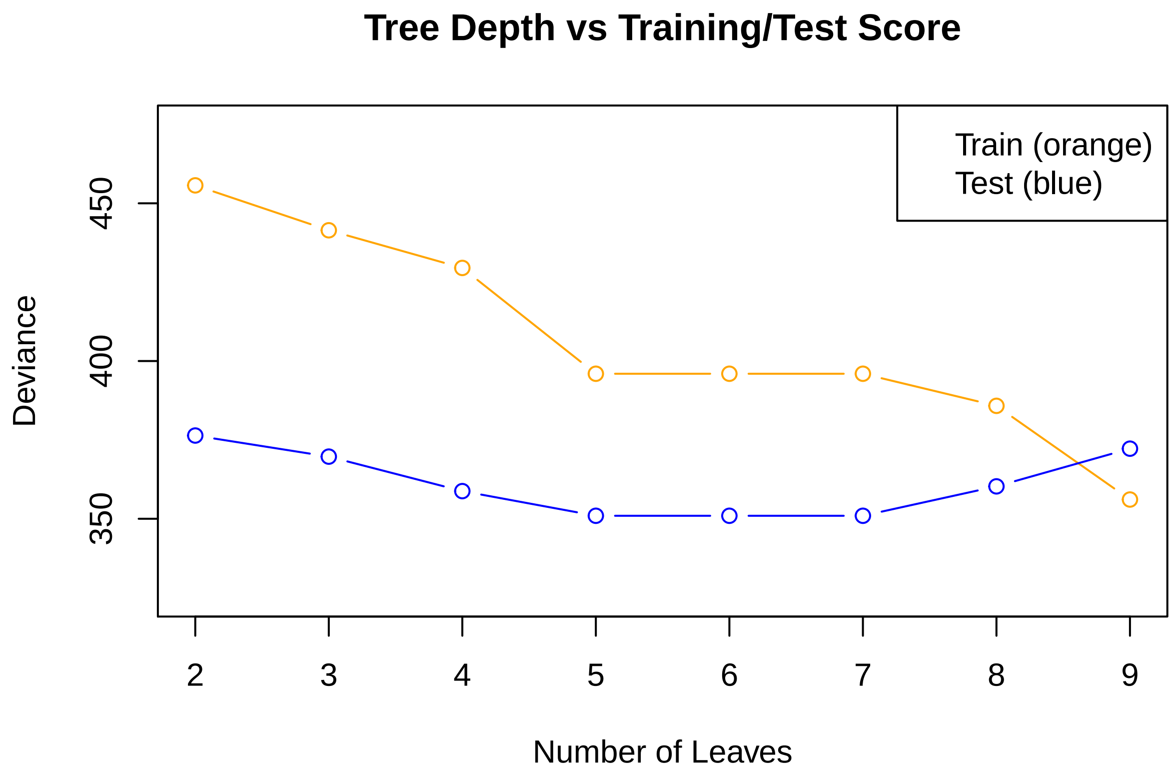
The best tree is the tree with index `5` and a test score of `350.952`.

```
## [1] 5
```

```
## [1] 350.952
```

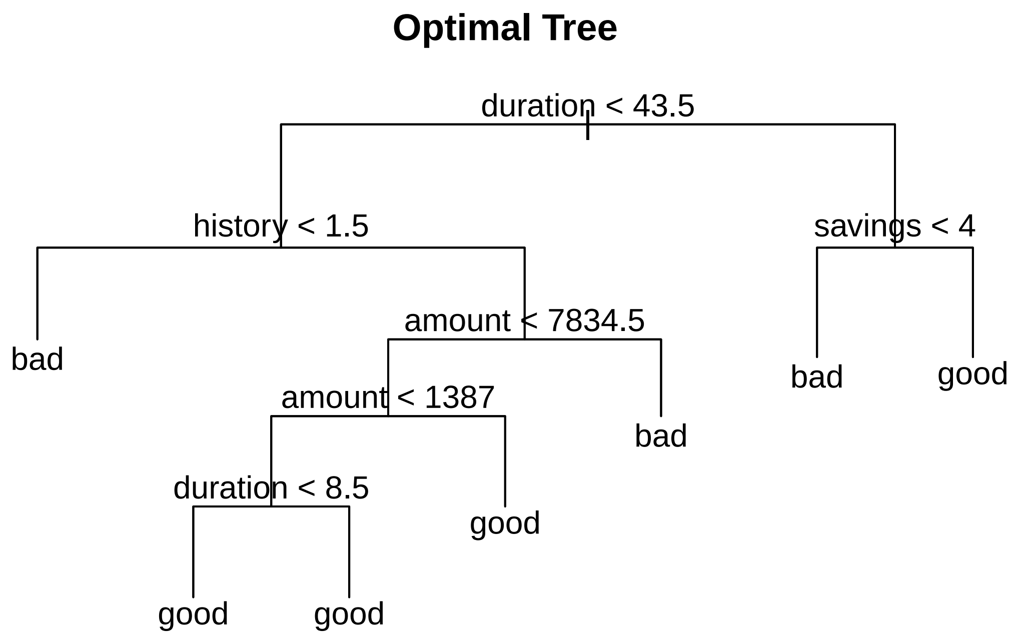
1.3.2 Dependency of Deviances

The following plots shows the Tree Depth vs the Training Score. The orange line indicates the training and the blue line the test score.



1.3.3 Optimal Tree

The following plot shows the optimal tree and it's variables. It has a depth of 4.



1.3.4 Interpretating the Tree Structure

Some blabla must be added.

1.3.5 Estimate of the Missclassification Rate

```
##
## Classification tree:
## snip.tree(tree = decisionTree_deviance, nodes = c(6L, 11L, 41L,
## 21L, 4L))
## Variables actually used in tree construction:
## [1] "duration" "history"  "amount"   "savings"
## Number of terminal nodes:  7
## Residual mean deviance:  1.018 = 396 / 389
## Misclassification error rate: 0.2323 = 92 / 396
```

	bad	good
bad	25	43
good	67	165

```
## [1] 0.2633333
```

1.4 Naïve Bayes

Task:

- Use training data to perform classification using Naïve Bayes.
- Report the confusion matrices and misclassification rates for the training and for the test data.
- Compare the results with those from step 3.

1.4.1 Classification with Naïve Bayes

Let's train the model and have a look at the summary.

```
##         Length Class  Mode
## apriori  2     table  numeric
## tables  19     -none- list
## levels   2     -none- character
## call     4     -none- call
```

1.4.2 Naïve Bayes Confusion Matrices and Misclassification Rates

Data for Naïve Bayes on train:

	bad	good
bad	62	55
good	52	231

```
## [1] 0.2675
```

Data for Naïve Bayes on test:

	bad	good
bad	50	45
good	42	163

```
## [1] 0.29
```

1.4.3 Comparison with Step 3

We can see that the misqualification rate for the optimized decision tree with `0.2633333` is better than the Naïve Bayes approach with a rate of `0.29`. We have to keep in mind that we first had to find the best tree and thus spend more time optimizing the hyper parameters.

Add more blabla.

2 Assignment 3: Uncertainty Estimation

```
set.seed(12345)
```

3 Assignemnt 4: Principal Components

```
set.seed(12345)
```

4 Appendix: Source Code

```
knitr::opts_chunk$set(echo = TRUE)
library(knitr)
library(ggplot2)
library(readxl)
library(tree)
library(e1071)

set.seed(12345)
creditscoring = read_excel("./creditscoring.xls")
creditscoring$good_bad = as.factor(creditscoring$good_bad)
kable(head(creditscoring[,(ncol(creditscoring)-10):ncol(creditscoring)]),
      caption = "creditscoring.xls")


n=dim(creditscoring)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.4))
train=creditscoring[id,]

id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.3))

valid=creditscoring[id2,]
id3=setdiff(id1,id2)
test=creditscoring[id3,]


# Create the models
decisionTree_deviance = tree(good_bad ~ ., data = train, split = "deviance")
decisionTree_gini = tree(good_bad ~ ., data = train, split = "gini")


# Prediction
prediction_deviance_train =
  predict(decisionTree_deviance, newdata = train, type = "class")
prediction_deviance_test =
  predict(decisionTree_deviance, newdata = test, type = "class")

predictiona_gini_train =
  predict(decisionTree_gini, newdata = train, type = "class")
prediction_gini_test =
  predict(decisionTree_gini, newdata = test, type = "class")
```

```r
summary(decisionTree_deviance)
#plot(decisionTree_deviance)


confusion_matrix_deviance = table(prediction_deviance_test, test$good_bad)
kable(confusion_matrix_deviance)


error_rate_deviance =
  1 - sum(diag(confusion_matrix_deviance)/sum(confusion_matrix_deviance))
print(error_rate_deviance)


summary(decisionTree_gini)
#plot(decisionTree_gini)


confusion_matrix_gini = table(prediction_gini_test, test$good_bad)
kable(confusion_matrix_gini)


error_rate_gini =
  1 - sum(diag(confusion_matrix_gini)/sum(confusion_matrix_gini))
print(error_rate_gini)


# Taken from the slides
trainScore = rep(0, 9)
testScore = rep(0, 9)

for(i in 2:9) {
  prunedTree = prune.tree(decisionTree_deviance, best = i)
  pred = predict(prunedTree, newdata = valid, type = "tree")
  trainScore[i] = deviance(prunedTree)
  testScore[i] = deviance(pred)
}


## Add one as the trim the first index
optimalTreeIdx = which.min(testScore[-1]) + 1
optimalTreeScore = min(testScore[-1])

print(optimalTreeIdx)
print(optimalTreeScore)


plot(2:9, trainScore[2:9], type = "b", col = "orange", ylim = c(325,475),
     main = "Tree Depth vs Training/Test Score", ylab = "Deviance",
     xlab = "Number of Leaves")
points(2:9, testScore[2:9], type = "b", col = "blue")
legend("topright", legend = c("Train (orange)", "Test (blue)"))
```

```r
optimalTree = prune.tree(decisionTree_deviance, best = optimalTreeIdx)
plot(optimalTree)
text(optimalTree, pretty = 1)
title("Optimal Tree")


prediction_optimalTree_test =
  predict(optimalTree, newdata = test, type = "class")

confusion_matrix_optimalTree = table(prediction_optimalTree_test, test$good_bad)

error_optimalTree =
  1 - sum(diag(confusion_matrix_optimalTree)/sum(confusion_matrix_optimalTree))


summary(optimalTree)
kable(confusion_matrix_gini)
print(error_optimalTree)


naiveBayesModel = naiveBayes(good_bad ~ ., data = train)
summary(naiveBayesModel)


# Prediction
prediction_bayes_train =
  predict(naiveBayesModel, newdata = train, type = "class")
prediction_bayes_test =
  predict(naiveBayesModel, newdata = test, type = "class")

confusion_matrix__bayes_train = table(prediction_bayes_train, train$good_bad)
confusion_matrix__bayes_test = table(prediction_bayes_test, test$good_bad)

error_bayes_train = 1 - sum(diag(confusion_matrix__bayes_train)/
                              sum(confusion_matrix__bayes_train))
error_bayes_test =  1 - sum(diag(confusion_matrix__bayes_test)/
                              sum(confusion_matrix__bayes_test))


kable(confusion_matrix__bayes_train)
print(error_bayes_train)


kable(confusion_matrix__bayes_test)
print(error_bayes_test)


set.seed(12345)


set.seed(12345)
```