lab2

Priya Kurian pullolickal

November 30, 2017

The data file creditscoring.xls contains data retrieved from a database in a private enterprise. Each row contains information about one customer. The variable good/bad indicates how the customers have managed their loans. The other features are potential predictors. Your task is to derive a prediction model that can be used to predict whether or not a new customer is likely to pay back the loan

Also if a loss matrix of good-bad 1 and bad-good 10 given ,Interpret how misclassification rate varies for the Naive Bayes

At first the data is loaded to R and divided into training/validation/test as 50/25/25.

```
credit_scoring <- read.csv("/home/george/Documents/732A95/lab2/creditscoring.csv")
n=dim(credit_scoring)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
dfTraining=credit_scoring[id,]
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.25))
dfValidation=credit_scoring[id2,]
id3=setdiff(id1,id2)
dfTest=credit_scoring[id3,]
```

2.2

To fit a decision tree on the training data two measures of impurity (1.Deviance 2.Gini index) has been used.The parameter that gives lower misclassification rate will be used to do the later analysis.

First tree function is used to fit decision tree with split parameter as "deviance" to make predictions on test and trainig data.

```
library(tree)
set.seed(12345)
tree1 <- tree(good_bad ~ .,data=dfTraining,split = "deviance")
tree_pred<-predict(tree1,dfTest,type="class")
table_deviance<-table(dfTest$good_bad,tree_pred)
table_deviance
```

```
##       tree_pred
##        bad good
##   bad   28   48
##   good  19  155
```

```
accuracy_deviance <- (sum(diag(table_deviance)))/sum(table_deviance)
misclassification1<-1-accuracy_deviance
cat("missclassification rate for deviance_test:", misclassification1)
```

```
## missclassification rate for deviance_test: 0.268
```

```
tree_pred_train<-predict(tree1,dfTraining,type="class")
cat("missclassification rate for train_deviance:", mean(tree_pred_train!=dfTraining$good_bad))
```

```
## missclassification rate for train_deviance: 0.212
```

Next “Gini” is used as split parameter to make predictions on test and trainig data.

```
#Gini Index
set.seed(12345)
tree1_gini<- tree(good_bad ~ .,data=dfTraining,split = "gini")
tree_pred_gini<-predict(tree1_gini,dfTest,type="class")
table_gini<-table(dfTest$good_bad,tree_pred_gini)
table_gini
```

```
##       tree_pred_gini
##        bad good
##   bad   18   58
##   good  33  141
```

```
accuracy_gini <- (sum(diag(table_gini)))/sum(table_gini)
misclassification2<-1-accuracy_gini
cat("missclassification rate for Gini_Test:", misclassification2)
```

```
## missclassification rate for Gini_Test: 0.364
```

```
#Train

tree_pred_train<-predict(tree1_gini,dfTraining,type="class")
cat("missclassification rate for train_gini:", mean(tree_pred_train!=dfTraining$good_bad))
```

```
## missclassification rate for train_gini: 0.242
```

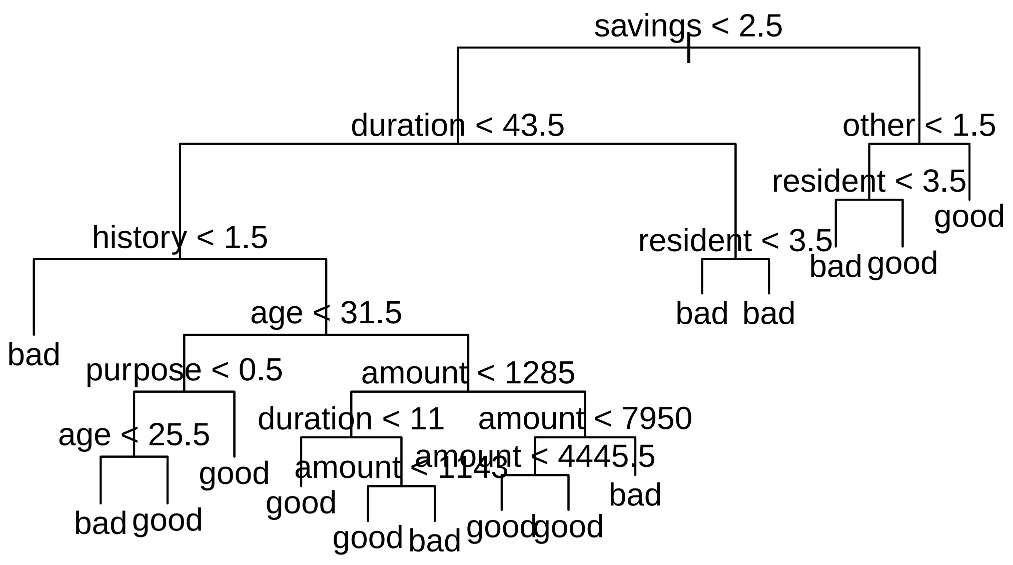
Depending on the misclassification rate , deviance measure provides the better results compared to Gini in both training and test set.

2.3

We have to find the optimal tree depth.To do that we use the training and validation data.

The graphs of the dependence of deviances for the training and the validation data on the number of leaves is presented below:

```
set.seed(12345)
library(ggplot2)
classifier<- tree(good_bad ~ .,data = dfTraining, split = "deviance" )
y_pred<- predict(classifier , newdata = dfTest,type = "class")


plot(classifier)
text(classifier)
```

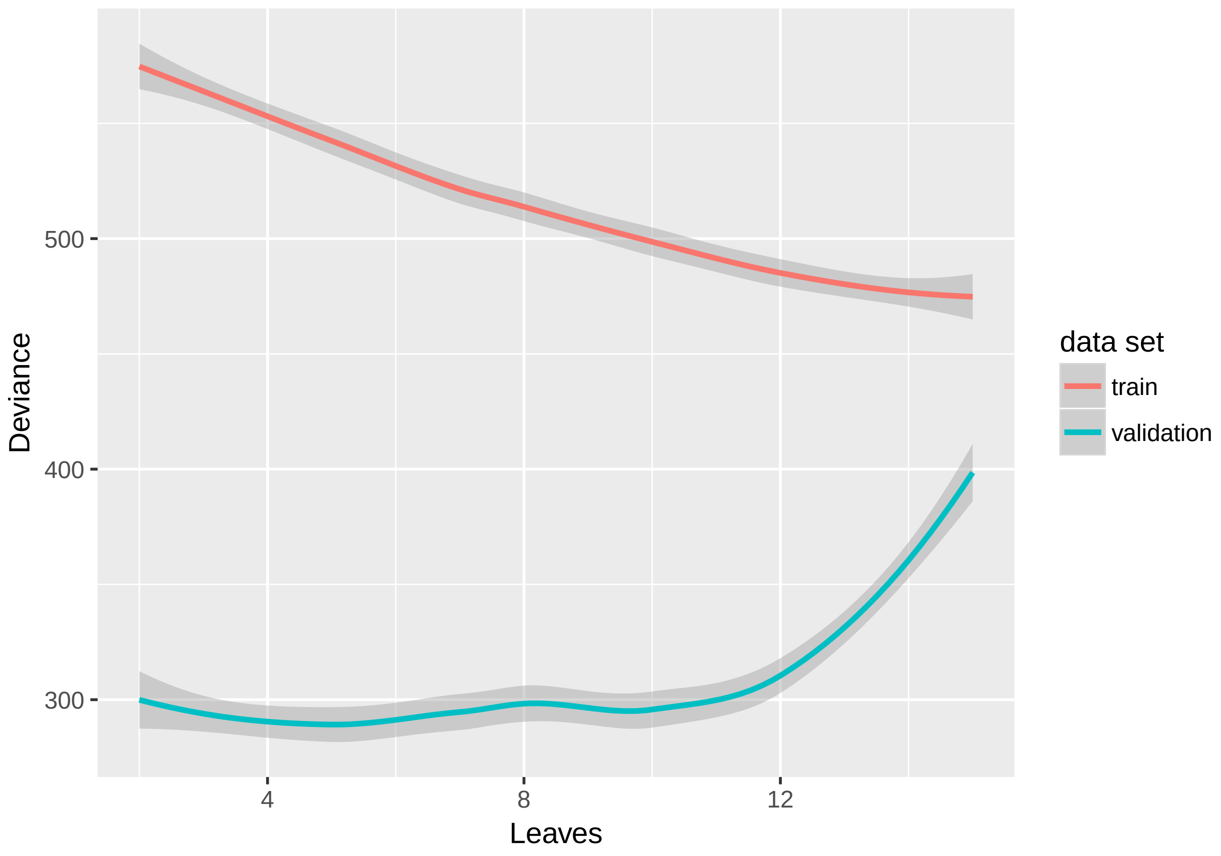


Now we use the training and validation dataset to find the optimal tree depth

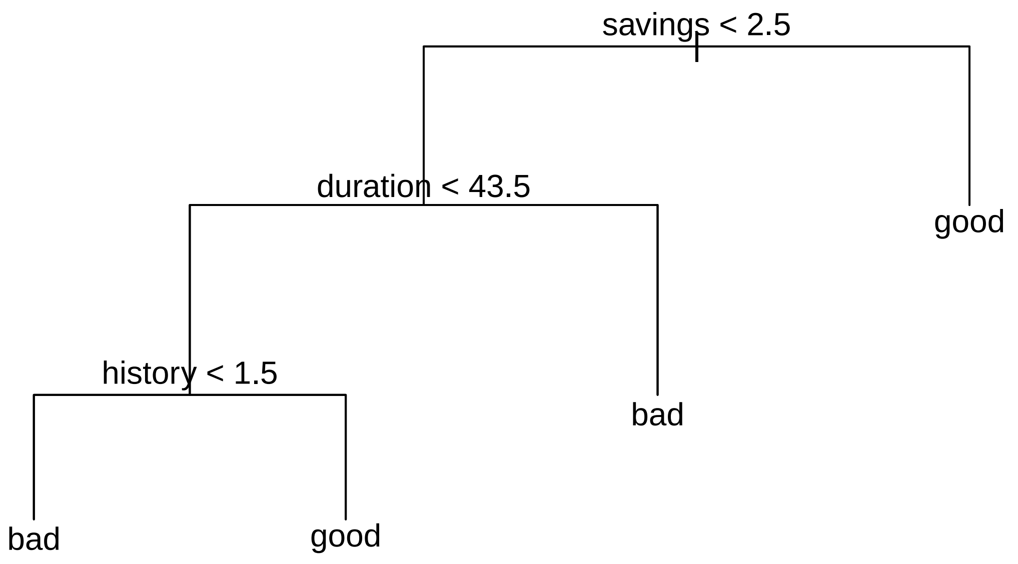
```r
n <-summary(classifier)[4]$size
possible_leaves <- seq(2,n,1)
train_deviance <- c()
validation_deviance <- c()
for(i in 2:n) {
  prunedTree <- prune.tree(classifier, best=i)
  pred1 <- predict(prunedTree, newdata=dfValidation, type="tree",na.rm=TRUE)
  pred2 <- predict(prunedTree, newdata=dfTraining, type="tree",na.rm=TRUE)
  train_deviance[i] <- deviance(pred2,na.rm=TRUE)
  validation_deviance[i] <- deviance(pred1,na.rm=TRUE)
}
data <- data.frame(Leaves=vector("numeric"),
                   deviance=vector("numeric"))
for(i in 2:n)
{
  data[i,"Leaves"] <- i
  data[i,"deviance"] <- train_deviance[i]
  data[i,"set"] <- "train"
}

for(i in (n+1):(2*n))
{
  data[i,"Leaves"] <- i-n
  data[i,"deviance"] <- validation_deviance[i-n]
  data[i,"set"] <- "validation"

}
set.seed(12345)
ggplot(data = data ,aes(x = Leaves , y = deviance , col = set)) +
  geom_smooth() + labs(x="Leaves", y="Deviance", color="data set")
```

```
## `geom_smooth()` using method = 'loess'
```

```
## Warning: Removed 2 rows containing non-finite values (stat_smooth).
```



```r
optimal_tree <- prune.tree(classifier, best=4)
plot(optimal_tree)
text(optimal_tree, pretty=0)
```



It is obsesrved that with the 4 leaves the best model can be acquired which has less deviance(pruning). The depth of the tree is 3.

Tree Interpretation

The decision making factors are savings,duration and history.If a person's saving is greater than 2.5 ,then he has higher chance of getting loan.same way if the loan duration is less than 43.5 or the history value is greater than 1.5,he has higher chance of getting loan.Misclassification rate of test data is 0.268.

misclassification rate of test data

```
pr<-predict(classifier,newdata=dfTest,type="class")
conf_mat_test <- table(dfTest$good_bad,pr )

cat("missclassification rate for test data:", mean(pr!=dfTest$good_bad))
```

```
## missclassification rate for test data: 0.268
```

2.4

We then use a Naivebayes classifier to perform classification using trainig data. The confusion matrices and misclassification rates for the training and test data are reported below.

```
library(e1071)
model <- naiveBayes(good_bad ~ ., data = dfTraining)
f1<-predict(model,newdata = dfTraining,type="class")
table(dfTraining$good_bad,f1)
```

```
##       f1
##        bad good
##   bad   95   52
##   good  98  255
```

```
cat("missclassification rate for navie train data:", mean(f1!=dfTraining$good_bad))
```

```
## missclassification rate for navie train data: 0.3
```

```
f2<-predict(model,newdata=dfTest,type="class")
table(dfTest$good_bad,f2)
```

```
##       f2
##        bad good
##   bad   46   30
##   good  49  125
```

```
cat("missclassification rate for navie test data:", mean(f2!=dfTest$good_bad))
```

```
## missclassification rate for navie test data: 0.316
```

Misclassification rate is higher for Naive Bayes when compared to the Deviance.Hence decision tree with deviance performs better than naivebayes.

2.5 Naive Bayes classification using the loss matrix.

The loss matrix is used to weight misclassification differently.This refers to the false positives (or type I errors) and false negatives (type II errors), one type of error is more of a loss to us than another type of error.In credit fraud detection, a model that identifies too many false positives is probably a better model than that identifies too many false negatives. So that, we may not miss too many real frauds. Hence False Posiive(FP) is given higher value as 10.

```
#test data
# loss_matrix<-matrix(c(0,10,1,0),ncol=2)
set.seed(12345)
model2 <- naiveBayes(good_bad~.,data=dfTraining)
f1_1<-predict(model2,newdata=dfTest,type="raw")
a<-ifelse(f1_1[,2]/f1_1[,1] >10, 1, 0)
```

```r
tab_test<-table(dfTest$good_bad,a)
accuracy_loss <- (sum(diag(tab_test)))/sum(tab_test)
mis_loss<-1-accuracy_loss
cat("missclassification rate for deviance_test:", mis_loss)
```

```
## missclassification rate for deviance_test: 0.508
```

```r
#training data
f1_2<-predict(model2,newdata=dfTraining,type="raw")
b<-ifelse(f1_2[,2]/f1_2[,1] >10, 1, 0)
tab_train<-table(dfTraining$good_bad,b)
accuracy_loss_train <- (sum(diag(tab_train)))/sum(tab_train)
mis_loss_train<-1-accuracy_loss_train
cat("missclassification rate for deviance_test:", mis_loss_train)
```

```
## missclassification rate for deviance_test: 0.546
```

When we are prediciting the model with type="raw" we get the probability of two class, then applying conditional probability if it is greater than 10 classified as 1 or less 0. Compared to step 4 the mis-classification rates have increased for the naivebayes classifier with the loss matrix.

APPENDIX

```r
#Assignment 2
credit_scoring<-read.csv("D:/MastersLiu/Statistics & Data Mining/732A95Introduction to Machine Learning
#Question1
n=dim(credit_scoring)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
dfTraining=credit_scoring[id,]
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.25))
dfValidation=credit_scoring[id2,]
id3=setdiff(id1,id2)
dfTest=credit_scoring[id3,]


# question 2 ,Deviance
library(tree)
set.seed(12345)
tree1 <- tree(good_bad ~ .,data=dfTraining,split = "deviance")
tree_pred<-predict(tree1,dfTest,type="class")
table_deviance<-table(dfTest$good_bad,tree_pred)
table_deviance
accuracy_deviance <- (sum(diag(table_deviance)))/sum(table_deviance)
misclassification1<-1-accuracy_deviance
cat("missclassification rate for deviance_test:", misclassification1)

tree_pred_train<-predict(tree1,dfTraining,type="class")
cat("missclassification rate for train_deviance:", mean(tree_pred_train!=dfTraining$good_bad))

#Gini Index
set.seed(12345)
tree1_gini<- tree(good_bad ~ .,data=dfTraining,split = "gini")
```

```r
tree_pred_gini<-predict(tree1_gini,dfTest,type="class")
table_gini<-table(dfTest$good_bad,tree_pred_gini)
table_gini
accuracy_gini <- (sum(diag(table_gini)))/sum(table_gini)
misclassification2<-1-accuracy_gini
cat("missclassification rate for Gini_Test:", misclassification2)

#Train

tree_pred_train<-predict(tree1_gini,dfTraining,type="class")
cat("missclassification rate for train_gini:", mean(tree_pred_train!=dfTraining$good_bad))



#question 3:
set.seed(12345)
library(ggplot2)
classifier<- tree(good_bad ~ .,data = dfTraining, split = "deviance" )
y_pred<- predict(classifier , newdata = dfTest,type = "class")


plot(classifier)
text(classifier)
n <-summary(classifier)[4]$size
possible_leaves <- seq(2,n,1)
train_deviance <- c()
validation_deviance <- c()
for(i in 2:n) {
  prunedTree <- prune.tree(classifier, best=i)
  pred1 <- predict(prunedTree, newdata=dfValidation, type="tree",na.rm=TRUE)
  pred2 <- predict(prunedTree, newdata=dfTraining, type="tree",na.rm=TRUE)
  train_deviance[i] <- deviance(pred2,na.rm=TRUE)
  validation_deviance[i] <- deviance(pred1,na.rm=TRUE)
}
data <- data.frame(Leaves=vector("numeric"),
                   deviance=vector("numeric"))
for(i in 2:n)
{
  data[i,"Leaves"] <- i
  data[i,"deviance"] <- train_deviance[i]
  data[i,"set"] <- "train"
}

for(i in (n+1):(2*n))
{
  data[i,"Leaves"] <- i-n
  data[i,"deviance"] <- validation_deviance[i-n]
  data[i,"set"] <- "validation"

}
set.seed(12345)
ggplot(data = data ,aes(x = Leaves , y = deviance , col = set)) +
  geom_smooth() + labs(x="Leaves", y="Deviance", color="data set")
```

```r
optimal_tree <- prune.tree(classifier, best=4)
plot(optimal_tree)
text(optimal_tree, pretty=0)


##misclassification rate of test data
pr<-predict(classifier,newdata=dfTest,type="class")
conf_mat_test <- table(dfTest$good_bad,pr )

cat("missclassification rate for test data:", mean(pr!=dfTest$good_bad))

#question 4

library(e1071)
model <- naiveBayes(good_bad ~ ., data = dfTraining)
f1<-predict(model,newdata = dfTraining,type="class")
table(dfTraining$good_bad,f1)
cat("missclassification rate for navie train data:", mean(f1!=dfTraining$good_bad))


f2<-predict(model,newdata=dfTest,type="class")
table(dfTest$good_bad,f2)
cat("missclassification rate for navie test data:", mean(f2!=dfTest$good_bad))



#question5

#test data
# loss_matrix<-matrix(c(0,10,1,0),ncol=2)
set.seed(12345)
model2 <- naiveBayes(good_bad~.,data=dfTraining)
f1_1<-predict(model2,newdata=dfTest,type="raw")
a<-ifelse(f1_1[,2]/f1_1[,1] >10, 1, 0)
tab_test<-table(dfTest$good_bad,a)
accuracy_loss <- (sum(diag(tab_test)))/sum(tab_test)
mis_loss<-1-accuracy_loss
cat("missclassification rate for deviance_test:", mis_loss)


#training data
f1_2<-predict(model2,newdata=dfTraining,type="raw")
b<-ifelse(f1_2[,2]/f1_2[,1] >10, 1, 0)
tab_train<-table(dfTraining$good_bad,b)
accuracy_loss_train <- (sum(diag(tab_train)))/sum(tab_train)
mis_loss_train<-1-accuracy_loss_train
cat("missclassification rate for deviance_test:", mis_loss_train)


############################
```

‘‘‘