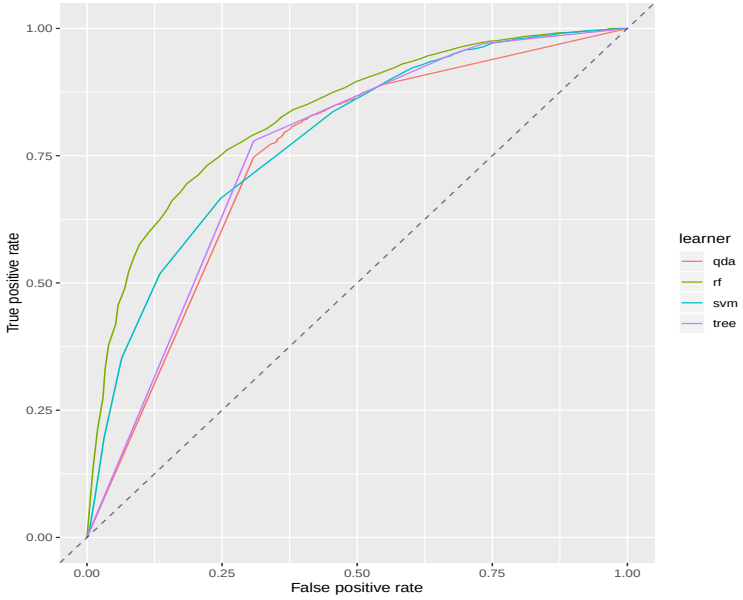
Data Mining Methods

Model Comparison



Prof. Dr. Christina Andersson

High Integrity Systems
Frankfurt University of Applied Sciences

Booklet to sprint 3

1 Getting Started

The focus in this sprint is on how to evaluate model performance. We haven't yet learned anything about the models, such as decision trees, themselves, i.e. concentrate on the model evaluation right now. Don't panic while you don't know what a desicion tree is yet. This will be explained in a later sprint.

2 Some Useful Resources

- Explanation of some classification measures for model performance. (We don't use python in this course, don't worry about the code): https://www.pluralsight.com/guides/evaluating-a-data-mining-model
- Another resource explaining classification measures: https://towardsdatascience.com/performance-measures-for-classification-models-a486c8976bf1
- A resource with a little bit more mathematics: https://sebastianraschka.com/blog/2016/model-evaluation-selection-part1.html
- Illustrating underfitting and overfitting: https://www.jeremyjordan.me/evaluating-a-machine-learning-model/

3 R

3.1 Packages

- **rpart**
- **rpart.plot**
- **ROCR**

3.2 Some Useful Commands

nrow	**set.seed**	**sample**	**setdiff**	**rpart**	**predict**
table	**read.csv**	**performance**	**plot**		

4 Exercises

1. (a) Load the dataset *iris* by typing *data(iris)*.
   (b) Load the packages *rpart* and *rpart.plot*. Use the command

```
iristree = rpart(Species ~ .,data=iris, method="class")
```

to construct a decision tree model for classifying iris flowers based on some input variables in the dataset *iris*. The target (response) variable is *Species*.

In this exercise, we're concentrating on learning different methods to assessing models. We're not discussing the data mining methods behind the modeling (such as understanding the building of a decision tree). This will come later.

(c) Divide the sample into a training data set with 2/3 of the observations and the rest in a test data set.

(d) Create a standard decision tree for the training data set.

(e) Make predictions with the decision tree model for the test data set.

(f) Take a look at the predicted values.

(g) Construct the confusion matrix. Interpretation?

2. Medical researchers want to explore the relationship between patient *age* and the presence (*disease*=1) or absence (*disease*=0) of a particular disease. Data are collected for 20 patients and stored in the data set *disease*.

(a) Read the data set *disease* into R, for example with the command *read.csv*.

(b) Use a decision tree (analogously to the previous exercise) to predict the class of the target variable for each of the 20 patients. Don't divide the data into training and validation data.

(c) Take a look at the predicted values.

(d) Construct the confusion matrix. Interpretation?

(e) Construct a decision tree again, but this time predict the probability of the target levels instead of the class. Take a look at the predicted values.

(f) Construct a ROC curve.

(g) Calculate the area under the ROC curve and interpret the result.

5 Solutions to Exercises

1. (a) Load the dataset *iris* by typing *data(iris)*.

Solution:

```
> data(iris)
```

(b) Load the packages *rpart* and *rpart.plot*. Use the command

```
iristree = rpart(Species ~ .,data=iris, method="class")
```

to construct a decision tree model for classifying iris flowers based on some input variables in the dataset *iris*. The target (response) variable is *Species*.

In this exercise, we're concentrating on learning different methods to assessing models. We're not discussing the data mining methods behind the modeling (such as understanding the building of a decision tree). This will come later.

Solution:

```
library(rpart)
library(rpart.plot)
iristree = rpart(Species ~ .,data=iris, method="class")
```

(c) Divide the sample into a training data set with 2/3 of the observations and the rest in a test data set.

Solution:

```
n = nrow(iris)
set.seed(42)
train.id = sample(n, size = 2/3*n)
test.id = setdiff(1:n, train.id)
```

(d) Create a standard decision tree for the training data set.

Solution:

```
traintree = rpart(Species ~ .,data=iris, method="class",
subset=train.id)
```

(e) Make predictions with the decision tree model for the test data set.

Solution:

```
> pred=predict(traintree, iris[-train.id,], type="class")
```

(f) Take a look at the predicted values.

Solution:

```
> head(pred)
     7     11     12     14     19     23
setosa setosa setosa setosa setosa setosa
Levels: setosa versicolor virginica
```

(g) Construct the confusion matrix. Interpretation?

Solution:

```
> table(pred, iris[-train.id, "Species"])

pred         setosa versicolor virginica
  setosa         13          0         0
  versicolor      0         16         1
  virginica       0          1        19
```

2. Medical researchers want to explore the relationship between patient *age* and the presence (*disease*=1) or absence (*disease*=0) of a particular disease. Data are collected for 20 patients and stored in the data set *disease*.

   (a) Read the data set *disease* into R, for example with the command *read.csv*.

   Solution:

```
> ff=read.csv(file="/your_path/disease.txt", head=TRUE, sep=",")
> ff
   Age Disease
1   25       0
2   29       0
3   30       0
4   31       0
5   32       0
6   41       0
7   41       0
8   42       0
9   44       1
10  49       1
11  50       0
12  59       1
13  60       0
14  62       0
15  68       1
16  72       0
17  79       1
18  80       0
19  81       1
20  84       1
```

   (b) Use a decision tree (analogously to the previous exercise) to predict the class of the target variable for each of the 20 patients. Don't divide the data into training and validation data.

   Solution:

```
diseasetree = rpart(Disease ~ .,data=ff, method="class")
pred=predict(diseasetree, ff, type="class")
```

   (c) Take a look at the predicted values.

   Solution:

```
> pred
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

```
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
Levels: 0 1
```

(d) Construct the confusion matrix. Interpretation?

Solution:

```
table(pred, ff[,"Disease"])
pred 0 1
   0 8 0
   1 5 7
```

(e) Construct a decision tree again, but this time predict the probability of the target levels instead of the class. Take a look at the predicted values.

Solution:

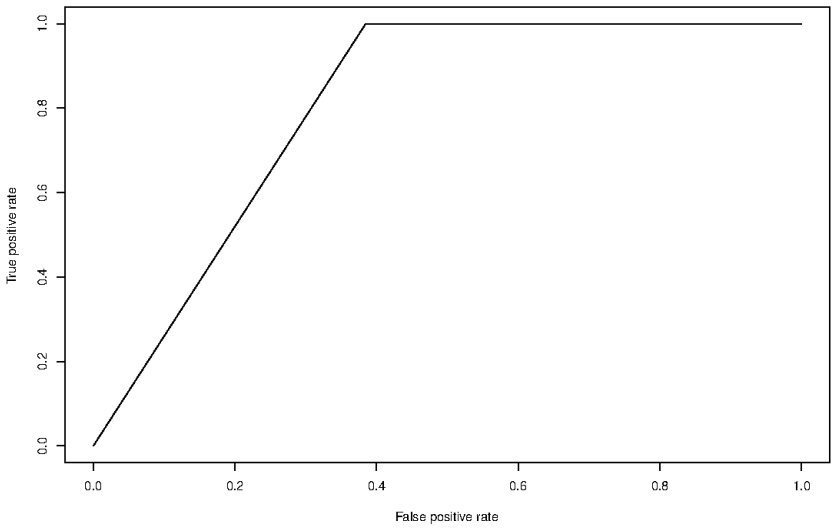
```
> diseasetree = rpart(Disease ~ .,data=ff, method="class")
> p2=predict(diseasetree, ff, type="prob")
> p2
           0         1
1  1.0000000 0.0000000
2  1.0000000 0.0000000
3  1.0000000 0.0000000
4  1.0000000 0.0000000
5  1.0000000 0.0000000
6  1.0000000 0.0000000
7  1.0000000 0.0000000
8  1.0000000 0.0000000
9  0.4166667 0.5833333
10 0.4166667 0.5833333
11 0.4166667 0.5833333
12 0.4166667 0.5833333
13 0.4166667 0.5833333
14 0.4166667 0.5833333
15 0.4166667 0.5833333
16 0.4166667 0.5833333
17 0.4166667 0.5833333
18 0.4166667 0.5833333
19 0.4166667 0.5833333
20 0.4166667 0.5833333
```

(f) Construct a ROC curve.

Solution:

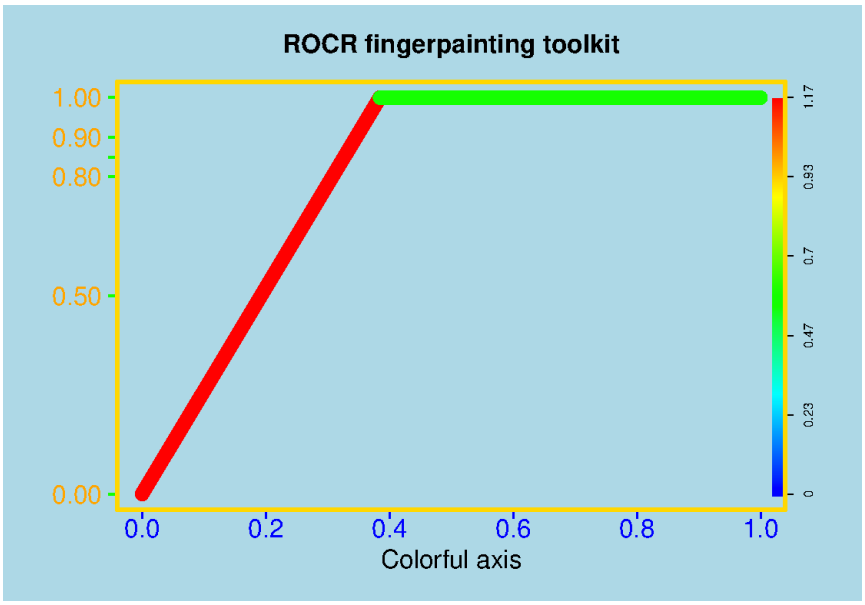
```
> pr3=p2[,2]
> pr = prediction(pr3, ff$Disease)
```

```
> perf = performance(pr,"tpr","fpr")
> plot(perf)
```



or if you want it a bit more colorful:

```
par(bg="lightblue", mai=c(1.2,1.5,1,1))
plot(perf, main="ROCR fingerpainting toolkit", colorize=TRUE,
xlab="Colorful axis",ylab="", box.lty=7, box.lwd=5,
box.col="gold", lwd=17, colorkey.relwidth=0.5,
xaxis.cex.axis=2,xaxis.col='blue',
xaxis.col.axis="blue", yaxis.col='green', yaxis.cex.axis=2,
yaxis.at=c(0,0.5,0.8,0.85,0.9,1),
yaxis.las=1, xaxis.lwd=2, yaxis.lwd=3,yaxis.col.axis="orange",
cex.lab=2, cex.main=2)
```



(g) Calculate the area under the ROC curve and interpret the result.

Solution:

```
> performance(pr,"auc")
An object of class "performance"
Slot "x.name":
[1] "None"

Slot "y.name":
[1] "Area under the ROC curve"

Slot "alpha.name":
[1] "none"

Slot "x.values":
list()

Slot "y.values":
[[1]]
[1] 0.8076923


Slot "alpha.values":
list()
```

Better than the random model.