Introduction to Machine Learning in R
NPFL 054

Easy homework assigned on April 14, 2021

Contact teacher: Martin Holub
holub@ufal.mff.cuni.cz

Both following tasks closely relates to the assigned obligatory Homework #2. You will use data set '`Caravan`', which is a part of *ISLR* package available in R. Your presentations should be helpful also for all other students who work on Homework #2.

Task 1 – Precision and Recall depending on the classification threshold

In this exercise you will focus on evaluation of a binary classification task. You will predict the value of binary attribute `Purchase`. First, split the 5,822 available development examples into training and test set, just the same way as in Homework #2, i.e. 1000 randomly selected examples for test, and the rest for training. Now, build a default Random Forest model (with default parameter setting) and with 300 trees, like

```
> RF.300 = randomForest(Purchase ~ ., caravan.train, ntree = 300)
```

a) Use your test set to evaluate the RF model and make a plot with three curves in one picture to show how three different performance measures depend on the classification (cut-off) threshold. The three curves will be *precision* (use red color), *recall* (black color), and *F1 measure* (blue color). There should be the cut-off on the x-axis, and the performance measures on the y-axis.

Technical hints:

- To deal with different cut-off values you need to use 'probability' type prediction, like
  ```
  > prediction.test.prob = predict(RF.300, caravan.test, type="prob")
  ```
- and then to use a particular threshold value `t`, like
  ```
  > prediction.test = factor(prediction.test.prob[ ,2] >= t,
                             levels = c("FALSE", "TRUE"))
  ```
- To plot the required curves you should compute confusion matrix for each cut-off value in a sequence like
  ```
  > cutoff.seq = seq(0, 1, by=0.001)
  ```
- To draw more curves into one plot in R, use first `plot()` function, and then `points()`.

b) Now use the same data and the same cut-off values to make a plot with *precision* and *recall*, how they depend on *FPR*. There should be *FPR* on the x-axis, and the other performance measures on the y-axis. *Recall* in black and *precision* in red color. The *recall* curve will be in fact the *ROC* curve.

c) Do the same plots for several different number of trees in the RF model.

Task 2 – Classifier precision and ROC

This exercise is more theoretical. Consider a binary classification task and the relationship between *precision* and *ROC* curve. Using a given test set, each point on the *ROC* corresponds to a particular cut-off setting and both *FPR* and *TPR* values were computed from a particular confusion matrix. To answer the following questions you should assume that you know the number of positive examples in the test set P, and the number of negative examples N as well, so that $P + N = |T|$, where $|T|$ is the test set size.

a) First, as a special case consider a diagonal *ROC*, i.e. *TPR* = *FPR*. What is the *precision* at different points on this special *ROC*? Answer it for *FPR* values 0.25, 0.5, and 0.75. Can you generalize from these three points?

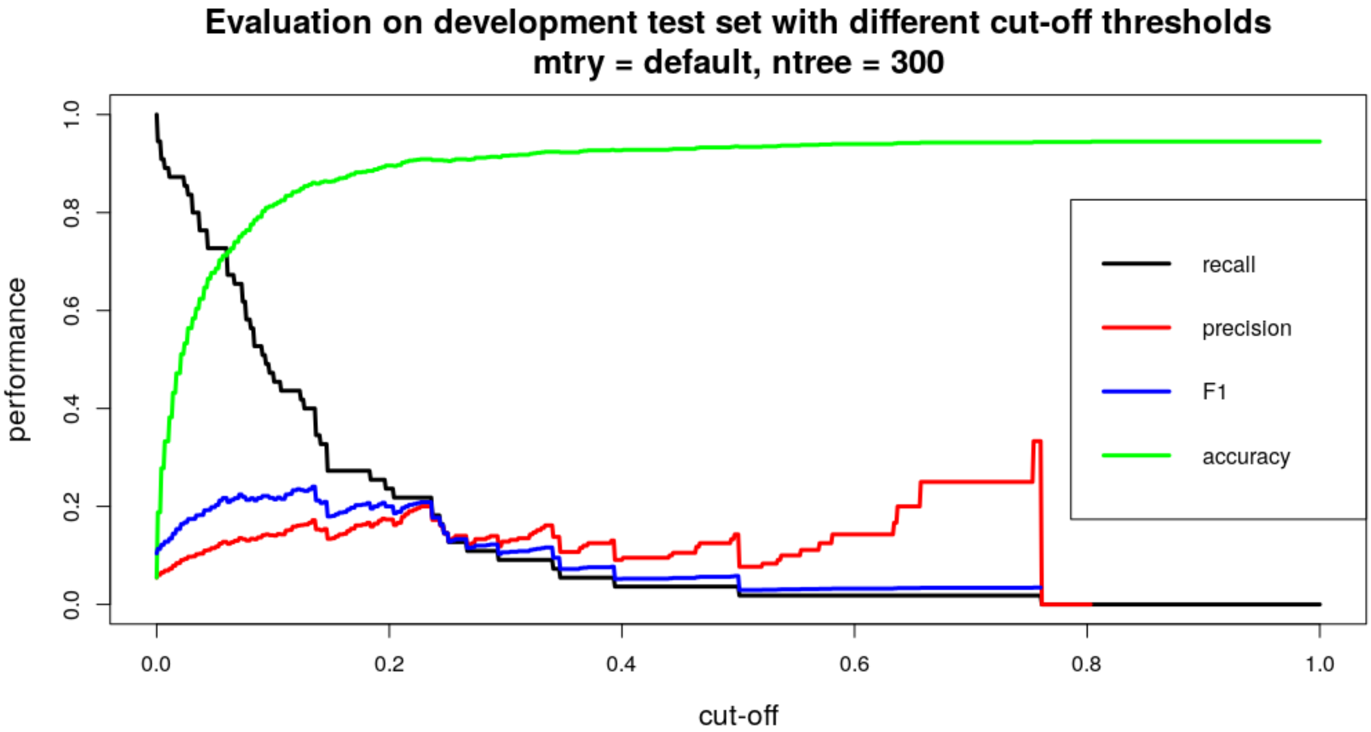
b) What would be the *ROC* shape if the (hypothetical) predictor had a fixed *precision* for different *FPR* values? Write a function in R that plots the *ROC* curve for a given fixed *precision*. The function should have three parameters, like

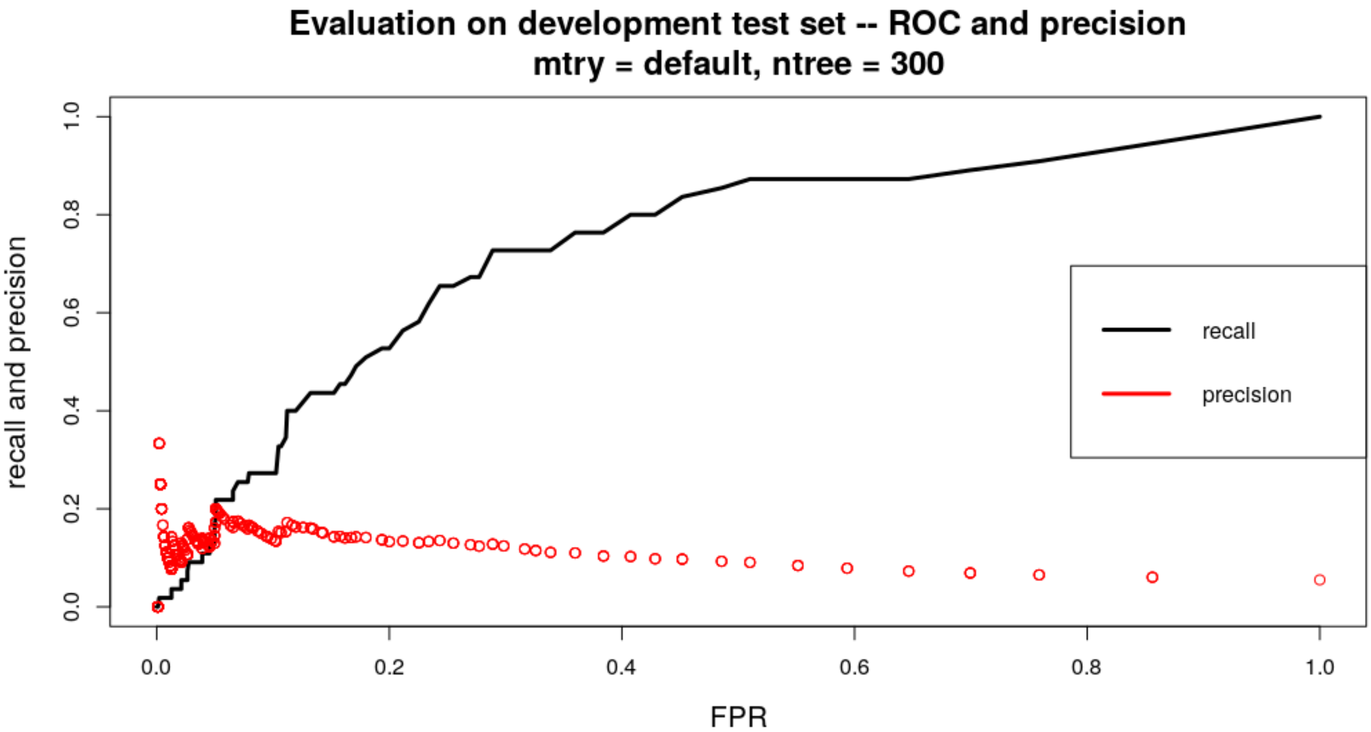
```
> plot_fixed_precision_roc = function(P, N, precision) { . . . }
```

Then, try to express the shape of this special *ROC* also analytically.

c) Now you should be already able to compute *precision* for any given pair of *FPR* and *TPR* values. Given some particular P and N, write a formula to compute *precision* for any given point at *ROC*.

Task 1 – Illustrations





Task 2 – Solution

Basic definitions and relationships between variables:

Observed variable	**Meaning (= the number of)**
P = TP + FN N = TN + FP P + N = |**T**|	positives in the test set negatives in the test set test set size
TP TN	true positives true negatives
FP FN	false positives false negatives

Performance measures	
TPR = TP/P	false positive rate = recall = sensitivity
FPR = FP/N	false positive rate
TP/(TP + FP)	precision

a) If TPR = FPR, then FP = TP * N/P. Thus precision = 1/(1 + N/P) = P/(P + N), which means that precision is constant at any point of the diagonal *ROC*.

Obviously, this poor level of precision would be reached even if the set of examples predicted as positives is selected as a random subset of the test set.

b) We have TP = TPR*P and FP = FPR*N, and thus

precision = TP/(TP + FP) = TPR*P/(TPR*P + FPR*N).

Hence

TPR = precision/(1 − precision) * N/P * FPR,

which is a linear function if precision is a fixed constant.

Since TPR is always ≤ 1, the maximum value of FPR is

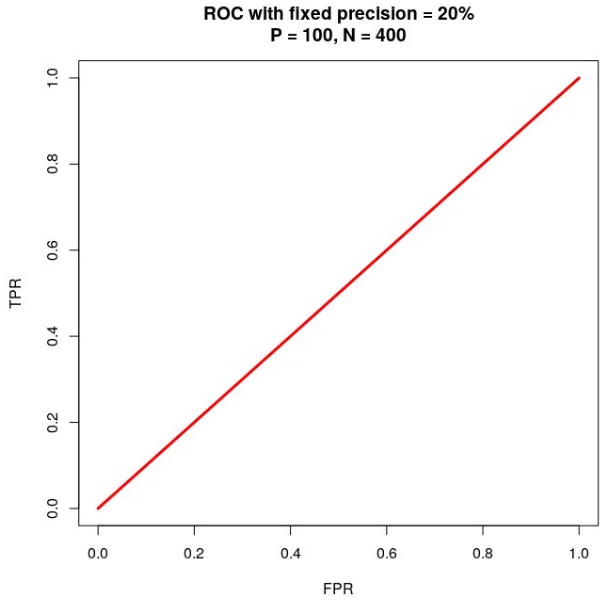
FPR_{max} = min(1, P/N * (1 − precision)/precision).

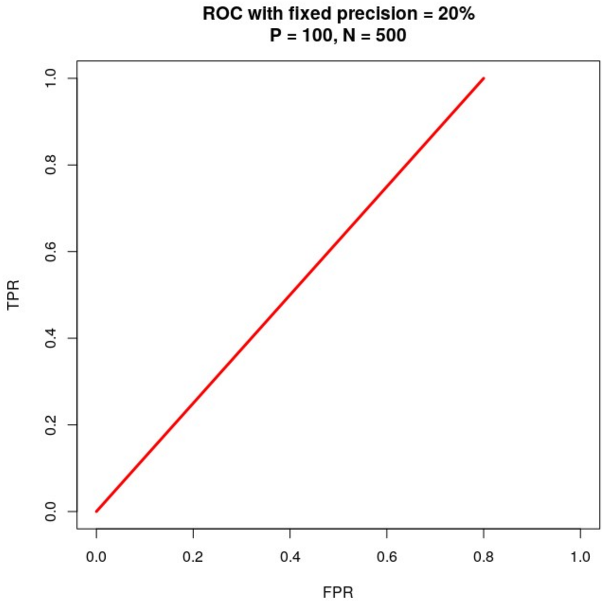
```
plot_fixed_precision_roc = function(P, N, precision){
      FPR_max = min(1, P/N * (1 - precision)/precision)
      TPR_max = min(1, precision/(1 - precision) * N/P)
      plot(  c(0, FPR_max), c(0, TPR_max), type="l", col="red", lwd=3,
             xlim=c(0,1), ylim=c(0,1),
             xlab="FPR", ylab="TPR",
             main=paste0("ROC with fixed precision = ",
                         round(100*precision,1), "%\n",
                         "P = ", P, ", N = ", N)
      )
}
```

c) As derived above, precision = TPR*P/(TPR*P + FPR*N).

Task 2 – Illustrations

```
> plot_fixed_precision_roc(100,400,0.2)
> plot_fixed_precision_roc(100,500,0.2)
```






```
> plot_fixed_precision_roc(100,500,0.6)
> plot_fixed_precision_roc(500,250,0.5)
```