ROCFunctions example usage

Anton Antonov
MathematicaForPrediction project at GitHub
MathematicaVsR project at GitHub
October 2016

Introduction

The package [2] provides Mathematica implementations of Receiver Operating Characteristic (ROC) functions calculation and plotting. The ROC framework is used for analysis and tuning of binary classifiers, [3]. (The classifiers are assumed to classify into a positive/true label or a negative/false label.)

The function `ROCFuntions` gives access to the individual ROC functions through string arguments. Those ROC functions are applied to special objects, called ROC Association objects.

Each ROC Association object is an `Association` that has the following four keys: "TruePositive", "FalsePositive", "TrueNegative", and "FalseNegative" .

Given two lists of actual and predicted labels a ROC Association object can be made with the function `ToROCAssociation` .

For more definitions and example of ROC terminology and functions see [3, 4].

Minimal example

Note that here although we use both of the provided Titanic training and test data, the code is doing only training. The test data is used to find the best tuning parameter (threshold) through ROC analysis.

Get packages

These commands load the packages [1,2]:

In[1]:=
```
Import[
 "https://raw.githubusercontent.com/antononcube/MathematicaForPrediction/master/
   MathematicaForPredictionUtilities.m"]
Import[
 "https://raw.githubusercontent.com/antononcube/MathematicaForPrediction/master/
   ROCFunctions.m"]
```

Using Titanic data

Here is the summary of the Titanic data used below:

In[3]:=
```
titanicData =
  (Flatten@*List) @@@ ExampleData[{"MachineLearning", "Titanic"}, "Data"];
columnNames = (Flatten@*List) @@
   ExampleData[{"MachineLearning", "Titanic"}, "VariableDescriptions"];
RecordsSummary[titanicData, columnNames]
```

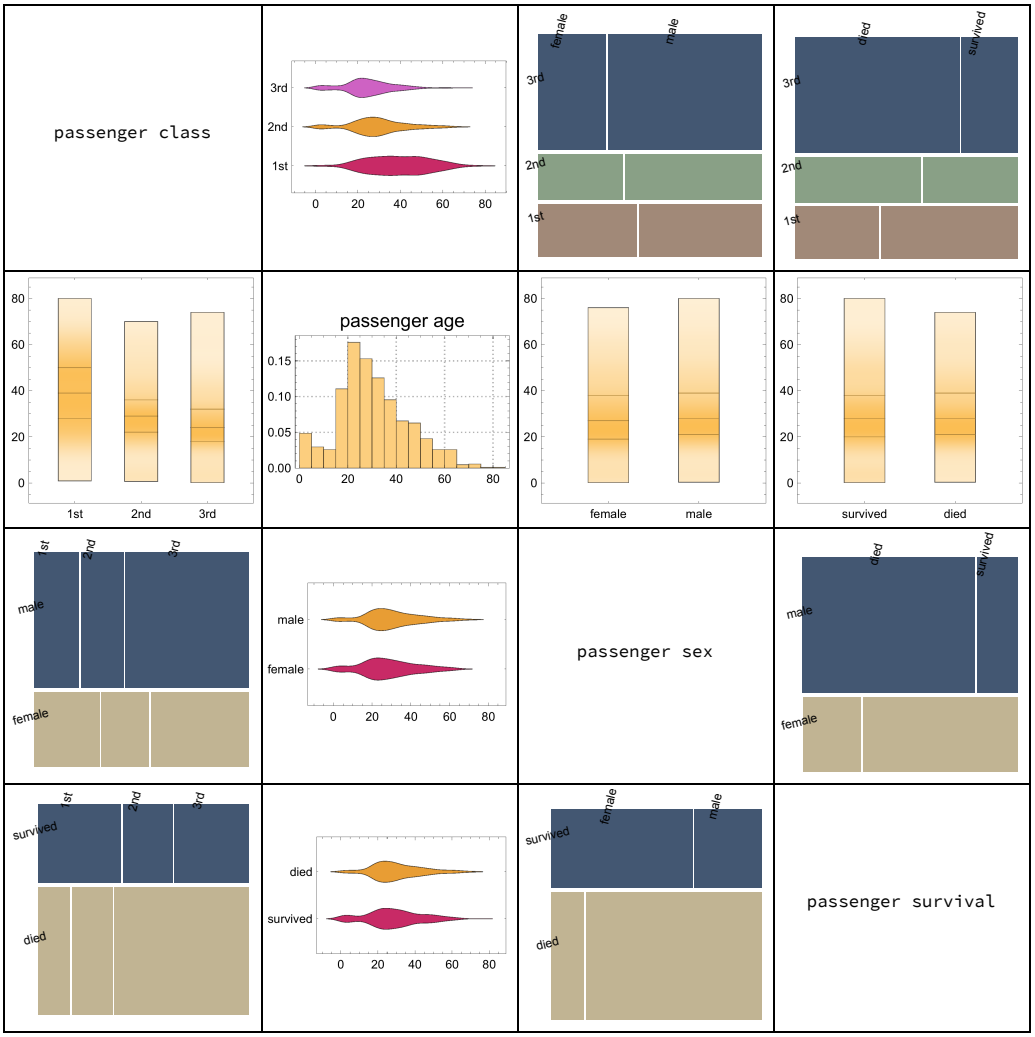
Out[5]=

1 passenger class		2 passenger age		3 passenger sex		4 passenger survival	
3rd	709	Missing[]	263	male	843	died	809
1st	323	24.	47	female	466	survived	500
2nd	277	22.	43				
		21.	41				
		30.	40				
		18.	39				
		(Other)	836				

This variable dependence grid shows the relationships between the variables.

In[6]:=
```
Magnify[#, 0.7] &@VariableDependenceGrid[titanicData, columnNames]
```

Out[6]=



Get training and testing data

In[7]:=
```
data = ExampleData[{"MachineLearning", "Titanic"}, "TrainingData"];
data = ((Flatten@*List) @@@ data)[[All, {1, 2, 3, -1}]];
trainingData = DeleteCases[data, {___, _Missing, ___}];
Dimensions[trainingData]
```

Out[10]= {732, 4}

In[11]:=
```
data = ExampleData[{"MachineLearning", "Titanic"}, "TestData"];
data = ((Flatten@*List) @@@ data)[[All, {1, 2, 3, -1}]];
testData = DeleteCases[data, {___, _Missing, ___}];
Dimensions[testData]
```

Out[14]= {314, 4}

Replace categorical with numerical values

In[15]:=
```
trainingData = trainingData /. {"survived" → 1,
    "died" → 0, "1st" → 0, "2nd" → 1, "3rd" → 2, "male" → 0, "female" → 1};
```

In[16]:=
```
testData = testData /. {"survived" → 1, "died" → 0,
    "1st" → 1, "2nd" → 2, "3rd" → 3, "male" → 0, "female" → 1};
```

Do linear regression

In[17]:=
```
lfm = LinearModelFit[{trainingData[[All, 1 ;; -2]], trainingData[[All, -1]]}]
```

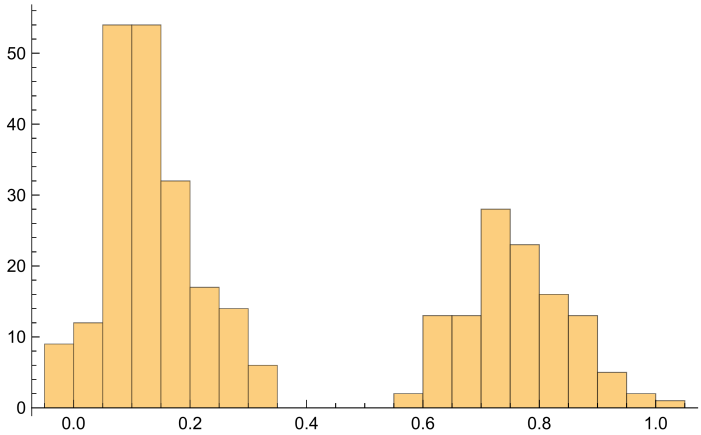
Out[17]= FittedModel[−0.0136025 #1 + 0.00515243 #2 + 0.634046 #3]

Get the predicted values

In[18]:=
```
modelValues = lfm @@@ testData[[All, 1 ;; -2]];
```

In[19]:=
```
Histogram[modelValues, 20]
```

Out[19]=



In[20]:=
```
RecordsSummary[modelValues]
```

Out[20]=
```
1 column 1
Min      -0.0386604
1st Qu   0.103461
Median   0.18827
Mean     0.363972
3rd Qu   0.71834
Max      1.01203
```

Obtain ROC associations over a set of parameter values

In[21]:=
```
testLabels = testData[[All, -1]];
```

In[22]:=
```
thRange = Range[0.1, 0.9, 0.025];
aROCs = Table[ToROCAssociation[{0, 1},
    testLabels, Map[If[# > θ, 1, 0] &, modelValues]], {θ, thRange}];
```

Evaluate ROC functions for given ROC association

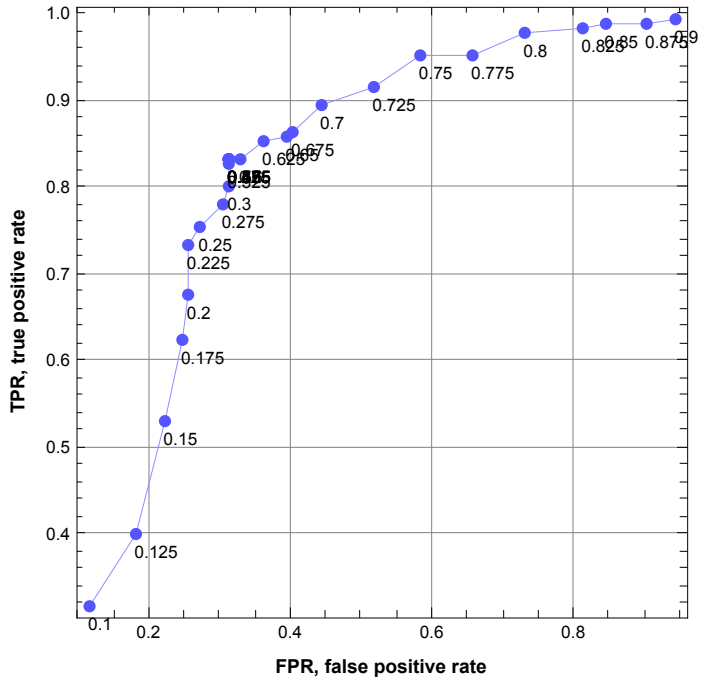
In[24]:=
```
Through[ROCFunctions[{"PPV", "NPV", "TPR", "ACC", "SPC"}][aROCs[[3]]]]
```

Out[24]=
$$\left\{\frac{34}{43}, \frac{19}{37}, \frac{17}{32}, \frac{197}{314}, \frac{95}{122}\right\}$$

Standard ROC plot

In[25]:=
```
ROCPlot[thRange, aROCs, "PlotJoined" → Automatic,
 "ROCPointCallouts" → True, "ROCPointTooltips" → True, GridLines → Automatic]
```

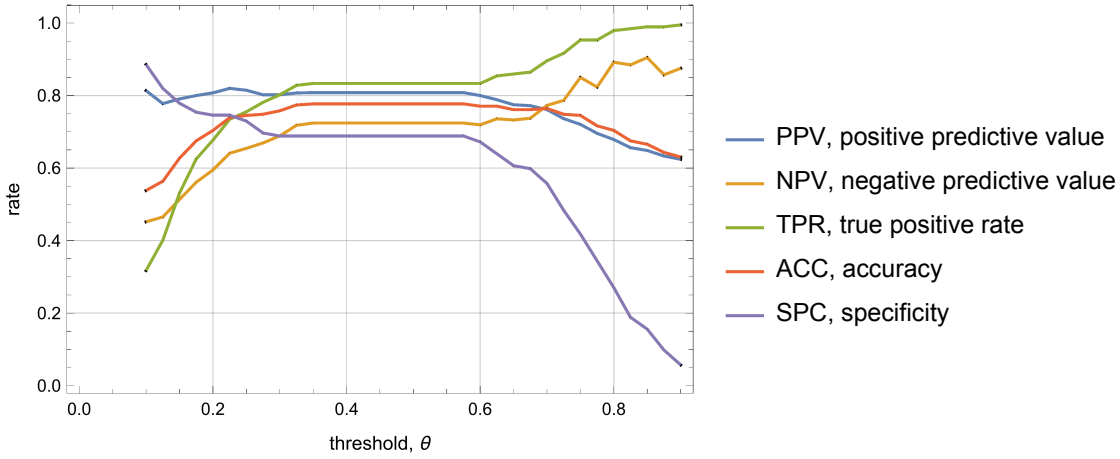
Out[25]=



Plot ROC functions wrt to parameter values

In[26]:=
```
ListLinePlot[Map[Transpose[{thRange, #}] &, Transpose[
   Map[Through[ROCFunctions[{"PPV", "NPV", "TPR", "ACC", "SPC"}][#]] &, aROCs]]],
 Frame → True, FrameLabel → Map[Style[#, Larger] &, {"threshold, θ", "rate"}],
 PlotLegends → Map[# <> ", " <> (ROCFunctions["FunctionInterpretations"][#]) &,
   {"PPV", "NPV", "TPR", "ACC", "SPC"}], GridLines → Automatic]
```

Out[26]=



Finding the intersection point of PPV and TPR

We want to find a point that provides balanced positive and negative labels success rates. One way to do this is to find the intersection point of the ROC functions PPV (positive predictive value) and TPR (true positive rate).

Examining the plot above we can come up with the initial condition for x.

In[27]:=
```
ppvFunc = Interpolation[Transpose@{thRange, ROCFunctions["PPV"] /@ aROCs}];
tprFunc = Interpolation[Transpose@{thRange, ROCFunctions["TPR"] /@ aROCs}];
FindRoot[ppvFunc[x] - tprFunc[x] == 0, {x, 0.2}]
```

Out[29]=
```
{x → 0.3}
```

Area under the ROC curve

The Area Under the ROC curve (AUROC) tells for a given range of the controlling parameter “what is the probability of the classifier to rank a randomly chosen positive instance higher than a randomly chosen negative instance, (assuming ‘positive’ ranks higher than ‘negative’)”, [3,4]

Calculating AUROC is easy using the Trapezoidal quadrature formula:

In[30]:=
```
N@Total[Partition[
    Sort@Transpose[{ROCFunctions["FPR"] /@ aROCs, ROCFunctions["TPR"] /@ aROCs}],
    2, 1] /. {{x1_, y1_}, {x2_, y2_}} :> (x2 - x1) (y1 + (y2 - y1) / 2)]
```

Out[30]=
```
0.698685
```

It is also implemented in [2]:

In[31]:=
```
N@ROCFunctions["AUROC"][aROCs]
```

Out[31]=
```
0.698685
```

References

[1] Anton Antonov, MathematicaForPrediction utilities, (2014), source code MathematicaForPrediction at GitHub, package MathematicaForPredictionUtilities.m.

[2] Anton Antonov, Receiver operating characteristic functions Mathematica package, (2016), source code MathematicaForPrediction at GitHub, package ROCFunctions.m .

[3] Wikipedia entry, Receiver operating characteristic. URL: http://en.wikipedia.org/wiki/Receiver_operating_characteristic .

[4] Tom Fawcett, An introduction to ROC analysis, (2006), Pattern Recognition Letters, 27, 861–874. (Link to PDF.)