Linear regression with ROC

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

Introduction

This document demonstrates how to do in Mathematica linear regression (easily using the built-in function `LinearModelFit`) and to tune the binary classification with the derived model through the so called Receiver Operating Characteristic (ROC) framework, [5].

The data used in this document is from [1] and it has been analyzed in more detail in [2]. In this document we only show to how to ingest and do very basic analysis of that data before proceeding with the linear regression model and its tuning. The package [4] provides the needed ROC functionalities.

Used packages

These commands load the packages [3,4]:

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

Reading data

The code below imports the data.

```
In[3]:= lines = Import["~/Datasets/adult/adult.data"];
        lines = Select[lines, Length[#] > 3 &];
        Dimensions[lines]

Out[5]= {32 561, 15}
```

```
In[6]:= linesTest = Import["~/Datasets/adult/adult.test"];
        linesTest = Select[linesTest, Length[#] > 3 &];
        Dimensions[linesTest]

Out[8]= {16 281, 15}
```

```
In[9]:= columnNames = StringSplit[
          "age,workclass,fnlwgt,education,education-num,marital-status,occupation,
            relationship,race,sex,capital-gain,capital-loss,hours-per-week,native-
            country", ","]

Out[9]= {age, workclass, fnlwgt, education, education-num, marital-status, occupation,
         relationship, race, sex, capital-gain, capital-loss, hours-per-week, native-country}

In[10]:= AppendTo[columnNames, "income"]

Out[10]= {age, workclass, fnlwgt, education, education-num,
          marital-status, occupation, relationship, race, sex, capital-gain,
          capital-loss, hours-per-week, native-country, income}

In[11]:= aColumnNames = AssociationThread[columnNames → Range[Length[columnNames]]];
```

In[12]:=
```
Magnify[#, 0.6] &@GridTableForm[lines[[1 ;; 12]],
  TableHeadings → Map[Style[#, Blue, FontFamily → "Times"] &, columnNames]]
```

Out[12]=

#	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	inco
1	39	State-gov	77 516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=
2	50	Self-emp-not-inc	83 311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=
3	38	Private	215 646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=
4	53	Private	234 721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=
5	28	Private	338 409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=
6	37	Private	284 582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	0	0	40	United-States	<=
7	49	Private	160 187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0	0	16	Jamaica	<=
8	52	Self-emp-not-inc	209 642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States	>5
9	31	Private	45 781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14 084	0	50	United-States	>5
10	42	Private	159 449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	5178	0	40	United-States	>5
11	37	Private	280 464	Some-college	10	Married-civ-spouse	Exec-managerial	Husband	Black	Male	0	0	80	United-States	>5
12	30	State-gov	141 297	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	India	>5

In[13]:=
```
Magnify[#, 0.6] &@GridTableForm[linesTest[[1 ;; 12]],
  TableHeadings → Map[Style[#, Blue, FontFamily → "Times"] &, columnNames]]
```

Out[13]=

#	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	incom
1	25	Private	226 802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	40	United-States	<=5
2	38	Private	89 814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	50	United-States	<=5
3	28	Local-gov	336 951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	40	United-States	>50
4	44	Private	160 323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	7688	0	40	United-States	>50
5	18	?	103 497	Some-college	10	Never-married	?	Own-child	White	Female	0	0	30	United-States	<=5
6	34	Private	198 693	10th	6	Never-married	Other-service	Not-in-family	White	Male	0	0	30	United-States	<=5
7	29	?	227 026	HS-grad	9	Never-married	?	Unmarried	Black	Male	0	0	40	United-States	<=5
8	63	Self-emp-not-inc	104 626	Prof-school	15	Married-civ-spouse	Prof-specialty	Husband	White	Male	3103	0	32	United-States	>50
9	24	Private	369 667	Some-college	10	Never-married	Other-service	Unmarried	White	Female	0	0	40	United-States	<=5
10	55	Private	104 996	7th-8th	4	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	10	United-States	<=5
11	65	Private	184 454	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	6418	0	40	United-States	>50
12	36	Federal-gov	212 465	Bachelors	13	Married-civ-spouse	Adm-clerical	Husband	White	Male	0	0	40	United-States	<=5

Assignment of training and testing data

As usual in classification and regression problems we work with two data sets: a training data set and a testing data set. Here we split the original training set into two sets training set and tuning set. The tuning set is going to be used to find a good value of a tuning parameter through ROC.

Training data

In[14]:=
```
data = lines;
data⟦All, -1⟧ = Map[StringTrim, data⟦All, -1⟧];
```

In[16]:=
```
trainingInds = RandomSample[Range[Length[data]], Ceiling[Length[data] * 0.75]];
tuningInds = Complement[Range[Length[data]], trainingInds];
trainingData = data⟦trainingInds⟧;
tuningData = data⟦tuningInds⟧;
```

Testing data

In[20]:=
```
testData = linesTest;
testData⟦All, -1⟧ = Map[StringDrop[StringTrim[#], -1] &, testData⟦All, -1⟧];
```

Some preliminary data analysis

Before doing regression it is a good idea to do some preliminary analysis of the data. For that we are going to use functions defined in the package [3].

In[22]:=
```
Magnify[#, 0.7] &@Grid[ArrayReshape[RecordsSummary[data, columnNames], {3, 5}],
  Dividers → All, Alignment → {Left, Top}]
```

Out[22]=

1 age Min 17 1st Qu 28 Median 37 Mean 1 256 257 / 32 561 3rd Qu 48 Max 90	2 workclass Private 22 696 Self-emp-not-inc 2541 Local-gov 2093 ? 1836 State-gov 1298 Self-emp-inc 1116 (Other) 981	3 fnlwgt Min 12 285 1st Qu 471 297 / 4 Median 178 356 Mean 6 179 373 392 / 32 561 3rd Qu $\frac{474109}{2}$ Max 1 484 705	4 education HS-grad 10 501 Some-college 7291 Bachelors 5355 Masters 1723 Assoc-voc 1382 11th 1175 (Other) 5134	5 education-num Min 1 1st Qu 9 Median 10 Mean 328 237 / 32 561 3rd Qu 12 Max 16
6 marital-status Married-civ-spouse 14 976 Never-married 10 683 Divorced 4443 Separated 1025 Widowed 993 Married-spouse-absent 418 Married-AF-spouse 23	7 occupation Prof-specialty 4140 Craft-repair 4099 Exec-managerial 4066 Adm-clerical 3770 Sales 3650 Other-service 3295 (Other) 9541	8 relationship Husband 13 193 Not-in-family 8305 Own-child 5068 Unmarried 3446 Wife 1568 Other-relative 981	9 race White 27 816 Black 3124 Asian-Pac-Islander 1039 Amer-Indian-Eskimo 311 Other 271	10 sex Male 21 790 Female 10 771
11 capital-gain 1st Qu 0 3rd Qu 0 Median 0 Min 0 Mean $\frac{35089324}{32561}$ Max 99 999	12 capital-loss 1st Qu 0 3rd Qu 0 Median 0 Min 0 Mean $\frac{2842700}{32561}$ Max 4356	13 hours-per-week Min 1 1st Qu 40 Median 40 Mean 1 316 684 / 32 561 3rd Qu 45 Max 99	14 native-country United-States 29 170 Mexico 643 ? 583 Philippines 198 Germany 137 Canada 121 (Other) 1709	15 income <=50K 24 720 >50K 7841

In[23]:=
```
Magnify[#, 0.7] &@Grid[ArrayReshape[RecordsSummary[testData, columnNames], {3, 5}],
  Dividers → All, Alignment → {Left, Top}]
```

Out[23]=

1 age Min 17 1st Qu 28 Median 37 Mean 210 391 / 5427 3rd Qu 48 Max 90	2 workclass Private 11 210 Self-emp-not-inc 1321 Local-gov 1043 ? 963 State-gov 683 Self-emp-inc 579 (Other) 482	3 fnlwgt Min 13 492 1st Qu 466 865 / 4 Median 177 831 Mean 3 084 202 270 / 16 281 3rd Qu 238 384 Max 1 490 400	4 education HS-grad 5283 Some-college 3587 Bachelors 2670 Masters 934 Assoc-voc 679 11th 637 (Other) 2491	5 education-num Min 1 1st Qu 9 Median 10 Mean 163 997 / 16 281 3rd Qu 12 Max 16
6 marital-status Married-civ-spouse 7403 Never-married 5434 Divorced 2190 Widowed 525 Separated 505 Married-spouse-absent 210 Married-AF-spouse 14	7 occupation Prof-specialty 2032 Exec-managerial 2020 Craft-repair 2013 Sales 1854 Adm-clerical 1841 Other-service 1628 (Other) 4893	8 relationship Husband 6523 Not-in-family 4278 Own-child 2513 Unmarried 1679 Wife 763 Other-relative 525	9 race White 13 946 Black 1561 Asian-Pac-Islander 480 Amer-Indian-Eskimo 159 Other 135	10 sex Male 10 860 Female 5421
11 capital-gain 1st Qu 0 3rd Qu 0 Median 0 Min 0 Mean $\frac{5871499}{5427}$ Max 99 999	12 capital-loss 1st Qu 0 3rd Qu 0 Median 0 Min 0 Mean $\frac{1431088}{16281}$ Max 3770	13 hours-per-week Min 1 1st Qu 40 Median 40 Mean $\frac{657626}{16281}$ 3rd Qu 45 Max 99	14 native-country United-States 14 662 Mexico 308 ? 274 Philippines 97 Puerto-Rico 70 Germany 69 (Other) 801	15 income <=50K 12 435 >50K 3846

Looking at the column "income" we can see that for both datasets the people who earn more than \$50000 is ≈ 25% of all people. We will consider ">50" to be the more important class label for the classifiers built below.

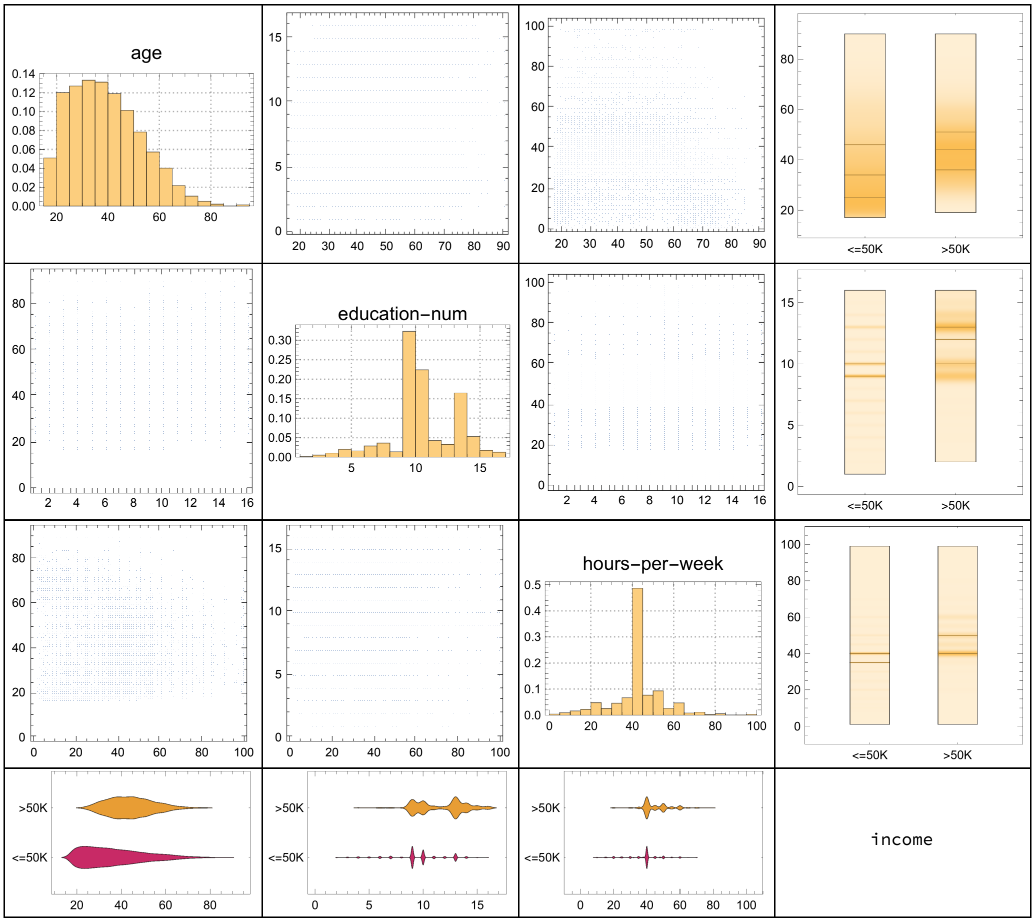
For simplicity of the exposition below we are going to use only the columns "age", "education-num", "hours-per-week", "income".

In[24]:=
```
columnNamesExplantoryVars = {"age", "education-num", "hours-per-week"};
columnNameResponseVar = "income";
columnNamesForAnalysis =
  Append[columnNamesExplantoryVars, columnNameResponseVar];
```

Here is variable dependence grid for those variables:

In[27]:=
```
Magnify[#, 0.7] &@VariableDependenceGrid[
  data⟦All, aColumnNames /@ columnNamesForAnalysis⟧, columnNamesForAnalysis]
```

Out[27]=



We can see from the last row of the plot above that the variables “age”, “education-num”, “hours-per-week” can explain “income” at least to a degree. We see that higher values of “age”, “education-num”, “hours-per-week” are associated closer with “>50K”. For more detailed analysis see [2].

LinearModelFit

`LinearModelFit` has several signatures. Doing Linear regression over the data we have is most convenient with the signature `LinearModelFit[{m,v}]` (using a design matrix *m* and a response vector *v*.)

As mentioned above in order to keep the exposition simple we do the regression with the three numerical columns ”age”, “education-num”, and “hours-per-week”. With the replacement rules `{"<=50K"→ 0,">50K"→1}` we convert the data column "income" into a vector of 0's and 1's. The result of `LinearModelFit` is a function based on the training set of data.

In[28]:=
```
lfmFunc = LinearModelFit[
  {trainingData⟦All, aColumnNames /@ {"age", "education-num", "hours-per-week"}⟧,
   trainingData⟦All, aColumnNames["income"]⟧ /. {"<=50K" → 0, ">50K" → 1}}]
```

Out[28]=
```
FittedModel[ 0.00187859 #1 + 0.0167414 #2 + 0.000599763 #3 ]
```

We use the model from the training data on the test data:

In[29]:=
```
tuningLables = tuningData⟦All, aColumnNames["income"]⟧ /. {"<=50K" → 0, ">50K" → 1};
```

Next we are going to evaluate what is the classification success of the derived model.

In[30]:=
```
modelValues = lfmFunc @@@
   tuningData⟦All, aColumnNames /@ {"age", "education-num", "hours-per-week"}⟧;
```

In[31]:=
```
modelValues⟦1 ;; 12⟧
```

Out[31]=
```
{0.246049, 0.294229, 0.218629, 0.342148, 0.275292, 0.326165,
 0.217739, 0.315059, 0.266714, 0.218629, 0.16063, 0.256483}
```

Although the response vector given to `LinearModelFit` is of 0's and 1's the regression model values are reals within a smaller than [0, 1] range.

In[32]:=
```
RecordsSummary[modelValues]
```

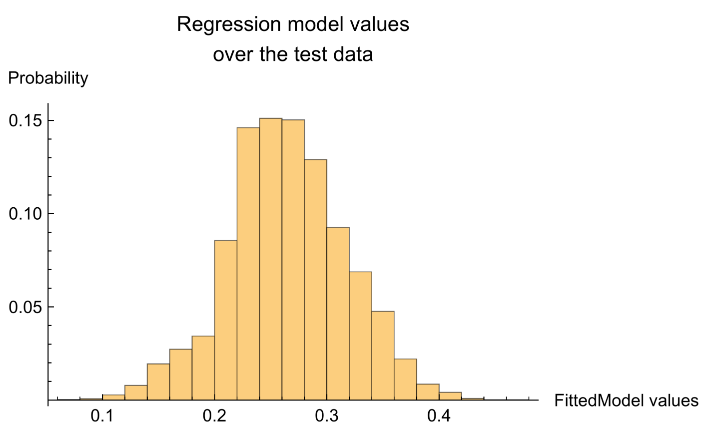
Out[32]=

1 column 1	
Min	0.0689454
1st Qu	0.229504
Median	0.263319
Mean	0.264793
3rd Qu	0.299186
Max	0.463377

Here is a histogram of values from the regression model:

In[33]:=
```
Histogram[modelValues, Automatic, "Probability",
 PlotLabel → "Regression model values\nover the test data",
 AxesLabel → {"FittedModel values", "Probability"}]
```

Out[33]=



We pick a threshold above which the model values are considered to be 1's (and hence ">=50K").

In[34]:=
```
With[{θ = 0.3}, modelLabels = Map[If[# > θ, 1, 0] &, modelValues]];
```

In[35]:=
```
modelLabels⟦1 ;; 12⟧
```

Out[35]=
```
{0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0}
```

Here is a table that shows classification success of the regression model with chosen threshold:

In[36]:=
```
labelsROC = SortBy[Tally[Transpose[{tuningLables, modelLabels}]], First];
labelsROC = Flatten /@
   MapThread[Append, {labelsROC, labelsROC⟦All, 2⟧ / Total[labelsROC⟦All, 2⟧] // N}];
TableForm[labelsROC, TableHeadings →
  {{"true negative", "false positive", "false negative", "true positive"},
   {"test labels", "model labels", "freq", "%"}}]
```

Out[38]//TableForm=

	test labels	model labels	freq	%
true negative	0	0	5218	0.641032
false positive	0	1	953	0.117076
false negative	1	0	927	0.113882
true positive	1	1	1042	0.12801

We want to determine the threshold that gives the best classification success. What is “best” can be viewed and determined in several ways. We are going to use the so called Receiver Operating Characteristic (ROC); see [5].

(The table above is similar to the confusion matrix produced by *Mathematica*’s function `Classify` made available through `ClassifierMeasurements`. See the documentation for these function.)

LinearModelFit with ROC

In this section we take a more systematic approach of determining the best threshold to be used to separate the regression model values.

We are going to call ***positive*** the income values “>50K” and ***negative*** the income values “<=50K”. Again see [2]. As we mentioned above, we will consider “>50” to be the more important class label for the classifiers built below.

For the ROC functionalities are employed through the package [4].

Linear regression classification definitions

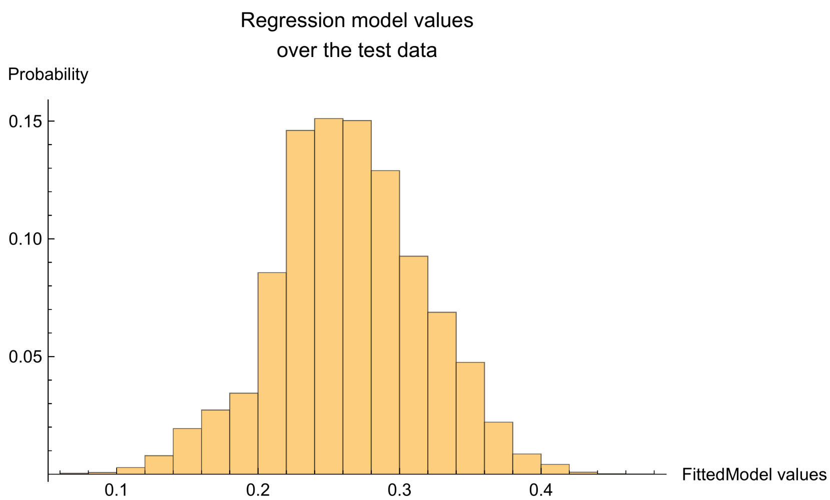
In[39]:=
```
Clear[ModelLabelsByThreshold]
ModelLabelsByThreshold[modelValues_?VectorQ, θ_?NumberQ] :=
  Map[If[# > θ, 1, 0] &, modelValues];
ModelLabelsByThreshold[lfmFunc_FittedModel, testData_?MatrixQ,
   aColumnNames_Association, θ_?NumberQ] :=
  Block[{t, testLables, modelValues, modelLabels},
   testLables = testData[[All, aColumnNames["income"]]] /. {"<=50K" → 0, ">50K" → 1};
   modelValues = lfmFunc @@@
     testData[[All, aColumnNames /@ {"age", "education-num", "hours-per-week"}]];
   ModelLabelsByThreshold[modelValues, testLables, θ]
  ];
```

Computations to find the best threshold

Looking at the histogram of classification values:

In[42]:=
```
Histogram[modelValues, Automatic, "Probability",
 PlotLabel → "Regression model values\nover the test data",
 AxesLabel → {"FittedModel values", "Probability"}]
```

Out[42]=



we decide to use the following a range of thresholds:

In[43]:=
```
thRange = Range[0.15, 0.42, 0.01];
```

Compute the model values

In[44]:=
```
AbsoluteTiming[
 tuningLabels =
  testData⟦All, aColumnNames[columnNameResponseVar]⟧ /. {"<=50K" → 0, ">50K" → 1};
 modelValues = lfmFunc @@@ testData⟦All, aColumnNames /@ columnNamesExplantoryVars⟧;
]
```

Out[44]= {1.7642, Null}

Compute the ROC data for each threshold of the range of thresholds, and convert the ROC data to associations convenient for the computation of ROC functions:

In[45]:=
```
aROCs = Table[ToROCAssociation[{1, 0}, tuningLabels,
    ModelLabelsByThreshold[modelValues, θ]], {θ, thRange}];
```

Here is an example of evaluating some of the ROC functions over one of the elements of `aROCs`:

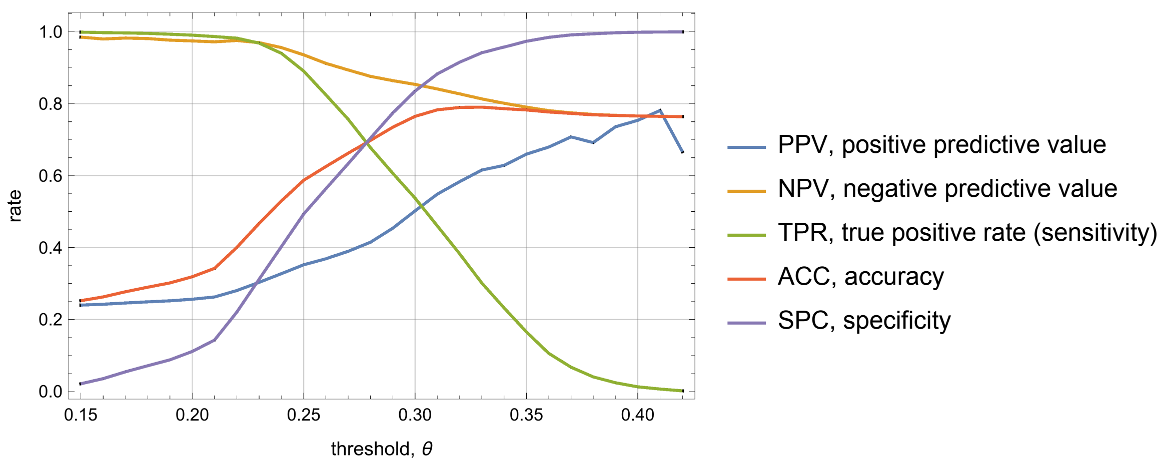
In[46]:=
```
Through[ROCFunctions[{"PPV", "NPV", "ACC"}][aROCs⟦3⟧]]
```

Out[46]= $\left\{\frac{1917}{7796}, \frac{677}{689}, \frac{4511}{16281}\right\}$

Plot of the functions PPV, NPV, TPR, SPC, ACC:

In[47]:=
```
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[47]=



Using the plot above we can select a threshold for separating the model values into 0's and 1's ("<=50K" and ">50K" respectively). We can see that when the classification accuracy increases the positive predictive value decreases. A good threshold value is 0.3 because TPR and ACC have satisfactory, large enough values.

Let us find the intersection point of the curves PPV and TPR. Examining the plot above we can come up with the initial condition for *x*.

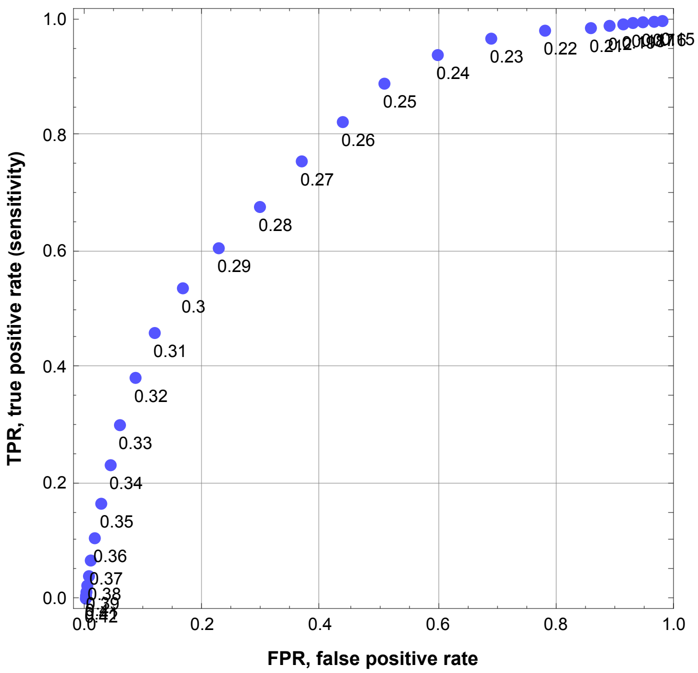
In[48]:=
```
Clear[θ]
ppvFunc = Interpolation[Transpose@{thRange, ROCFunctions["PPV"] /@ aROCs}];
tprFunc = Interpolation[Transpose@{thRange, ROCFunctions["TPR"] /@ aROCs}];
sol = FindRoot[ppvFunc[θ] - tprFunc[θ] == 0, {θ, 0.25}]
```

Out[51]=
```
{θ → 0.302858}
```

Here is a different plot used typically in ROC analysis:

In[52]:=
```
ROCPlot[thRange, aROCs, "PlotJoined" → False,
 "ROCPointCallouts" → True, "ROCPointTooltips" → True, GridLines → Automatic]
```

Out[52]=



Accuracy over the test data

We split the original training data into two parts for training and tuning. Using the found threshold, let us use evaluate the classification process over the test data.

In[53]:=
```
modelValues = lfmFunc @@@ testData⟦All, aColumnNames /@ columnNamesExplantoryVars⟧;
modelLabels = Map[If[# > (θ /. sol), ">50K", "<=50K"] &, modelValues];
```

Using the actual labels and the predicted labels (`modelLabels`) let us find the contingency values with `ToROCAssociation`:

In[55]:=
```
clRes = ToROCAssociation[{">50K", "<=50K"},
  testData⟦All, aColumnNames[columnNameResponseVar]⟧, modelLabels]
```

Out[55]=
```
<|TruePositive → 1983, FalsePositive → 1882,
 TrueNegative → 10 553, FalseNegative → 1863|>
```

Using the above result let us compute all of the ROC functions:

In[56]:=
```
GridTableForm[
 Transpose[{ROCFunctions["FunctionNames"], N@Through[ROCFunctions[][clRes]]}],
 TableHeadings → {"name", "value"}]
```

Out[56]=

#	name	value
1	TPR	0.515601
2	SPC	0.848653
3	PPV	0.513066
4	NPV	0.849952
5	FPR	0.151347
6	FDR	0.486934
7	FNR	0.484399
8	ACC	0.769977

Using a built-in classifier

If we use one of the built-in classifiers we can see that we got better results for the important class label “>50K” through the ROC analysis using an inferior base classifier (linear regression).

In[57]:=
```
AbsoluteTiming[cf = Classify[data⟦All, aColumnNames /@ columnNamesExplantoryVars⟧ →
     data⟦All, aColumnNames[columnNameResponseVar]⟧, Method → "NeuralNetwork"];
 cfPredictedLabels = cf /@ testData⟦All, aColumnNames /@ columnNamesExplantoryVars⟧;
]
```

Out[57]= {24.3918, Null}

In[58]:=
```
cfCLRes = ToROCAssociation[{">50K", "<=50K"},
   testData⟦All, aColumnNames[columnNameResponseVar]⟧, cfPredictedLabels]
GridTableForm[Transpose[{ROCFunctions["FunctionNames"],
    N@Through[ROCFunctions[][cfCLRes]]}], TableHeadings → {"name", "value"}]
```

Out[58]= ⟨|TruePositive → 1497, FalsePositive → 874,
TrueNegative → 11 561, FalseNegative → 2349|⟩

Out[59]=

#	name	value
1	TPR	0.389236
2	SPC	0.929715
3	PPV	0.631379
4	NPV	0.831129
5	FPR	0.0702855
6	FDR	0.368621
7	FNR	0.610764
8	ACC	0.802039

References

[1] Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science. Census Income Data Set, URL: http://archive.ics.uci.edu/ml/datasets/Census+Income .

[2] Anton Antonov, "Classification and association rules for census income data", (2014), MathematicaForPrediction at WordPress.com , URL: https://mathematicaforprediction.wordpress.com/2014/03/30/-classification-and-association-rules-for-census-income-data/ .

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

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

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