PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R

Jan Grau [1], Ivo Grosse [1,2] and Jens Keilwagen [3]

[1]Institute of Computer Science, Martin Luther University Halle–Wittenberg, Halle (Saale), Germany

[2]German Centre for Integrative Biodiversity Research (iDiv) Halle-Jena-Leipzig, Leipzig, Germany

[3]Julius Kühn-Institut (JKI) - Federal Research Centre for Cultivated Plants, Quedlinburg, Germany

grau@informatik.uni-halle.de

This package computes the areas under the precision-recall (PR) and ROC curve for weighted (e.g., soft-labeled) and unweighted data. In contrast to other implementations, the interpolation between points of the PR curve is done by a non-linear piecewise function. In addition to the areas under the curves, the curves themselves can also be computed and plotted by a specific S3-method. Users should be aware of the small sample size problem [1].
We thank Toby Dylan Hocking for suggesting the use of `cumsum` *for computing ROC and PR curves.*

1 ROC and PR curves for hard-labeled data

We first consider an example, where the classification task is to distinguish data points originating from two different classes (termed positive/negative or foreground/background). In the example, we assume that the test data set contains 300 data points from the foreground (positive) and 500 data points from the background (negative) class.
To make this example running R code, we generate classification scores by drawing values from two different Gaussian distributions:

```
> fg<-rnorm(300);
> bg<-rnorm(500,-2);
```

In a real application, however, `fg` would contain the classification scores of our classifier for each of the 300 foreground data points, and `bg` would contain the classification scores for each of the 500 background data points.
With the classification scores for these data points at hand, we can now use the functions `roc.curve` and `pr.curve` of the PRROC R-package to compute the area under the ROC and the area under the PR curve of our classifier:

```
> roc<-roc.curve(scores.class0 = fg, scores.class1 = bg)
> pr<-pr.curve(scores.class0 = fg, scores.class1 = bg)
```

Evaluating the resulting object for AUC-ROC in R, we get printed the AUC value

```
> roc

  ROC curve

    Area under curve:
     0.9162667

    Curve not computed ( can be done by using curve=TRUE )
```

and the output reminds us that, as of now, we only computed AUC-ROC, but not the ROC curve itself.
The result for the AUC-PR object is similar

```
> pr

  Precision-recall curve

    Area under curve (Integral):
     0.8777665

    Area under curve (Davis & Goadrich):
     0.8777661

    Curve not computed ( can be done by using curve=TRUE )
```

but prints out two different AUC-PR values, one using the interpolation of Davis & Goadrich [2] and one using the continuous interpolation of Boyd *et al.* [3] and Keilwagen *et al.* [4].
To also compute the ROC and the PR curve, we add a parameter `curve = TRUE` to both functions:

```
> roc<-roc.curve(scores.class0 = fg, scores.class1 = bg, curve = TRUE)
> pr<-pr.curve(scores.class0 = fg, scores.class1 = bg, curve = TRUE)
```

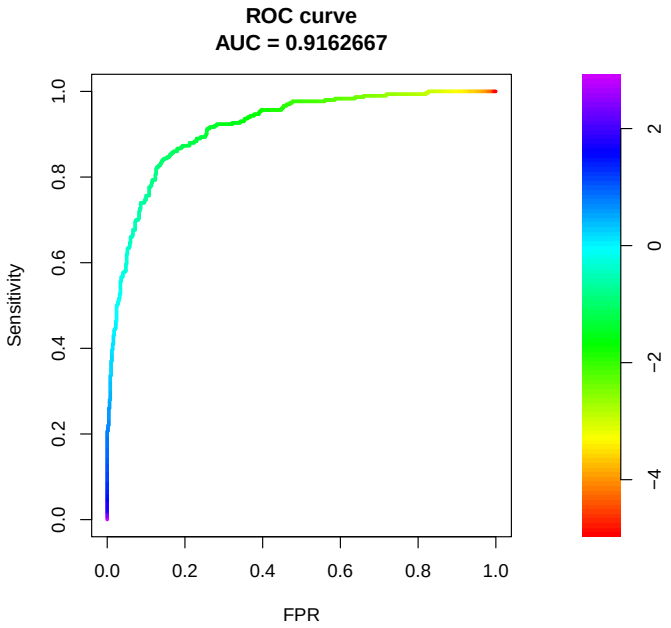
Printing the results now also shows that both curves have been determined:

```
> roc

  ROC curve

    Area under curve:
     0.9162667

    Curve for scores from  -4.935784  to  2.888259
    ( can be plotted with plot(x) )

> pr

  Precision-recall curve

    Area under curve (Integral):
     0.8777665

    Area under curve (Davis & Goadrich):
     0.8777661

    Curve for scores from  -4.935784  to  2.888259
    ( can be plotted with plot(x) )
```

We can now use the object for the ROC curve to obtain a plot of the curve

```
> plot(roc)
```

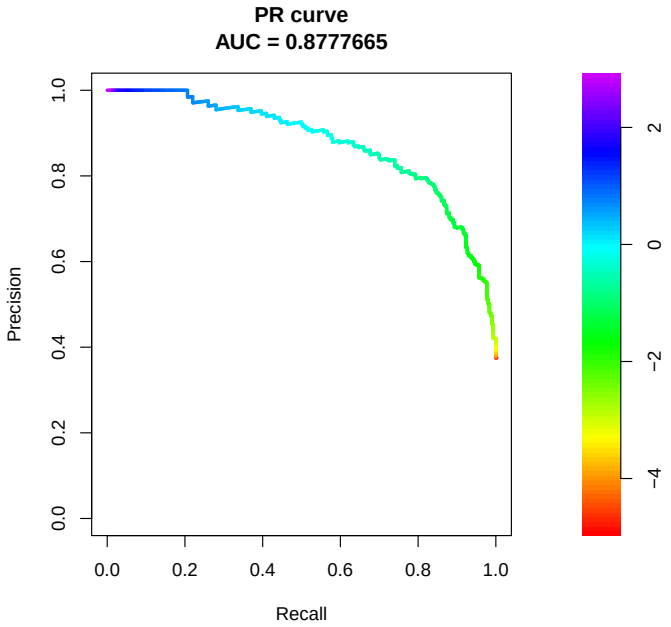
and get the following plot:



The color scale on the right side of the plot gives an indication, which classification threshold results in a certain point on the curve, i.e., a certain pair of sensitivity and false positive rate.
In complete analogy, we call `plot` for the PR curve

```
> plot(pr)
```

and obtain



As an alternative interface to the `roc.curve` and `pr.curve` functions for hard-labeled data, we can provide a joint vector of classification scores together with a vector of class labels, where a value of 1 means that a data point belongs to the foreground (positive) class and a value of 0 corresponds to the background (negative) class.
Here, we simulate this scenario by concatenating the previous two score arrays and generating a label vector `lab` with the corresponding class labels:

```
> x<-c(fg,bg);
> lab<-c(rep(1,length(fg)),rep(0,length(bg)))
```

We call `roc.curve` and `pr.curve` with the joint vector `x` specified for parameter `scores.class0` and the label vector specified for parameter `weights.class0`

```
> roc<-roc.curve(scores.class0 = x, weights.class0 = lab);
> pr<-pr.curve(scores.class0 = x, weights.class0 = lab);
```

and obtain exactly the same AUC-ROC and AUC-PR values as before:

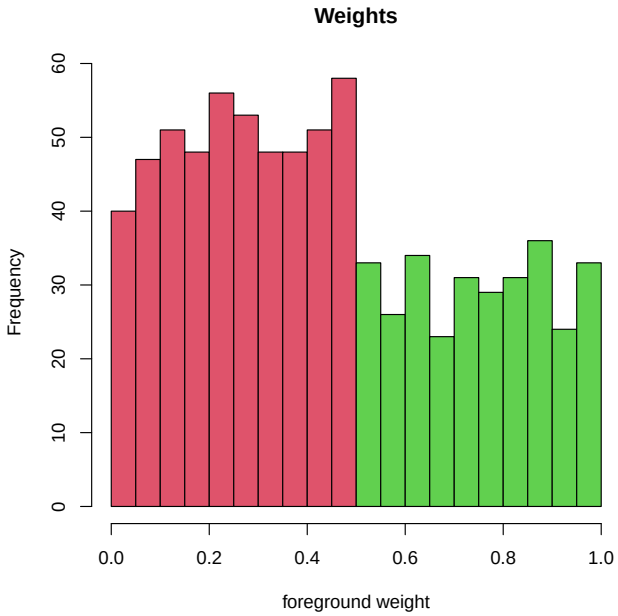
```
> roc

  ROC curve

    Area under curve:
     0.9162667

    Curve not computed ( can be done by using curve=TRUE )

> pr

  Precision-recall curve

    Area under curve (Integral):
     0.8777665

    Area under curve (Davis & Goadrich):
     0.8777661

    Curve not computed ( can be done by using curve=TRUE )
```

2 ROC and PR curves for soft-labeled data

In bioinformatics applications, the separation of data points into two classes is often not as clear as implied by a hard-labeling, where each data point either belongs to the foreground class or belongs to the background class. For instance, class separation might be based on some measurement (i.e., intensities on a micorarray distinguishing active from inactive genes) and hard-labeling forces us to use a threshold on the measured values, where foreground data points are on one side of the threshold and background data points on the other side of the threshold.
This setting seems to be arbitrary as the distance to the threshold is not reflected in the class label. Hence, if two data points are located on the same side of the threshold, they have the same impact on the classification performance, no matter if one of these points is very close to the threshold and the other is far away. In contrast, two data points that are very close to each other but on different sides of the threshold will be treated as completely different.
One possibility to circumvent these problems is soft-labeling, where each data point is assigned a probability of belonging to the foreground class and the converse probability of belonging to the negative class.
Here, we simulate such a scenario by drawing foreground probabilities for the foreground data points from the interval $(0.5, 1)$ and foreground probabilities for the negative data points from the interval $(0, 0.5)$.

```
> wfg<- c(runif(300,min=0.5,max=1),runif(500,min=0,max=0.5))
```

The distribution of the generated foreground probabilities of the foreground data points (green) and background data points (red) is shown in the following histogram.



In real applications, we would generate foreground probabilities from measurement values, for instance by applying an appropriately parameterized logistic function.
Given a joint vector of classification scores `x` as in the previous section and the just generated foreground probabilities, we compute the ROC and PR curve and the areas under these curves given the soft-labels by providing the scores as parameter `scores.class0` and the corresponding foreground probabilities as `weights.class0`:

```
> wroc<-roc.curve(scores.class0 = x, weights.class0 = wfg, curve = TRUE)
> wpr<-pr.curve(scores.class0 = x, weights.class0 = wfg, curve = TRUE)
```

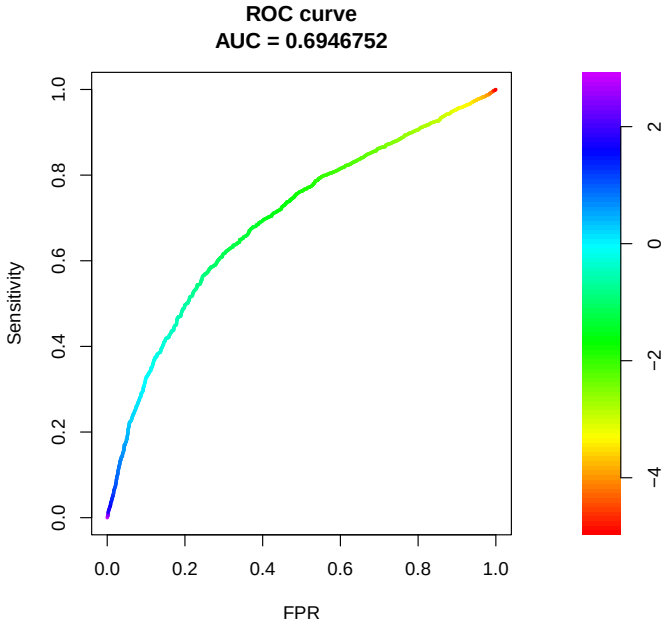
Internally, both functions of the PRROC R-package assume that the scores of the background data points are identical to those of the foreground data points (since each data point belongs to both classes with a certain probability) and that the background probabilities are just the converse probabilities of the foreground probabilities.
Hence, the following two calls yield exactly the same results as the previous ones:

```
> wroc<-roc.curve(scores.class0 = x, scores.class1 = x,
+   weights.class0 = wfg, weights.class1 = 1-wfg, curve = TRUE)
> wpr<-pr.curve(scores.class0 = x, scores.class1 = x,
+   weights.class0 = wfg,weights.class1 = 1-wfg, curve = TRUE)
```

Again, we can plot the ROC curve given these soft-labels using the `plot` function

```
> plot(wroc)
```

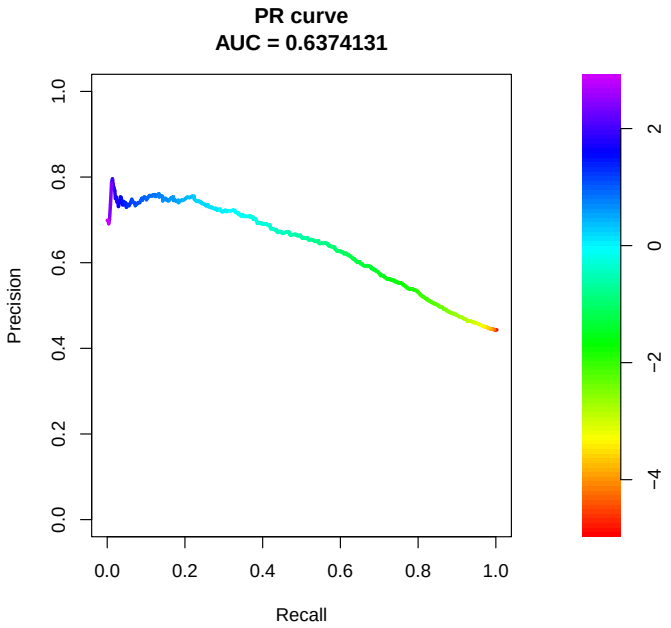
and obtain the following plot.



We proceed for the PR curve given the soft-labels in exactly the same manner

```
> plot(wpr)
```

yielding the following plot.



For PR curves, the minimal possible PR curve is not equal to a straight line at precision = 0. For soft-labeled data, neither the PR curve nor the ROC curve typically reach a maximum AUC value of 1 or a minimum AUC value of 0.
To allow for a better impression of the (relative) performance of a classifier at hand, the PRROC package can also compute the maximum curve and its AUC value (parameter `max.compute = T`), the minimum curve and its AUC value (`min.compute = T`) and the curve and AUC value of a random classifier (`rand.compute = T`).

```
> wpr<-pr.curve(scores.class0 = x, weights.class0 = wfg, curve = TRUE,
+   max.compute = T, min.compute = T, rand.compute = T)
> wroc<-roc.curve(scores.class0 = x, weights.class0 = wfg, curve = TRUE,
+   max.compute = T, min.compute = T, rand.compute = T)
```

This also provides relative AUC values, i.e., the minimal AUC subtracted from the original AUC and the result divided by the difference of maximum and minimum AUC, when evaluating the PR and ROC curve objects in R:

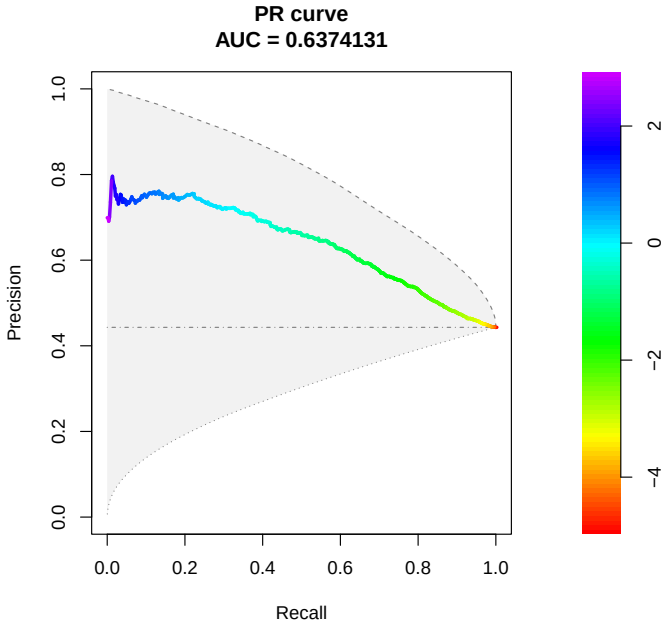
```
> wpr

  Precision-recall curve

    Area under curve (Integral):
     0.6374131

    Relative area under curve (Integral):
     0.6832175

    Area under curve (Davis & Goadrich):
    cannot be computed for weighted data

    Curve for scores from  -4.935784  to  2.888259
    ( can be plotted with plot(x) )



    Maximum AUC:
     0.7987053   NA


    Minimum AUC:
     0.2895479   NA


    AUC of a random classifier:
     0.4431009   0.4431009
```

```
> wroc

  ROC curve

    Area under curve:
     0.6946752

    Relative area under curve:
     0.8007416

    Curve for scores from  -4.935784  to  2.888259
    ( can be plotted with plot(x) )



    Maximum AUC:
     0.8236585


    Minimum AUC:
     0.1763415


    AUC of a random classifier:
     0.5
```

If computed, the maximum and minimum curve and the curve of the random classifier may be included into the PR curve (and ROC curve) plots using parameters `max.plot`, `min.plot`, and `rand.plot`, respectively. In addition, the area between maximum and minimum curve may be shaded.

```
> plot(wpr,max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE,
+   fill.area = TRUE)
```

This procedure gives the following plot.



Often, we not only want to assess the performance of a single classifier, but want to compare the performance of different alternative classifiers for a given problem.
Here, we again generate classification scores of a second classifier from Gaussian distributions and compute ROC and PR curve for this classifier as well:

```
> y<-c(rnorm(300,sd=2),rnorm(500,-5,sd=2))
> wpr2<-pr.curve(scores.class0 = y, weights.class0 = wfg, curve = TRUE,
+   max.compute = TRUE, min.compute = TRUE, rand.compute = TRUE)
> wroc2<-roc.curve(scores.class0 = y, weights.class0 = wfg, curve = TRUE,
+   max.compute = TRUE, min.compute = TRUE, rand.compute = TRUE)
```

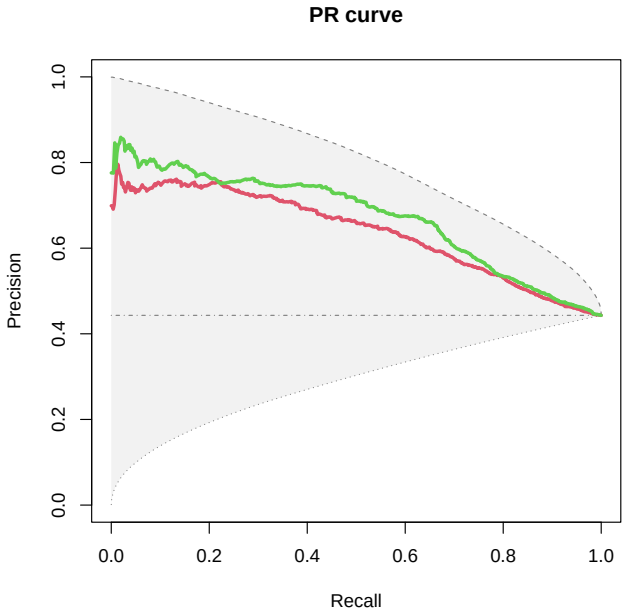
Now, we can first plot the curve (the PR curve in this case) of the first classifier and assign a color to this curve using the parameter `color`. In addition, we might want to switch of the reporting of the AUC value in the title of the plot, since after adding a second curve, it may be unclear, which curve this value refers to.

```
> plot(wpr, max.plot = TRUE, min.plot = TRUE, rand.plot = TRUE,
+   fill.area = T, color=2, auc.main = FALSE);
```

Afterwards, we can add the curve for the second classifier using the parameter `add = TRUE` and specify a color for this curve as well.

```
> plot(wpr2, add = TRUE, color = 3);
```

Using the two plots commands above, we obtain a plot with two PR curves, one in red for the first classifier and one in green for the second classifier.

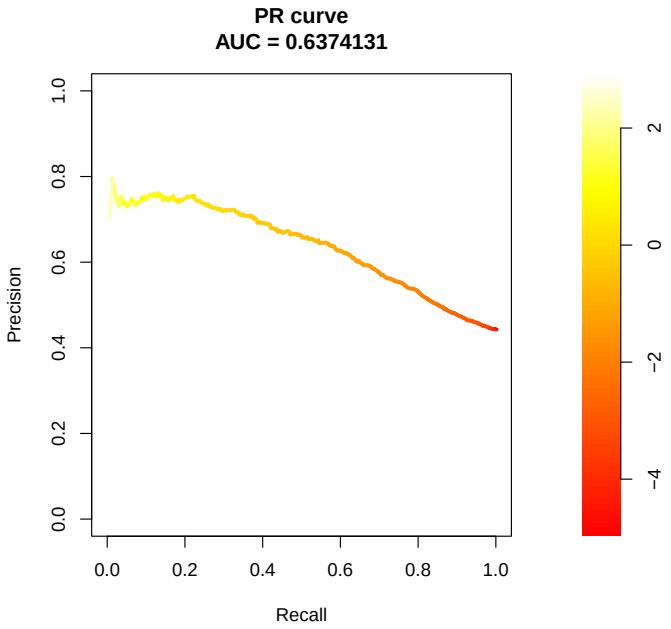


Subsequently, low-level R plotting functions like adding a legend with `legend` may be applied.

3 Parameters of the `plot` function

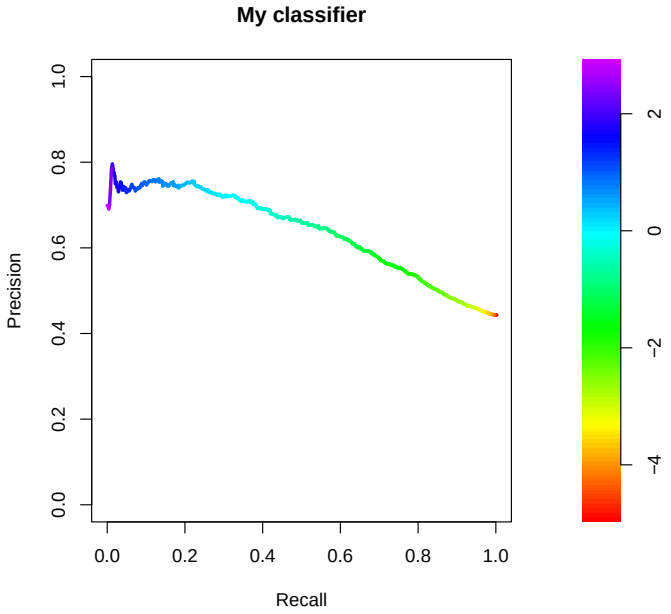
The `plot` function of the PRROC package has several additional parameters controlling the appearance of ROC and PR curve plots.
We can specify the colors for the color scale on the threshold values using the parameter `scale.color`.

```
> plot(wpr,scale.color = heat.colors(100));
```



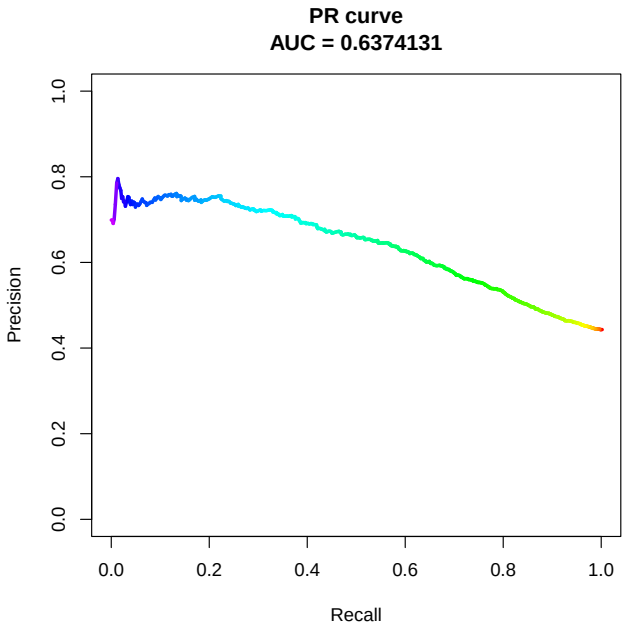
We can change the title of the plot with parameter `main` and choose whether to show the AUC value using parameter `auc.main`.

```
> plot(wpr, auc.main = FALSE, main = "My classifier")
```



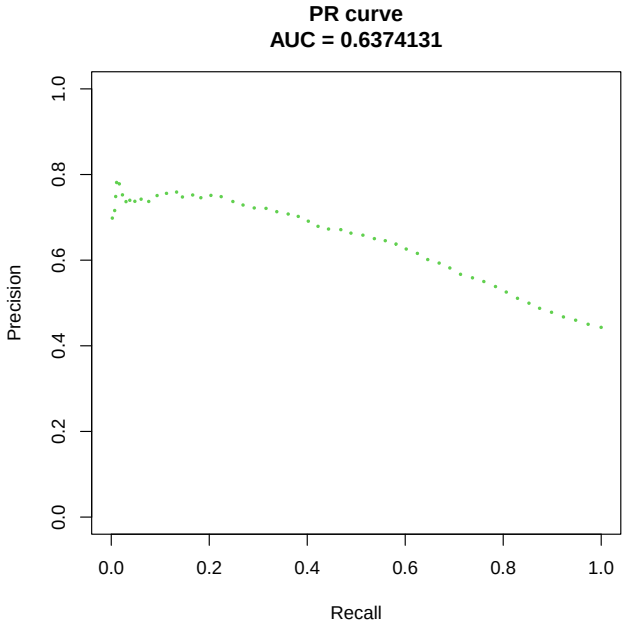
We can switch off the color scale using `legend = FALSE`.

```
> plot(wpr, legend = FALSE)
```



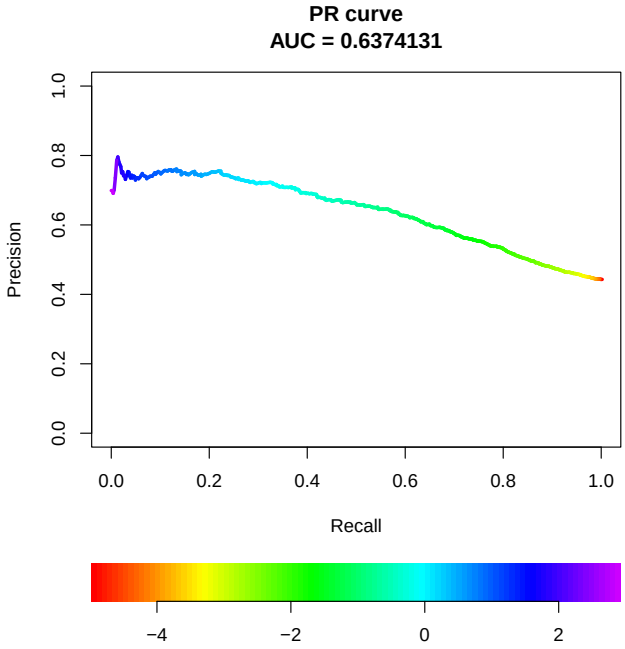
We can modify the color of a curve using the parameter `color` (which automatically switches of the color scale) and modify, for instance, the line type of the curve using standard R `par` parameters.

```
> plot(wpr, color=3, lty="dotted");
```



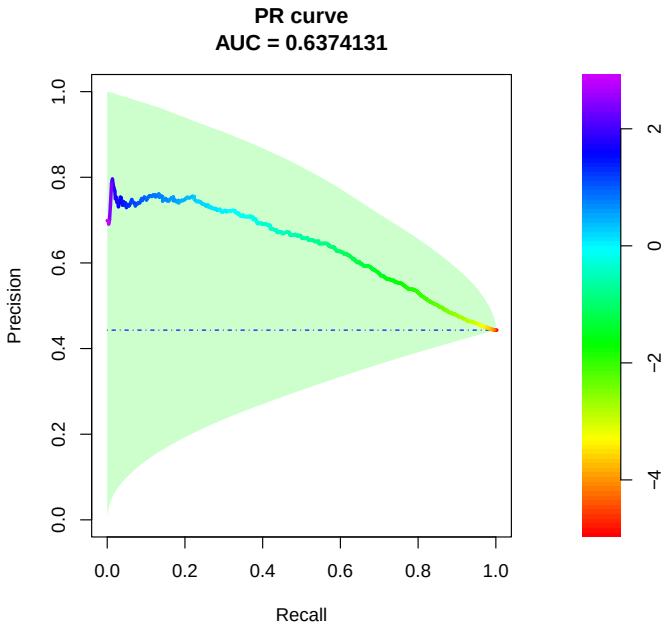
We can change the location of the color scale by specifying the border with parameter `legend`, where the numbers 1 to 4 have the same meaning as `axis` in standard R (1: bottom, 2: left, 3: top, 4: right).

```
> plot(wpr,legend=1);
```



And we can modify the color of the shading between maximum and minimum curve (parameter `fill.color`) and of the additional (maximum, minimum, random) curves (`maxminrand.col`).

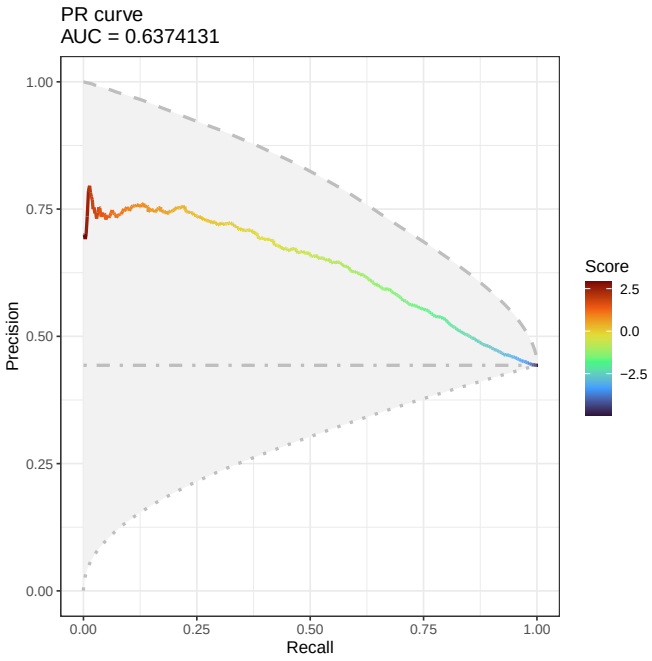
```
> plot(wpr, rand.plot = TRUE, fill.area = TRUE,
+   fill.color = rgb(0.8,1,0.8), maxminrand.col = "blue" );
```



4 Plotting curves using ggplot2

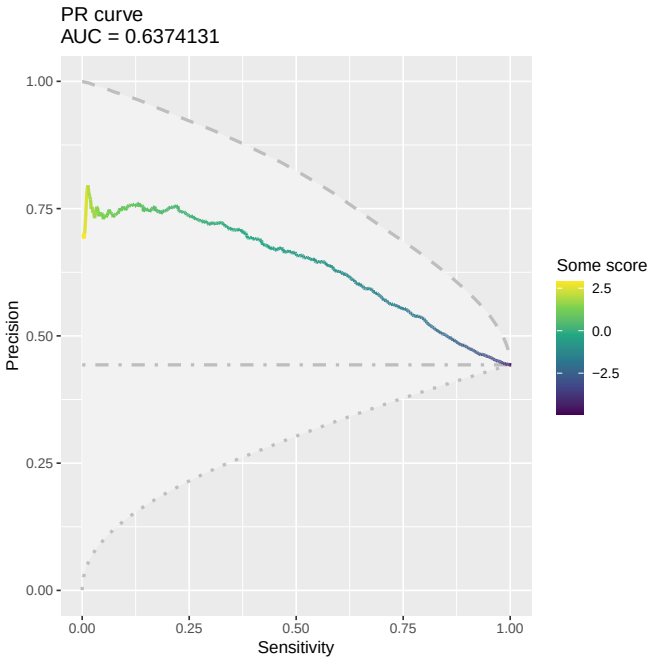
The function `ggprroc` provides basic functionality to plot ROC and PR curves within the ggplot2 ecosystem. The style of the plot is adapted to roughly match `plot.PRROC`.

```
> pl <- ggprroc( wpr, max.plot = TRUE, min.plot = TRUE,
+                rand.plot = TRUE, fill.area=TRUE )
> pl
```



Alternative themes, colour scales etc. may be added to the plot using `ggplot2` mechanisms.

```
> pl + scale_color_viridis_c(option="D",name="Some score") +
+       xlab("Sensitivity")+
+       theme_gray()
```



5 Plotting curves using other plot packages

We can obtain the points of the ROC or PR curve, e.g., for plotting using other packages:

```
> curve.points<-wpr$curve
```

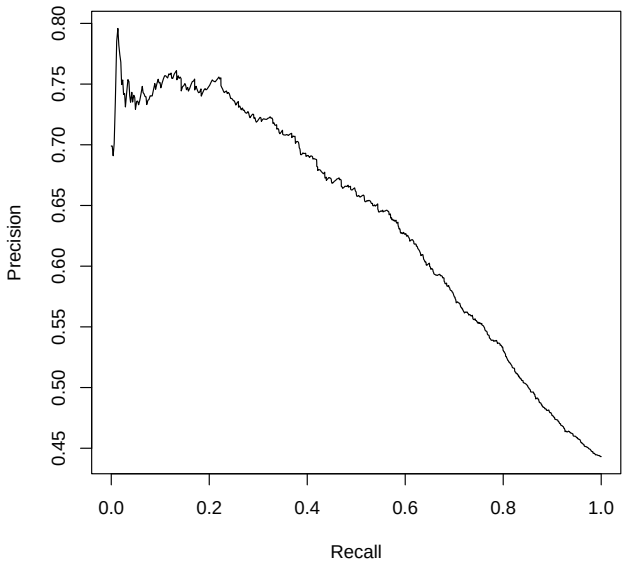
The resulting matrix contains three columns. In case of an ROC curve, the first column contains the false positive rates, the second column contains the corresponding sensitivities, and the third column contains the corresponding classification thresholds. In case of a PR curve, the first column contains the recalls (sensitivities), the second column contains the corresponding precisions, and the third column contains the corresponding classification thresholds, for instance

```
> curve.points[1:5,]
```

```
          [,1]      [,2]      [,3]
[1,] 1.0000000 0.4431009 -4.935784
[2,] 1.0000000 0.4431009 -4.935784
[3,] 0.9987774 0.4431131 -4.766034
[4,] 0.9982695 0.4434427 -4.693354
[5,] 0.9970938 0.4434762 -4.625728
```

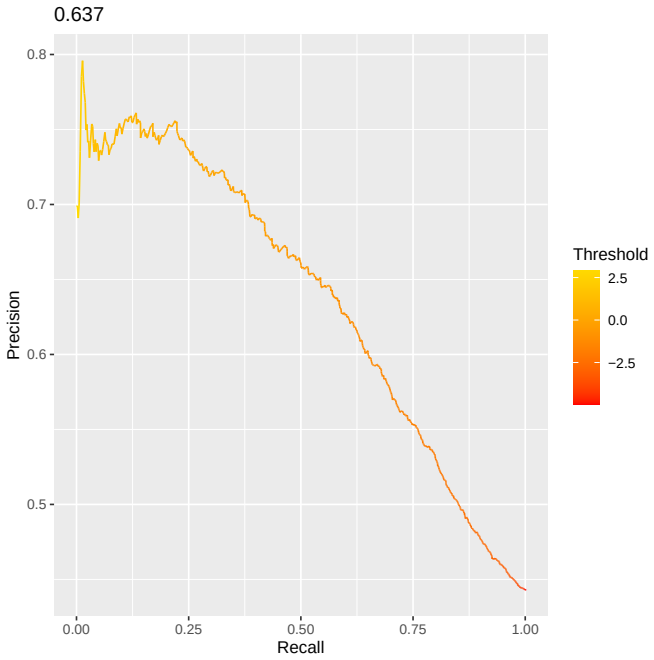
Using this matrix, ROC and PR curves can be plotted using independently of `plot.PRROC`. For instance, we can plot the PR curve from the last section using the standard `plot` command

```
> plot(curve.points[,1],curve.points[,2],
+          xlab="Recall",ylab="Precision",t="l")
```



or using more sophisticated plotting routines, e.g., from `ggplot2` [5]:

```
> (
+ ggplot(data.frame(wpr$curve),aes(x=X1,y=X2,color=X3))
+         + geom_line()
+         + labs(x="Recall",y="Precision",
+                   title=format(wpr$auc.integral,digits=3),
+                   colour="Threshold")
+         + scale_colour_gradient2(low="red", mid="orange",high="yellow")
+ )
```



References

[1] B. Hanczar, J. Hua, C. Sima, J. Weinstein, M. L. Bittner, and E. R. Dougherty. Small- Sample Precision of ROC-related Estimates. *Bioinformatics*, 26(6), 822-830, 2010.

[2] J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 233–240, New York, NY, USA, 2006. ACM.

[3] K. Boyd, K. Eng, and C. Page. Area under the precision-recall curve: Point estimates and confidence intervals. In H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8190 of *Lecture Notes in Computer Science*, pages 451–466. Springer Berlin Heidelberg, 2013.

[4] J. Keilwagen, I. Grosse, and J. Grau. Area under precision-recall curves for weighted and unweighted data. *PLoS ONE*, 9(3):e92209, 03 2014.

[5] H. Wickham. *ggplot2: elegant graphics for data analysis*. Springer New York, 2009.