platt - Robust Platt scaling of prediction values

Günter Klambauer, Andreas Mayr, and Sepp Hochreiter

Institute of Bioinformatics, Johannes Kepler University Linz
Altenberger Str. 69, 4040 Linz, Austria
klambauer@bioinf.jku.at

Version 0.99.4, December 27, 2016

Contents

1 Introduction

Platt scaling (Platt, 1999) maps the outputs of machine learning methods to probabilistic outputs using a sigmoid function:

$$P(y = 1|\hat{y}_k) = \frac{1}{1 + \exp(A\,\hat{y}_k + B)},$$

where $\hat{y}_k$ is the prediction of a machine learning method for data point x_k. A and B are parameters of the sigmoid. This sigmoid model is equivalent to assuming that the output of the machine learning method is proportional to the log odds of a positive example.

The parameters A and B are fit by maximum-likelihood-estimation using a training set with labelled data (x_k, y_k), in which the classes must be coded as 0 and 1. The objective is:

$$\min_{A,B} - \sum_{k=1}^{N} y_k \log\left(\frac{1}{1 + \exp(A\,\hat{y}_k + B)}\right) + (1 - y_k)\log\left(1 - \left(\frac{1}{1 + \exp(A\,\hat{y}_k + B)}\right)\right)$$

By optimizing this objective we obtain parameters A and B for the sigmoid, which we use for transforming the outputs of the machine learning methods into probabilistic outputs. A fast and robust optimization algorithm is implemented in this package.

2 Getting started

To load the package, enter the following in your R session:

```
> library(platt)
```

We have provided an example data set called "MMP" (mitochondrial membrane potential). This data sets includes the cross-validation predictions of neural networks (column "NN"), support vector machines (column "SVM") and random forests (column "RF"). We can see the three columns containing the cross-validation predictions and a fourth column containing the labels:

```
> library(platt)
> data(MMP)
> head(MMP)
```

```
            NN       SVM    RF target
1 4.697468e-03 -0.902294 0.074      0
2 4.697468e-03 -0.902294 0.036      0
3 1.801957e-05 -0.902294 0.070      0
4 2.241387e-06 -0.902294 0.042      0
5 4.964420e-06 -0.902294 0.060      0
6 1.054861e-05 -0.902294 0.000      0
```

We can estimate the sigmoid on the cross-validation predictions by using the following:

```
> plattScalingResult <- plattScaling(MMP$SVM,MMP$target)
```

The object `plattScalingResult` contains the mapped values and the two parameters of the sigmoid A and B.

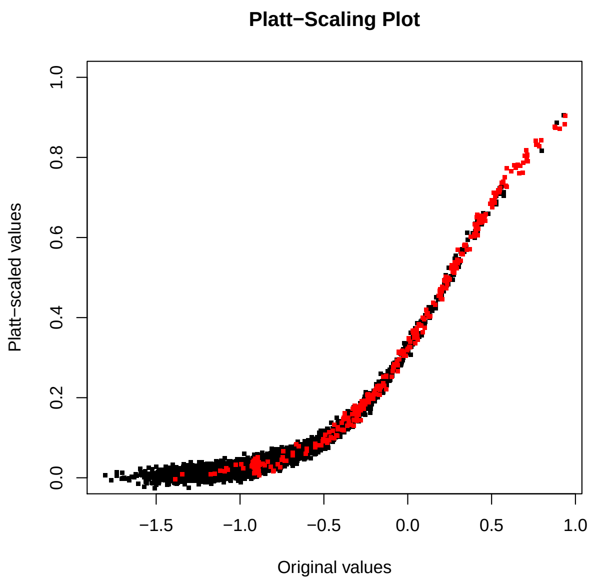
```
> str(plattScalingResult)
```

```
List of 5
 $ pred   : num [1:4726] 0.0302 0.0302 0.0302 0.0302 0.0302 ...
 $ A      : num -1.26
 $ B      : num 3.47
 $ norm   : num [1:2] -0.902 0.414
 $ success: logi TRUE
 - attr(*, "class")= chr "plattScalingResult"
```

3 Plotting the sigmoid function

We are now showing the original values against the Platt-scaled values in the follwing plot:

```
> x1 <- MMP$SVM[MMP$target==0]
> y1 <- plattScalingResult$pred[MMP$target==0]
> plot(x1+rnorm(mean=0,sd=0.01,n=length(x1)),
+                 y1+rnorm(mean=0,sd=0.01,n=length(x1)),
+                 main="Platt-Scaling Plot",ylim=c(0,1),pch=".",cex=4,
+                 xlab="Original values", ylab="Platt-scaled values")
> x2 <- MMP$SVM[MMP$target==1]
> y2 <- plattScalingResult$pred[MMP$target==1]
> points(x2+rnorm(mean=0,sd=0.01,n=length(x2)),
+                 y2+rnorm(mean=0,sd=0.01,n=length(x2)),
+                 pch=".",col="red",cex=4)
```



4 Mapping new values to the sigmoid

If we are given a vector of new values, e.g. from a test data set, from the prediction method, we can readily map them to the sigmoid using the function `predictProb`:

```
> newValues <- c(-1.22,0.51,-0.43, 1.1,-1.01)
> newValuesPlattScaled <- predictProb(plattScalingResult,newValues)
> (cbind(newValues,newValuesPlattScaled))
```

```
     newValues newValuesPlattScaled
[1,]     -1.22           0.01173739
[2,]      0.51           0.69263352
[3,]     -0.43           0.11529439
[4,]      1.10           0.93095369
[5,]     -1.01           0.02195810
```

5 A new probabilistic way to make ensemble predictions

We now have multiple predictions of several machine learning methods that we aim to combine to a single probability that the molecule is active. The probability that a molecule is active given the probabilistic output p_i of a model i is $p(y = 1|p_i)$ and we have n models.

If we have predictions $p_1, \dots, p_n$ of n models, we want to calculate $p(z = 1 \mid p_1, \dots, p_n)$ using $p(y = 1|p_i)$. This can be achieved by using the following formula:

$$\begin{aligned}
&\frac{\prod_{i=1}^{n} p(z=1 \mid p_i)}{\prod_{i=1}^{n} p(z=1 \mid p_i) \;+\; \prod_{i=1}^{n} p(z=0 \mid p_1) \; \left(\frac{p(z=1)}{p(z=0)}\right)^{n-1}} \;=\\
&\frac{\prod_{i=1}^{n} p(z=1 \mid p_i)\, p(p_i)}{\prod_{i=1}^{n} p(z=1 \mid p_i)\, p(p_i) \;+\; \prod_{i=1}^{n} p(z=0 \mid p_i)\, p(p_i) \; \left(\frac{p(z=1)}{p(z=0)}\right)^{n-1}} \;=\\
&\frac{\prod_{i=1}^{n} p(z=1, p_i)}{\prod_{i=1}^{n} p(z=1, p_i) \;+\; \prod_{i=1}^{n} p(z=0, p_i) \; \left(\frac{p(z=1)}{p(z=0)}\right)^{n-1}} \;=\\
&\frac{\prod_{i=1}^{n} p(p_i \mid z=1)\, p^n(z=1)}{\prod_{i=1}^{n} p(p_i \mid z=1)\, p^n(z=1) \;+\; \prod_{i=1}^{n} p(p_i \mid z=0)\, p(z=0)\, p^{n-1}(z=1)} \;=\\
&\frac{p(p_1,\ldots,p_n \mid z=1)\, p(z=1)}{p(p_1,\ldots,p_n \mid z=1)\, p(z=1) \;+\; p(p_1,\ldots,p_n \mid z=0)\, p(z=0)} \;=\\
&\frac{p(p_1,\ldots,p_n, z=1)}{p(p_1,\ldots,p_n, z=1) \;+\; p(p_1,\ldots,p_n, z=0)} \;=\\
&\frac{p(p_1,\ldots,p_n, z=1)}{p(p_1,\ldots,p_n)} \;=\\
&p(z=1 \mid p_1,\ldots,p_n)\;.
\end{aligned} \tag{1}$$

In the formula above the expressions $p(z = 1)$ and $p(z = 0)$ are the prior probabilities that a compound is active or inactive, respectively. These values can be estimated from the relative frequencies of actives and inactives on the training set.

We have assumed above that:

$$\prod_{i=1}^{n} p(p_i \mid z=1) \;=\; p(p_1,\ldots,p_n \mid z=1) \tag{2}$$

$$\prod_{i=1}^{n} p(p_i \mid z=0) \;=\; p(p_1,\ldots,p_n \mid z=0)\;. \tag{3}$$

Another formula with equivalent order. The following formula can also be used:

$$\frac{\prod_{i=1}^{n} p(z=1 \mid p_i)}{\prod_{i=1}^{n} p(z=1 \mid p_i) \;+\; \prod_{i=1}^{n} p(z=0 \mid p_1)} \tag{4}$$

This formula is equivalent with respect to the order, as can be seen in the following:

$$\begin{aligned}
&\frac{a_1}{a_1 \;+\; b_1\, c} \;>\; \frac{a_2}{a_2 \;+\; b_2\, c}\\
\Leftrightarrow\;& a_1\, a_2 \;+\; a_1\, b_2\, c \;>\; a_1\, a_2 \;+\; a_2\, b_1\, c\\
\Leftrightarrow\;& a_1\, b_2\, c \;>\; a_2\, b_1\, c\\
\Leftrightarrow\;& a_1\, b_2 \;>\; a_2\, b_1\\
\Leftrightarrow\;& a_1\, a_2 \;+\; a_1\, b_2 \;>\; a_1\, a_2 \;+\; a_2\, b_1\\
\Leftrightarrow\;& \frac{a_1}{a_1 \;+\; b_1} \;>\; \frac{a_2}{a_2 \;+\; b_2}\;.
\end{aligned} \tag{5}$$

Implementation. The function `ensemble` implements the above method to combine probabilistic predictions of different machine learning methods to a single prediciton.

```
> p1 <- plattScaling(MMP$NN,MMP$target)$pred
> p2 <- plattScaling(MMP$RF,MMP$target)$pred
> p3 <- plattScaling(MMP$SVM,MMP$target)$pred
> df <- data.frame(p1,p2,p3)
> ensemblePrediction <- ensemble(df,
+                  class1Prob=length(which(MMP$target==1))/nrow(MMP))
> (table(ensemblePrediction>0.5, MMP$target))
```

```
          0    1
  FALSE 4134  113
  TRUE   239  240
```

References

Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in large margin classifiers*.