Beta calibration

Introduction

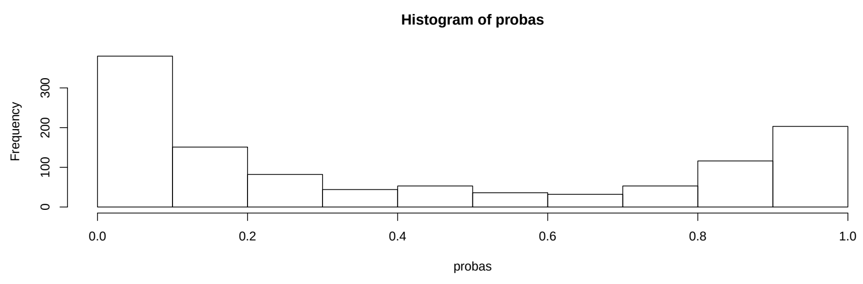
Logistic calibration is designed to correct for a specific kind of distortion where classifiers tend to score on too narrow a scale. However, many classifiers including Naive Bayes and standard Adaboost suffer from the opposite distortion where scores tend too much to the extremes.

In this tutorial, we will motivate Beta calibration and our betacal R package.

Probability estimation with Adaboost

First, let's train an Adaboost model with 200 decision trees to estimate class probabilities for the well-known spam dataset. The dataset will be divided into a training set (50%), a test set (25%) and a calibration set (25%). The classifier will be trained on the training set and we'll estimate class probabilities for the test set.

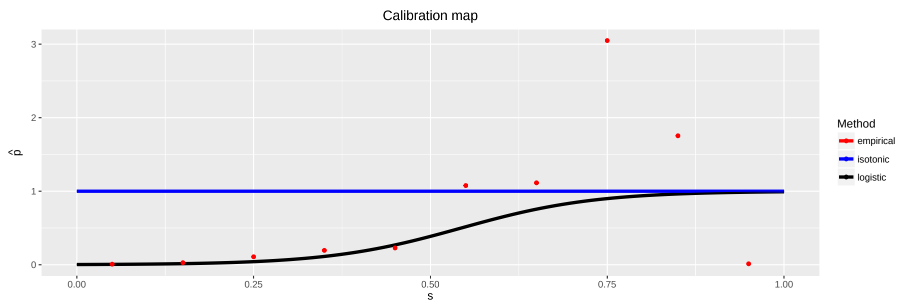
```
library(fastAdaboost)
library(caret)

data <- read.table("spambase.data", sep = ",")
names(data)[58] <- paste("label")

data$label <- as.factor(data$label)

train.index <- createDataPartition(data$label, p=.5, list=FALSE)
train <- data[ train.index,]
test  <- data[-train.index,]

cal.index <- createDataPartition(test$label, p=.5, list=FALSE)
cal <- test[ cal.index,]
test  <- test[-cal.index,]

ada <- adaboost(label~., train, 50)

probas <- (predict(ada, newdata=test, type="response"))
probas <- probas$prob[, 2]

hist(probas)
```



Calibrating the scores

We can clearly see from the histogram that the probabilities produced by the model tend to assume extreme values. Therefore, it might be useful to apply calibration techniques to try and fix these distortions. Two calibration methods have been widely used in machine learning literature: logistic calibration and isotonic regression. The first one is a parametric method that assumes an underlying distribution of the scores composed of two Gaussians of equal variance, (one for the positives and another for the negatives). The second method is a non-parametric approach, therefore it doesn't make any assumption about the distribution of the scores, however, it needs lots of data to produce a good model. Let's see the effect of applying these methods to the previously trained classifier's outputs.

```r
source("calmap.r")
source("fit.isoreg.r")

cal_probas <- (predict(ada, newdata=cal, type="response"))
cal_probas <- cal_probas$prob[, 2]
d <- data.frame("p"=cal_probas, "label"=cal$label)

lr <- glm(label~.,family=binomial(link='logit'), data=d)

idx <- duplicated(cal_probas)
Y.calib.pred.unique <- cal_probas[!idx]
Y.calib.unique <- cal$label[!idx]

iso <- isoreg(Y.calib.pred.unique, Y.calib.unique)

linspace <- seq(0, 1, length.out=100)
d <- data.frame("p"=linspace)
logistic <- (predict(lr, newdata=d, type="response"))
isotonic <- fit.isoreg(iso, linspace)
s_set <- data.frame(linspace, logistic, isotonic)
info <- data.frame("prob"=cal_probas, "labels"=cal$label)
l_set <- c("logistic", "isotonic")
c_set <- c("red", "blue")
plot_calibration_map(s_set, info, l_set, c_set, alpha=0)
```


Calibration map
p̂
s
Method
empirical
isotonic
logistic