Beta calibration

Introduction

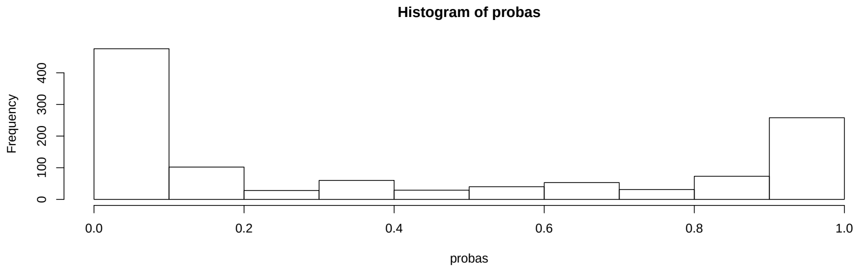
Logistic calibration is designed to correct for a specific kind of distortion where classifiers tend to score on too narrow a scale. However, many classifiers including Naive Bayes and standard Adaboost suffer from the opposite distortion where scores tend too much to the extremes.

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.

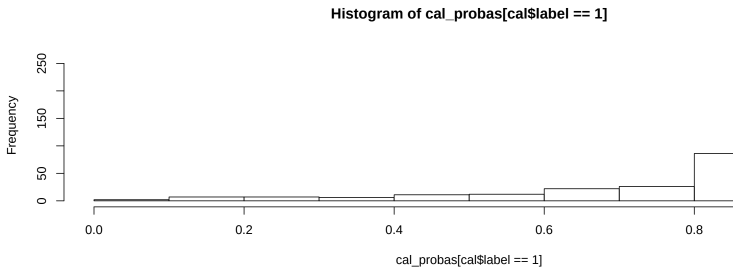
```
library(fastAdaboost)
library(caret)

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, 5)

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



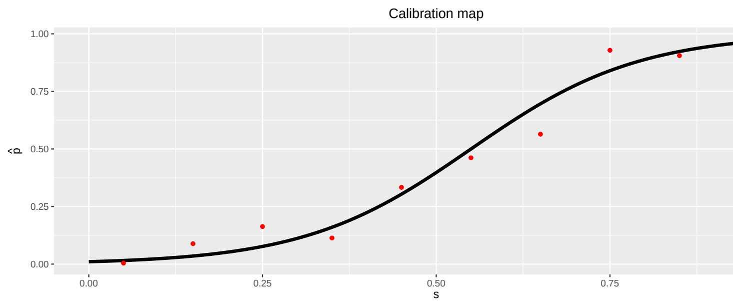
Calibrating the scores

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 calibrati
calibration methods have been widely used in machine
regression. The first one is a parametric method that assu
of two Gaussians of equal variance, (one for the positives
is a non-parametric approach, therefore it doesn't make
however, it needs lots of data to produce a good model.
previously trained classifier's outputs.

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

cal_probas <- (predict(ada, newdata=cal, type="respon
cal_probas <- cal_probas$prob[, 2]

hist(cal_probas[cal$label==1])
```



```
d <- data.frame("p"=cal_probas, "label"=cal$label)

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

#Eliminating duplicated scores to train the isotonic
# 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$la
l_set <- c("logistic")#, "isotonic")
c_set <- c("red")#, "blue")
plot_calibration_map(s_set, info, l_set, c_set, alpha
```



Calibration map

In the calibration map shown above, we can see that:

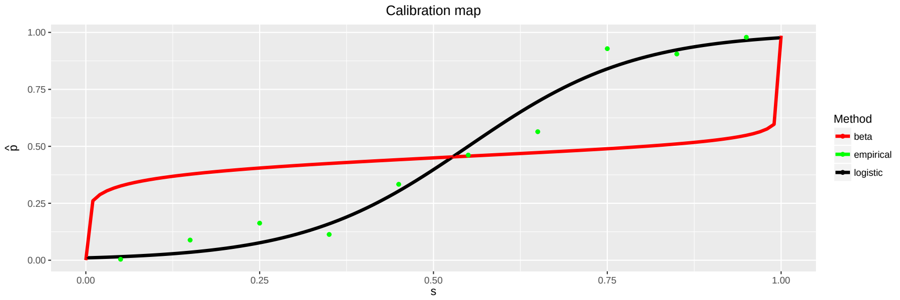
- Logistic calibration fit the scores in the first and last bins well, but was either underconfident or overconfident in the rest of the map, due to the limitations of its shape
- As a non-parametric method, Isotonic regression managed to fit the scores better than Logistic calibration by finding an inverted sigmoid shape

Fitting a beta calibration model with the betacal package

Our beta calibration is a parametric method which is able to circumvent logistic calibration's shape limitations by fitting three parameters, i.e. two for shape (a and b) and one for location (m). These parameters can be fitted by training a logistic regression model with two features extracted from the classifier's probability outputs. Beta calibration is adequate for calibrating probabilities, because the beta distribution has support [0, 1], while logistic calibration assumes the scores are distributed in two gaussians, which have infinite support.

To make it easy for practitioners to fit a beta calibration model, we have provided a Python package called betacal. The package can be installed via pip (pip install betacal). Below, we show how to use the package to fit the three-parameter version of beta calibration. For the other two versions, the practitioner can set parameters="ab" (fit shape parameters a and b and fix location parameter m = 0.5) or parameters="am" (fit shape parameter a, setting a = b, and fit location parameter m).

```r
library("betacal")

bc <- beta_calibration(cal_probas, cal$label, parameters = "abm")

beta <- beta_predict(linspace, bc)
s_set <- data.frame(linspace, logistic, beta)#, isotonic)
l_set <- c("logistic", "beta")#, "isotonic")
c_set <- c("red", "green")#, "blue")
plot_calibration_map(s_set, info, l_set, c_set, alpha=0)
```

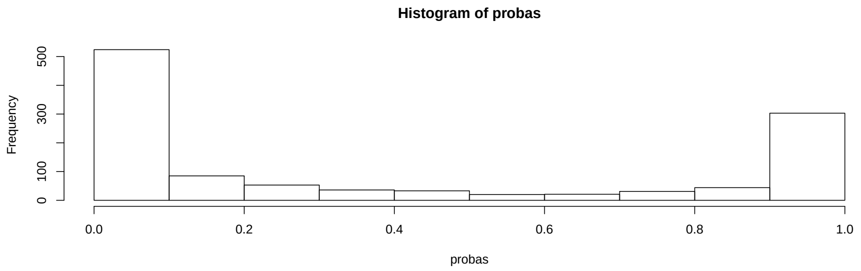


Beta calibration was able to find the inverted sigmoid shape, as isotonic regression did, resulting in a much better fit for the scores than logistic calibration. Now let's see what happens when a classifier outputs probabilities that tend to be concentrated around mean values.

Probability estimation with logistic regression

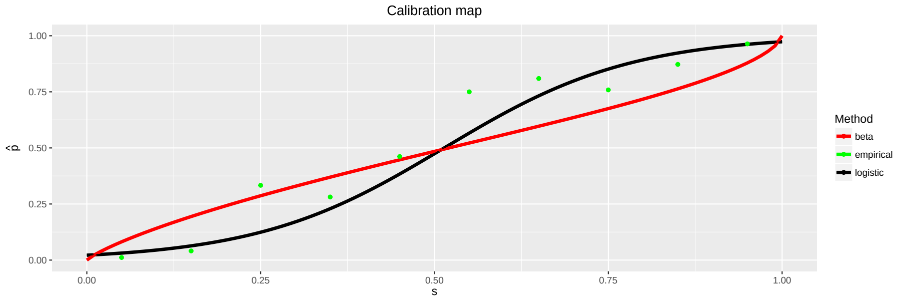
Below, we train a logistic regression using the same spam dataset. We can see that its probability outputs are concentrated around 0.5. This distortion is what logistic calibration works on best.

```
lrc <- glm(label˜.,family=binomial(link='logit'),data=train)

probas <- (predict(lrc, newdata=test, type="response"))

hist(probas)
```



Now let's train the calibration models.

```
cal_probas <- (predict(lrc, newdata=cal, type="response"))
d <- data.frame("p"=cal_probas, "label"=cal$label)
lr <- glm(label˜.,family=binomial(link='logit'), data=d)
linspace <- seq(0, 1, length.out=100)
d <- data.frame("p"=linspace)
logistic <- (predict(lr, newdata=d, type="response"))
bc <- beta_calibration(cal_probas, cal$label, parameters = "abm")
beta <- beta_predict(linspace, bc)
s_set <- data.frame(linspace, logistic, beta)#, isotonic)
info <- data.frame("prob"=cal_probas, "labels"=cal$label)
l_set <- c("logistic", "beta")#, "isotonic")
c_set <- c("red", "green")#, "blue")
plot_calibration_map(s_set, info, l_set, c_set, alpha=0)
```



Notice how all three methods find sigmoid shapes. This is because the sigmoid shape is part of the beta calibration family, which also contains an identity calibration map, which just keeps the scores with their original values (this is not possible with logistic calibration). In both maps, isotonic regression did a great job fitting the scores, but as a non-parametric method, it needs a lot of data to train (as is the case with the spam dataset), which is not a problem for beta calibration. Therefore, given that beta calibration is able to fit the classifier's output probabilities better than or at least equal to logistic calibration, we argue that a practicioner who wants to use a parametric calibration method should choose beta calibration.