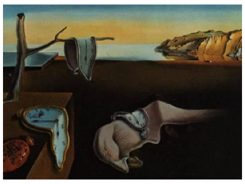

Journal of Statistical Software

June 2017, Volume 78, Issue 4. *doi: 10.18637/jss.v078.i04*

pvclass: An R Package for p Values for Classification

Niki Zumbrunnen
University of Bern

Lutz Dümbgen
University of Bern

Abstract

Let (X, Y) be a random variable consisting of an observed feature vector X and an unobserved class label $Y \in \{1, 2, \ldots, L\}$ with unknown joint distribution. In addition, let $\mathcal{D}$ be a training data set consisting of n completely observed independent copies of (X, Y). Instead of providing point predictors (classifiers) for Y, we compute for each $b \in \{1, 2, \ldots, L\}$ a p value $\pi_b(X, \mathcal{D})$ for the null hypothesis that $Y = b$, treating Y temporarily as a fixed parameter, i.e., we construct a prediction region for Y with a certain confidence. The advantages of this approach over more traditional ones are reviewed briefly. In principle, any reasonable classifier can be modified to yield nonparametric p values.

We describe the R package **pvclass** which computes nonparametric p values for the potential class memberships of new observations as well as cross-validated p values for the training data. Additionally, it provides graphical displays and quantitative analyses of the p values.

Keywords: classification, quantification of uncertainty, R.

1. Introduction

Let (X, Y) be a pair of random variables, consisting of an observed feature vector X with values in a feature space $\mathcal{X}$ and an unobserved class label $Y \in \mathcal{Y} := \{1, 2, \ldots, L\}$ with $L \geq 2$ possible values. Our aim is inference about Y with a given confidence, based on X and certain training data. In the sequel we provide a brief introduction to the particular paradigm of p values as introduced by Dümbgen, Igl, and Munk (2008). It is closely related to Neyman-Pearson classification, see Scott (2007), Zhao, Feng, Wang, and Tong (2015) and the references cited therein.

1.1. From classifiers to p values

For the moment let us assume that the joint distribution of X and Y is known, i.e., the a priori probabilities $w_y := \mathsf{P}(Y = y)$ and the conditional distributions $P_y := \mathcal{L}(X \,|\, Y = y)$ for all $y \in \mathcal{Y}$. Here and throughout, $\mathcal{L}(\cdot)$ stands for 'distribution of'. Further let P_y have density $f_y > 0$ with respect to some measure M on $\mathcal{X}$. In the simplest case, we choose a classifier $\hat{Y} \colon \mathcal{X} \to \mathcal{Y}$, i.e., a point predictor of Y, and claim that $Y = \hat{Y}(X)$. However we have no information about confidence. A Bayesian approach would be to calculate the posterior distribution of Y given X, i.e., the posterior weights

$$w_y(X) := \mathsf{P}(Y = y \,|\, X) = w_y f_y(X)/f(X)$$

with the density $f = \sum_{y \in \mathcal{Y}} w_y f_y$ of $\mathcal{L}(X)$ with respect to M. In fact, a classifier $\hat{Y}^*$ satisfying $\hat{Y}^*(X) \in \arg\max_{y \in \mathcal{Y}} w_y(X)$ is optimal in the sense of minimizing the risk $R(\hat{Y}) := \mathsf{P}(\hat{Y}(X) \neq Y)$. Furthermore, the posterior weights $w_y(X)$ carry additional information such as $\mathsf{P}(\hat{Y}^*(X) \neq Y \,|\, X) = 1 - \max_{y \in \mathcal{Y}} w_y(X)$. However, this depends very sensitively on the prior weights w_y. In some realistic settings, e.g., case-control studies in biostatistics, estimation of the conditional distributions P_y from training data is possible, but we have no information about the a priori probabilities w_y. Moreover, if some classes y have very small prior weights, the classifier $\hat{Y}^*$ tends to ignore these, i.e., the class-dependent risk $\mathsf{P}(\hat{Y}^* \neq Y \,|\, Y = y)$ may be rather large for some classes y.

To treat all classes impartially, we propose to treat Y temporarily as an unknown fixed parameter and to provide for each class label $\theta \in \mathcal{Y}$ a p value π_θ for the null hypothesis that $Y = \theta$. That means, $\pi_\theta \colon \mathcal{X} \to [0, 1]$ satisfies

$$\mathsf{P}(\pi_\theta(X) \leq \alpha \,|\, Y = \theta) \leq \alpha \quad \text{for all} \quad \alpha \in (0,1). \tag{1}$$

Given such p values π_θ, the set

$$\hat{\mathcal{Y}}_\alpha(X) := \{\theta \in \mathcal{Y} \colon \pi_\theta(X) > \alpha\}$$

is a $(1-\alpha)$-prediction region for Y, i.e.,

$$\mathsf{P}(Y \in \hat{\mathcal{Y}}_\alpha(X) \,|\, Y = \theta) \geq 1 - \alpha \quad \text{for any} \quad \theta \in \mathcal{Y}, \ \alpha \in (0,1).$$

Thus we can exclude all classes $\theta \notin \hat{\mathcal{Y}}_\alpha(X)$ with confidence $1 - \alpha$. If there is only one $\theta \in \hat{\mathcal{Y}}_\alpha(X)$, we have classified X uniquely with confidence $1 - \alpha$. There exist intrinsically difficult classification problems, especially in biomedical applications. But even in such situations, some observations may be classified uniquely with high confidence. And in settings with $L \geq 3$ potential classes, excluding at least one or more classes with a prescribed confidence might be useful.

1.2. Example

In case of $L = 2$ classes, the posterior weights may be written as

$$w_1(x) = \bigl(1 + (w_2/w_1)(f_2/f_1)(x)\bigr)^{-1} \quad \text{and} \quad w_2(x) = 1 - w_1(x),$$

i.e., $w_2(x)$ is a strictly increasing function of $(f_2/f_1)(x)$. Specifically, let $\mathcal{X} = (0, \infty)$, and suppose that $P_1 = \text{Gamma}(3,1)$, $P_2 = \text{Gamma}(6,1)$. Here $(f_2/f_1)(x) = x^3/24$. Figure 1 illustrates how sensitively the optimal point predictor $Y^*(x)$ and the posterior weights $w_y(x)$

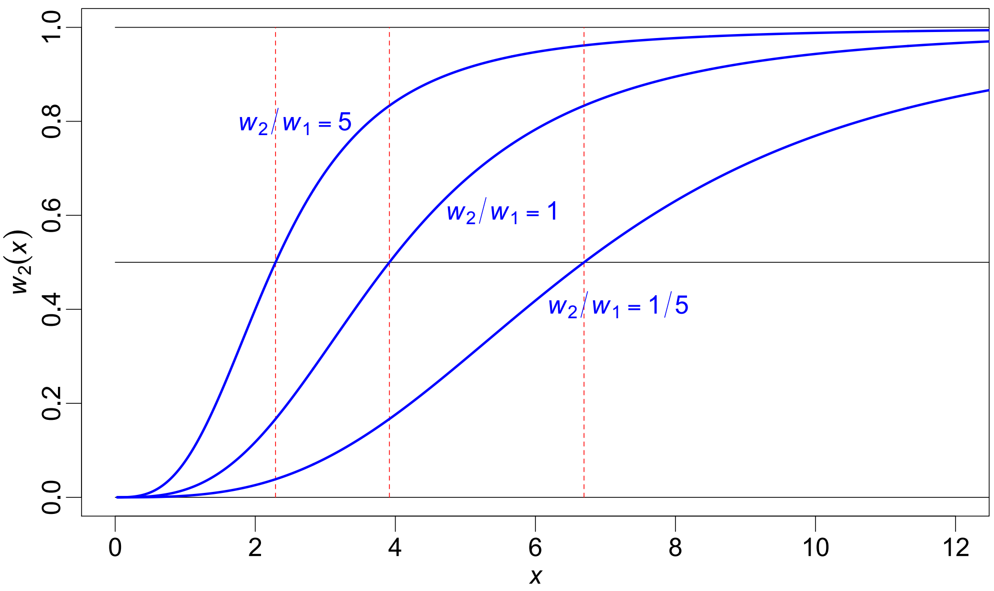

Figure 1: Posterior weights $w_2(x)$ for different ratios of prior probabilities w_2/w_1.

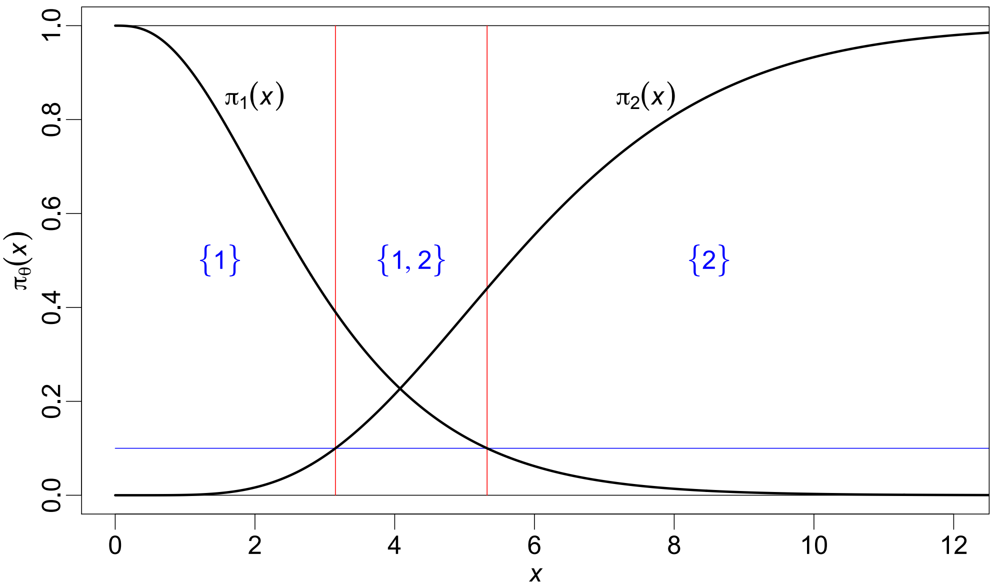

Figure 2: p value functions for class memberships.

depend on the prior ratio w_2/w_1. It shows $w_2(x)$ for $w_2/w_1 = 5, 1, 1/5$ from left to right. The corresponding boundaries of $Y^*(x)$ are drawn as vertical lines.

Alternatively, we can calculate p values which do not depend on the prior probabilities w_y. Since $w_2(x)$ is increasing in x, we define the p values

$$\begin{aligned}\pi_1(x) &:= \mathsf{P}(X \geq x \,|\, Y = 1) = 1 - F_1(x),\\ \pi_2(x) &:= \mathsf{P}(X \leq x \,|\, Y = 2) = F_2(x),\end{aligned}$$

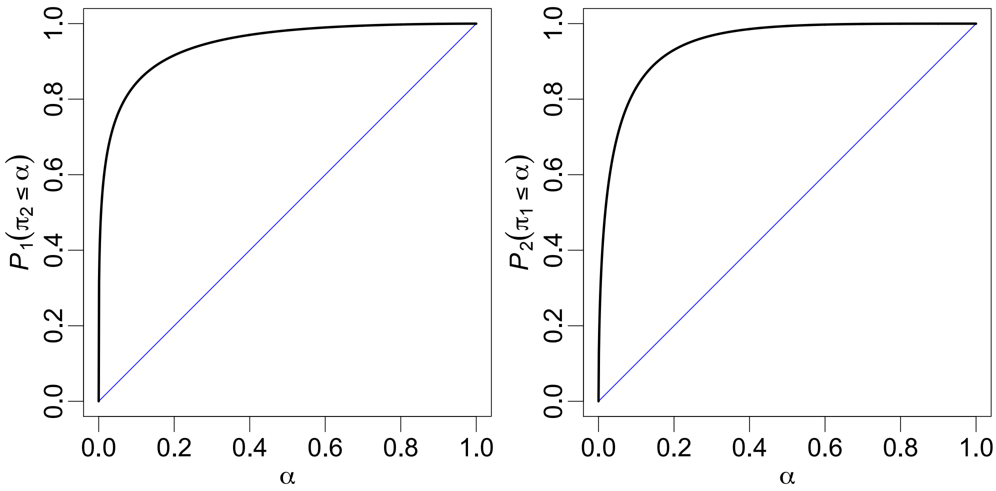

Figure 3: ROC functions $P_1(\pi_2 \leq \cdot)$ (left) and $P_2(\pi_1 \leq \cdot)$ (right).

where F_θ is the distribution function of P_θ. If $\pi_\theta(X) \leq \alpha$ we claim with confidence $1-\alpha$ that $Y \neq \theta$. Figure 2 shows the p value functions $\pi_1(x)$ and $\pi_2(x)$. In addition, the three regions where $\hat{\mathcal{Y}}_{0.1}(x) = \{1\}, \{2\}, \{1,2\}$ are marked. With other choices of the gamma parameters there could be a region where $\hat{\mathcal{Y}}_{0.1}(x) = \emptyset$.

To assess how well the two classes can be discriminated, it is also of interest to look at the two receiver operating characteristic (ROC) functions

$$\begin{aligned}
(0,1) \ni \alpha &\mapsto \mathsf{P}(\pi_2(X) \leq \alpha \,|\, Y = 1) = P_1(\pi_2 \leq \alpha) = F_1(F_2^{-1}(\alpha)), \\
(0,1) \ni \alpha &\mapsto \mathsf{P}(\pi_1(X) \leq \alpha \,|\, Y = 2) = P_2(\pi_1 \leq \alpha) = 1 - F_2(F_1^{-1}(1-\alpha)).
\end{aligned}$$

These functions are depicted in Figure 3. The first ROC function specifies for each test level α the probability that class 2 is rejected at this level if class 1 is the true one, i.e. the conditional probability that $2 \notin \hat{\mathcal{Y}}_\alpha(X)$ given $Y = 1$. The second ROC curve is analogous with the roles of classes 1 and 2 interchanged. For a general account of ROC curves in binary classification and hypothesis testing we refer to Altman and Bland (1994) and Fawcett (2006).

1.3. Optimal p values as benchmark

Suppose that $\mathcal{L}((f_\theta/f)(X))$ is continuous. This is true, for instance, if the P_y are non-degenerate Gaussian distributions on $\mathcal{X} = \mathbb{R}^q$ and not all identical. Then the Neyman-Pearson Lemma shows that the p value

$$\pi^*_\theta(x) = P_\theta\{z \in \mathcal{X} \colon (f_\theta/f)(z) \leq (f_\theta/f)(x)\} \tag{2}$$

is optimal in the sense that it minimizes the risk $\mathcal{R}_\alpha(\pi_\theta) := \mathsf{P}(\pi_\theta > \alpha)$ for any $\alpha \in (0,1)$, cf. Dümbgen *et al.* (2008). Two other representations of $\pi^*_\theta(x)$ are given by

$$\begin{aligned}
\pi^*_\theta(x) &= P_\theta\{z \in \mathcal{X} \colon w_\theta(z) \leq w_\theta(x)\} \\
&= P_\theta\{z \in \mathcal{X} \colon T^*_\theta(z) \geq T^*_\theta(x)\}
\end{aligned}$$

with

$$T^*_\theta(x) := \sum_{b \neq \theta} \frac{w_{b,\theta} f_b(x)}{f_\theta(x)} \quad \text{and} \quad w_{b,\theta} := \Big(\sum_{c \neq \theta} w_c / w_b \Big)^{-1}.$$

The first representation shows that $\pi^*_\theta(x)$ is a non-decreasing function of $w_\theta(x)$. The second representation shows that the prior weight w_θ itself is irrelevant for the optimal p value π^*_θ. Only the ratios w_c/w_b with $b, c \neq \theta$ matter. In particular, in case of $L = 2$ classes, $T^*_1(x) = (f_2/f_1)(x) = T^*_2(x)^{-1}$, and the optimal p values do not depend on the prior distribution of Y at all.

1.4. Training data and nonparametric p values

The joint distribution of (X, Y) is typically unknown. In this case we compute p values $\pi_\theta(X, \mathcal{D})$ and prediction regions

$$\hat{\mathcal{Y}}_\alpha(X, \mathcal{D}) := \{\theta \in \mathcal{Y} \colon \pi_\theta(X, \mathcal{D}) > \alpha\}$$

depending on the current feature vector X as well as on a training data set $\mathcal{D}$ consisting of n completely observed pairs $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$. We assume that (X, Y) and $(X_1, Y_1), \ldots, (X_n, Y_n)$ are independent with $\mathcal{L}(X_i \,|\, Y_i = y) = P_y$. This setting includes situations with stratified training data, e.g., case-control studies, as well as i.i.d. training data. Condition (1) can be extended in two ways:

$$\begin{aligned} \mathsf{P}(\pi_\theta(X, \mathcal{D}) \leq \alpha \,|\, Y = \theta) &\leq \alpha, & (3) \\ \mathsf{P}(\pi_\theta(X, \mathcal{D}) \leq \alpha \,|\, Y = \theta, \mathcal{D}) &\leq \alpha + o_p(1) \quad \text{as } n \to \infty, & (4) \end{aligned}$$

for any $\theta \in \mathcal{Y}, \alpha \in (0, 1)$. Condition (3) corresponds to "single use" of the p values in the sense that we consider how well *one* future observation (X, Y) is classified. This condition (3) can be guaranteed in various settings. Condition (4) corresponds to "multiple use" of the p values in the sense that the training data $\mathcal{D}$ are utilized to classify arbitrarily many future observations. Thus it is reasonable to condition on the training data $\mathcal{D}$ as well as the class label of future observations. As explained in Dümbgen *et al.* (2008), Condition (4) can be guaranteed under moderate assumptions, but the summand $o_p(1)$ cannot be avoided in general.

For the computation of the p values we condition on the $n + 1$ class labels $Y_1, Y_2, \ldots, Y_n$ and Y and treat them as fixed parameters. We write

$$N_\theta := \#\mathcal{G}_\theta \quad \text{and} \quad \mathcal{G}_\theta := \{i \in \{1, \ldots, n\} \colon Y_i = \theta\}$$

and assume tacitly that all these group sizes N_θ are positive.

To compute a nonparametric p value $\pi_\theta(X, \mathcal{D})$ for one particular class label θ, let $T_\theta(X, \mathcal{D})$ be a test statistic which is symmetric in $(X_i)_{i \in \mathcal{G}_\theta}$ and quantifies the implausibility of '$Y = \theta$'. To test the latter hypothesis we define $\mathcal{D}(X, \theta)$ to be the training data augmented by (X, θ), i.e., we assume temporarily that $Y = \theta$. If that is true, the $N_\theta + 1$ random variables X and X_i, $i \in \mathcal{G}_\theta$, are exchangeable. Hence the nonparametric p value

$$\pi_\theta(X, \mathcal{D}) := \frac{\#\{i \in \mathcal{G}_\theta \colon T_\theta(X_i, \mathcal{D}(X, \theta)) \geq T_\theta(X, \mathcal{D}(X, \theta))\} + 1}{N_\theta + 1} \tag{5}$$

does satisfy (3).

By definition, $\pi_\theta(X, \mathcal{D}) \geq (N_\theta + 1)^{-1}$. Therefore this procedure is only useful, if $N_\theta + 1 \geq \alpha^{-1}$. For instance if $\alpha = 0.05$, N_θ should be at least 19.

As to the test statistic $T_\theta(X, \mathcal{D})$, there are no restrictions except for the symmetry in $(X_i)_{i \in \mathcal{G}_\theta}$. The optimal p value $\pi^*_\theta(x)$ suggests using an estimator for the weighted likelihood ratio $T^*_\theta(x)$ or a strictly increasing transformation thereof. Section 2 contains explicit examples for $T_\theta(X, \mathcal{D})$.

More details on the nonparametric p values, including asymptotic properties such as (4), can be found in Dümbgen *et al.* (2008) and Zumbrunnen (2009).

1.5. Cross-validated p values and ROC functions

To visualize the separability of different classes by means of given p values $\pi_\theta(\cdot, \cdot)$, we compute cross-validated p values

$$\pi_\theta(X_i, \mathcal{D}_i)$$

for $i = 1, \ldots, n$ with $\mathcal{D}_i$ denoting the training data without observation (X_i, Y_i). That means, we treat each training observation temporarily as a new observation to be classified with the remaining data $\mathcal{D}_i$. Then we display the cross-validated p values graphically.

We also compute the empirical conditional inclusion probabilities

$$\hat{\mathcal{I}}_\alpha(b, \theta) := \frac{\#\{i \in \mathcal{G}_b \colon \theta \in \hat{\mathcal{Y}}_\alpha(X_i, \mathcal{D}_i)\}}{N_b}$$

and the empirical pattern probabilities

$$\hat{\mathcal{P}}_\alpha(b, S) := \frac{\#\{i \in \mathcal{G}_b \colon \theta \in \hat{\mathcal{Y}}_\alpha(X_i, \mathcal{D}_i) = S\}}{N_b}$$

for $b, \theta \in \mathcal{Y}$ and $S \subset \mathcal{Y}$. These numbers can be interpreted as estimators of

$$\mathcal{I}_\alpha(b, \theta \,|\, \mathcal{D}) := \mathsf{P}(\theta \in \hat{\mathcal{Y}}_\alpha(X, \mathcal{D}) \,|\, Y = b, \mathcal{D})$$

and

$$\mathcal{P}_\alpha(b, S \,|\, \mathcal{D}) := \mathsf{P}(\hat{\mathcal{Y}}_\alpha(X, \mathcal{D}) = S \,|\, Y = b, \mathcal{D}),$$

respectively.

Finally, for training data with large group sizes N_b we also display the L^2 empirical ROC functions as a plot matrix with L rows and L columns. In row b and column θ we depict the function

$$(0, 1) \ni \alpha \mapsto 1 - \hat{\mathcal{I}}_\alpha(b, \theta).$$

For any fixed α (horizontal axis), the value $1 - \hat{\mathcal{I}}_\alpha(b, \theta)$ (vertical axis) is an estimator for the probability that class θ is rejected at this level if class b is the true one, i.e. the conditional probability that $\theta \not\in \hat{\mathcal{Y}}_\alpha(X, \mathcal{D})$ given $Y = 1$ and $\mathcal{D}$. The plots on the diagonal ($b = \theta$) are essentially straight lines connecting $(0, 0)$ and $(1, 1)$. Deviations are due to the fact that $\pi_\theta(\cdot, \mathcal{D})$ has a discrete distribution.

1.6. Data example `buerk`

To illustrate the main functions of **pvclass** we use the data set `buerk` provided by **pvclass**. It was collected by Prof. Dr. Conny Georg Bürk at the university hospital in Lübeck, Germany, and contains data of 21556 surgeries in a certain time period (end of the nineties). Besides the mortality and the morbidity it contains 21 variables describing the condition of the patient and the surgery.

2. Choices of test statistics

In this section we describe the test statistics $T_\theta(\cdot,\cdot)$ implemented in the package **pvclass**. As indicated in Section 5, users could easily incorporate test statistics corresponding to their own favorite classifiers, see Section 5. From now on we assume that the feature vector X consists of q numerical covariates. If the raw data involve categorical coveralls, they should be coded by means of $\{0,1\}$-valued indicator variables as usual. In our procedures the latter are treated like numerical variables.

2.1. Plug-in estimator for standard model

In the standard model, where $P_\theta = \mathcal{N}_q(\mu_\theta, \Sigma)$ with mean vectors $\mu_\theta \in \mathbb{R}^q$ and a common symmetric, nonsingular covariance matrix $\Sigma \in \mathbb{R}^{q\times q}$, the test statistic for the optimal p value is given by

$$T^*_\theta(x) = \sum_{b\neq\theta} w_{b,\theta} \exp\left((x-\mu_{\theta,b})^\top \Sigma^{-1}(\mu_b - \mu_\theta)\right)$$

with $\mu_{\theta,b} := 2^{-1}(\mu_\theta + \mu_b)$. To compute the nonparametric p values, we replace the unknown parameters in T^*_θ with the corresponding estimators. We use N_b/n as a proxy for w_b and the the standard estimator

$$\hat\mu_\theta := \frac{1}{N_\theta}\sum_{i\in\mathcal{G}_\theta} X_i$$

for μ_θ. For Σ the package **pvclass** offers the standard estimator

$$\hat\Sigma := \frac{1}{n-L}\sum_{\theta\in\mathcal{Y}}\sum_{i\in\mathcal{G}_\theta}(X_i - \hat\mu_\theta)(X_i - \hat\mu_\theta)^\top.$$

as well as more robust M estimators $\hat\Sigma_M$ and $\hat\Sigma_{\text{sym}}$. The reason to use a robust estimator for Σ is that for the computation of $\pi_\theta(X,\mathcal{D})$ we add the new observation X temporarily to the the class θ and X may be an outlier with respect to the distribution P_θ.

The first M estimator, $\hat\Sigma_M$, is the maximum likelihood estimator in the model where $P_\theta = \mathcal{N}_q(\mu_\theta, c_\theta\Sigma)$ with $c_\theta > 0$ and $\Sigma \in \mathbb{R}^{q\times q}$ symmetric and positive definite with $\det(\Sigma) = 1$. For the calculations we use that $\hat\Sigma_M$ is the solution of the fixed point equation

$$\Sigma = q\sum_{b\in\mathcal{Y}} \frac{N_b}{n}\frac{M_b}{\operatorname{trace}(\Sigma^{-1}M_b)}$$

with $M_b := \sum_{j\in\mathcal{G}_b}(X_j - \hat\mu_b)(X_j - \hat\mu_b)^\top$. The second M estimator, $\hat\Sigma_{\text{sym}}$, is a generalization for more than one group of the symmetrized version of Tyler's M estimator, as it is defined in Dümbgen (1998). It is the solution of the fixed point equation

$$\Sigma = \frac{2q}{n-L} \sum_{b \in \mathcal{Y}} \frac{1}{N_b} \sum_{\substack{j,k \in \mathcal{G}_b \\ j<k}} \frac{(X_j - X_k)(X_j - X_k)^\top}{(X_j - X_k)^\top \Sigma^{-1} (X_j - X_k)}.$$

(If observations X_i within one group $\mathcal{G}_b$ are identical, the previous equation has to be modified somewhat.) For more details on M estimators we refer to Dümbgen, Pauly, and Schweizer (2015) and Dümbgen, Nordhausen, and Schuhmacher (2016).

2.2. Nearest neighbors and weighted nearest neighbors

Suppose that $d(\cdot,\cdot)$ is some metric on $\mathcal{X}$. Let $B(x,r) := \{y \in \mathcal{X} \colon d(x,y) \le r\}$, and for a fixed positive integer $k \le n$ define

$$\hat{r}_k(x) = \hat{r}_k(x,\mathcal{D}) := \min\{r \ge 0 \colon \#\{i \le n \colon X_i \in B(x,r)\} \ge k\}.$$

Further let $\hat{P}_\theta(B)$ denote the empirical distribution of the points X_i with $Y_i = \theta$, i.e.,

$$\hat{P}_\theta(B) := \frac{1}{N_\theta} \#\{i \in \mathcal{G}_\theta \colon X_i \in B\} \qquad \text{for } B \subset \mathcal{X}.$$

Then the k nearest neighbor estimator of $w_\theta(x)$ is given by

$$\hat{w}_\theta(x,\mathcal{D}) := \frac{\hat{w}_\theta \hat{P}_\theta(B(x,\hat{r}_k(x)))}{\sum_{b \in \mathcal{Y}} \hat{w}_b \hat{P}_b(B(x,\hat{r}_k(x)))}$$

with certain estimators $\hat{w}_b = \hat{w}_b(\mathcal{D})$ of w_b. In the package **pvclass** we use $\hat{w}_b = N_b/n$ and get

$$\begin{aligned} \hat{w}_\theta(x,\mathcal{D}) &:= \frac{\#\{i \in \mathcal{G}_\theta \colon d(x,X_i) \le \hat{r}_k\}}{\#\{i \le n \colon d(x,X_i) \le \hat{r}_k\}} \\ &= \sum_{i=1}^n 1\{d(x,X_i) \le \hat{r}_k\} 1\{Y_i = \theta\} \Big/ \sum_{i=1}^n 1\{d(x,X_i) \le \hat{r}_k\}. \end{aligned}$$

The k nearest neighbor estimator weights all k nearest neighbors of the observation X equally. However it would be reasonable to assign larger weights to training observations which are closer to X. **pvclass** also offers a weighted nearest neighbor estimator for $w_\theta(x)$. To compute this, we first order the training data according to their distance to X and then assign descending weights to them. Let $W(1) \ge W(2) \ge \cdots \ge W(n) \ge 0$ be given weights and

$$\hat{R}(x,X_i) := \#\{j \le n \colon d(x,X_j) \le d(x,X_i)\},$$

the rank of training observation X_i according to its distance to x. The weighted nearest neighbor estimator for $w_\theta(x)$ is then defined as

$$\hat{w}_\theta(x,\mathcal{D}) := \sum_{i=1}^n W(\widehat{R}(x,X_i)) 1\{Y_i = \theta\} \Big/ \sum_{i=1}^n W(\widehat{R}(x,X_i)).$$

In **pvclass** either the linear weight function

$$W(i) = \max(1 - (i/n)/\tau, 0)$$

or the exponential weight function

$$W(i) = (1 - i/n)^\tau$$

with a tuning parameter $\tau > 0$ can be used. For the linear weight function, τ should be in $(0, 1]$. Alternatively, one can specify arbitrary weights with an n dimensional vector W.

Often the different variables of a data set are measured on different scales. To take this into account, **pvclass** offers besides the fixed Euclidean distance two data-driven distances which are scale invariant. The first is the data driven Euclidean distance where we divide each component of X by its sample standard deviation and then use the Euclidean distance. For the second we assume that all classes have a common covariance matrix and estimate this with one of the estimators described in Section 2.1. Then we use the Mahalanobis distance with respect to the estimated covariance matrix $\hat{\Sigma}$, i.e., we use the distance

$$D_{\hat{\Sigma}}(x, y) := \sqrt{(x - y)^\top \hat{\Sigma}^{-1} (x - y)}.$$

2.3. Penalized multicategory logistic regression

One of our versions of penalized multicategory logistic regression is similar to the regularized multinomial regression introduced by Friedman, Hastie, and Tibshirani (2010), the other one is a variation of the procedure of Zhu and Hastie (2004). Let $\mathcal{X} = \mathbb{R}^q$ and X contain the values of $q - 1$ numerical or binary variables and a constant term. Our temporary working assumption is that

$$\mathsf{P}(Y = y \,|\, X = x) = \exp(b_y^\top x) \Big/ \sum_{z=1}^{L} \exp(b_z^\top x) \tag{6}$$

for unknown parameters $b_1, b_2, \ldots, b_L \in \mathbb{R}^q$. With estimators $\hat{b}_z = \hat{b}_z(\mathcal{D})$ to be specified below, p values are computed with the test statistics

$$T_\theta(x, \mathcal{D}) = \log\Big(\sum_{z=1}^{L} \exp(\hat{b}_z^\top x)\Big) - \hat{b}_\theta^\top x.$$

The parameter $b = \big(b_1^\top, b_2^\top, \ldots, b_L^\top\big)^\top \in \mathbb{R}^{Lq}$ is estimated via penalized maximum likelihood. We pretend temporarily that the training observations (X_i, Y_i) are independent copies of (X, Y) satisfying (6). The corresponding negative conditional log-likelihood function, given the X_i, is given by

$$\Lambda(b) := \sum_{i=1}^{n} \Big(-b_{Y_i}^\top X_i + \log\Big(\sum_{z=1}^{L} \exp(b_z^\top X_i) \Big) \Big).$$

The parametrization in (6) is not unique, because $P(Y = \theta \,|\, X = x)$ remains unchanged if we add one and the same arbitrary vector to all parameters b_z. One way to guarantee uniqueness of the parameter b is to require

$$\sum_{z=1}^{L} b_z \;=\; 0.$$

More generally, with $b_z = (b_{j,z})_{j=1}^q$ write $b_{[j]} := \big(b_{j,z}\big)_{z=1}^L$. With $1_L := (1, 1, \ldots, 1)^\top \in \mathbb{R}^L$, the previous condition means that

$$1_L^\top b_{[j]} \;=\; 0 \tag{7}$$

for all $j = 1, 2, \ldots, q$. To enforce (7) at least for some j we can add

$$R_0(b) \ := \ 2^{-1} \sum_{j=1}^{q} \sigma_j \big(1_L^\top b_{[j]}\big)^2$$

with numbers $\sigma_j \geq 0$ to $\Lambda(b)$. The choice of σ_j will depend on further regularization terms.

Regularization 1: Penalizing subvectors. For logistic regression there are various good reasons to regularize the functional Λ or $\Lambda + R_0$. One is to avoid numerical problems. Another is to guarantee existence of a minimizer in cases where Λ alone has no minimizer. This happens if one subgroup $\{X_i \colon Y_i = b_o\}$ is separated from $\{X_i \colon Y_i \neq b_o\}$ by a hyperplane. Moreover, we want to favor parameter vectors with only few large components. A first way to do this would be to add the penalty

$$\sum_{j=1}^{q} \tau_j \|b_{[j]}\|$$

with numbers $\tau_j \geq 0$ to $\Lambda(b) + R_0(b)$. Here and throughout, $\|\cdot\|$ denotes Euclidean norm. This regularization is motivated by Tibshirani's (1996) LASSO and similar in spirit to penalized logistic regression as proposed by Zhu and Hastie (2004). The latter authors used $\|\theta_{[j]}\|^2$ instead of $\|\theta_{[j]}\|$. Note that $\|b_{[j]} - c\,1_L\|^2$ becomes minimal if c equals the mean $1_L^\top b_{[j]}/L$. Hence minimizing $\Lambda(b) + R_0(b) + R_1(b)$ enforces Condition (7) whenever $\sigma_j + \tau_j > 0$ for all j.

Regularization 2: Component-wise penalties. A simple form of regularization, analogous to Tibshirani's (1996) LASSO is to add the penalty

$$\sum_{j=1}^{q} \tau_j \sum_{z=1}^{L} |b_{j,z}| \ = \ \sum_{j=1}^{q} \tau_j \|b_{[j]}\|_1$$

to $\Lambda(b) + R_0(b)$.

Choice of σ_j and τ_j. The three versions of penalized logistic regression are available in **pvclass**, specified by the parameters `pen.method` and τ_o:

`pen.method`	R	σ_j	τ_j
`vectors`	$R_0 + R_1$	1	$\tau_o S_j$
`simple`	$R_0 + R_2$	$1\{S_j = 0\}$	$\tau_o S_j$
`none`	R_0	1	–

Here S_j is the sample standard deviation (within groups) of the j-th components of X_i. For results about the existence of unique minimizers we refer to Zumbrunnen (2014).

3. Implementation and main functions

The package **pvclass** (Zumbrunnen and Dümbgen 2017) was written in the R programming language (R Core Team 2017) and depends on the recommended package **Matrix** (Bates and Mächler 2017).

The main functions of **pvclass** compute p values for the potential class memberships of new observations (`pvs`) as well as cross-validated p values for training data (`cvpvs`). With the function `analyze.pvs` the package **pvclass** also provides graphical displays and quantitative analyses of the p values.

In this section we illustrate the main functions with the data set `buerk` of Section 1.6. We use the mortality as class label Y. The original data set contains 21556 observations. To get a smaller data set, which is easier to handle, we take all the 662 observations with $Y = 1$ and with the function `sample` we choose randomly $3 \cdot 662$ observations with $Y = 0$. For the test data set we choose 100 observations from each of the classes. So we end up with a training data set containing 2448 observations, whereof 562 belong to class 1.

```
R> library("pvclass")
R> data("buerk")
R> set.seed(3)
R> X.raw <- as.matrix(buerk[, 1:21])
R> Y.raw <- buerk[, 22]
R> n0.raw <- sum(1 - Y.raw)
R> n1 <- sum(Y.raw)
R> n0 <- 3 * n1
R> X0 <- X.raw[Y.raw == 0, ]
R> X1 <- X.raw[Y.raw == 1, ]
R> tmpi0 <- sample(1:n0.raw, size = 3 * n1, replace = FALSE)
R> tmpi1 <- sample(1:n1, size = n1, replace = FALSE)
R> Xtrain <- rbind(X0[tmpi0[1:(n0 - 100)], ], X1[1:(n1 - 100), ])
R> Ytrain <- c(rep(0, n0 - 100), rep(1, n1 - 100))
R> Xtest <- rbind(X0[tmpi0[(n0 - 99):n0], ], X1[(n1 - 99):n1, ])
R> Ytest <- c(rep(0, 100), rep(1, 100))
```

3.1. Classify new observations

The function `pvs` computes nonparametric p values for the potential class memberships of new observations. It returns a matrix `PV` containing the p values. Precisely, for each new observation `NewX[i,]` and each class `b` the number `PV[i,b]` is a p value for the null hypothesis that $Y[i] = b$. With the option `method = "method"` or using directly on of the functions `pvs.method` one can choose one of the test statistics of Section 2. For the following example we use the weighted nearest neighbor statistic with an exponential weight function and `tau = 10`.

```
R> PV <- pvs(NewX = Xtest, X = Xtrain, Y = Ytrain, method = "wnn",
+    wtype = "exponential", tau = 10)
R> head(PV)
```

```
             1           2
[1,] 0.8839428 0.012433393
[2,] 0.9167992 0.007104796
[3,] 0.1314255 0.538188277
```

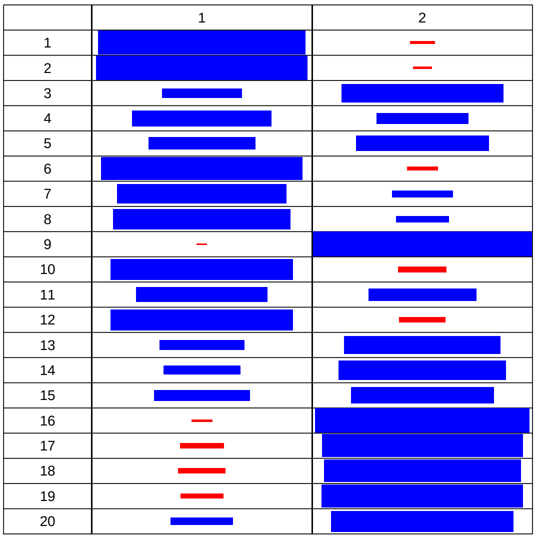

Figure 4: Illustration of the p values without indicating the class labels of the test data.

```
[4,] 0.3990461 0.174067496
[5,] 0.2342342 0.364120782
[6,] 0.8335983 0.019538188
```

Next we illustrate the p values graphically with the function `analyze.pvs` using the first ten observations of each class.

```
R> analyze.pvs(pv = PV[c(1:10, 101:110), ], alpha = 0.05)
```

For each p value a rectangle with an area proportional to the p value is drawn, see Figure 4. The rectangle is blue if the p value is greater than `alpha` and red otherwise. If we specify the class labels of the test data as in the next example, then the data are sorted per class and the class labels are shown in the plot, see Figure 5. If $L \leq 3$ the function `analyze.pvs` also prints the empirical pattern probabilities $\hat{\mathcal{P}}_\alpha(b, S)$ for all subsets $S \subset \mathcal{Y}$. Otherwise it prints the empirical conditional inclusion probabilities $\hat{\mathcal{I}}_\alpha(b, \theta)$ for all combinations of b and θ and the empirical pattern probabilities for $S = \emptyset, \mathcal{Y}$ and $\{\theta\}$ for all $\theta \in \mathcal{Y}$. Additionally ROC functions are plotted by default. We suppress this here with the argument `roc = FALSE`. An example of the ROC function plot can be found in the next section.

```
R> analyze.pvs(pv = PV[c(1:10, 101:110), ], Y = Ytest[c(1:10, 101:110)],
+    roc = FALSE)
```

```
b   P(b,{}) P(b,{1}) P(b,{2}) P(b,{1,2})
  1       0      0.4      0.1        0.5
  2       0      0.1      0.4        0.5
```

In this example the empirical pattern probabilities for uniquely correct classification $\hat{\mathcal{P}}_\alpha(b, \{b\})$ are 0.4 for both classes.

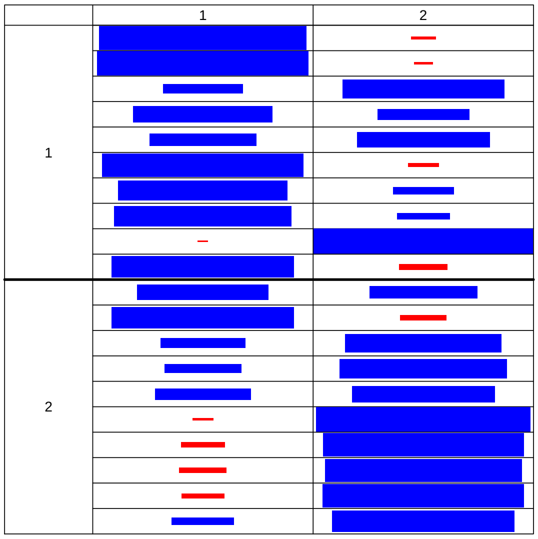

Figure 5: Illustration of the p values with class labels of the test data.

3.2. Cross-validated p values

The function `cvpvs` returns a matrix `PV` containing cross-validated nonparametric p values for the potential class memberships of the training data. Precisely, for each feature vector `X[i,]` and each class `b` the number `PV[i,b]` is a p value for the null hypothesis that $Y[i] = b$. For the following example we use the logistic regression with penalty parameter `tau.o = 2`. The option `progress = FALSE` suppresses the printing of the computation progress.

```
R> PV.cv <- cvpvs(X = Xtrain, Y = Ytrain, method = "logreg", tau.o = 2,
+    progress = FALSE)
R> PV.cv[1:3, ]

             1           2
[1,] 0.1320255 0.120781528
[2,] 0.2221633 0.044404973
[3,] 0.9772004 0.001776199

R> PV.cv[2001:2003, ]

              1         2
[1,] 0.00317965 0.8167260
[2,] 0.06677266 0.2526690
[3,] 0.01642819 0.6281139
```

The cross-validated p values can be illustrated graphically with `analyze.pvs` the same way as the p values for the new observations. In the following example we suppress the plot of the p values and get only the plot of the ROC functions, see Figure 6. The output shows the empirical pattern probabilities $\hat{\mathcal{P}}_\alpha(b, S)$ as described in Section 3.1.

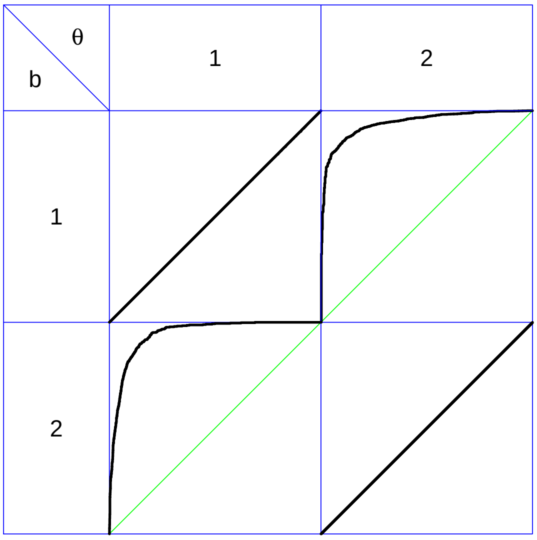

Figure 6: ROC curves of the cross-validated p values.

```
R> analyze.pvs(pv = PV.cv, Y = Ytrain, pvplot = FALSE, cex = 1.3)

b   P(b,{})   P(b,{1})   P(b,{2}) P(b,{1,2})
  1       0 0.79586426 0.04984093  0.1542948
  2       0 0.04982206 0.64412811  0.3060498
```

In this example the empirical pattern probabilities for uniquely correct classification are $\hat{\mathcal{P}}_\alpha(0, \{0\}) = 0.788$ for class 0 and $\hat{\mathcal{P}}_\alpha(1, \{1\}) = 0.721$ for class 1.

4. Choice of tuning parameters

Some of the previous test statistics depend on a tuning parameter $\tau > 0$, i.e., $T_\theta(x, \mathcal{D}) = T_\theta^{(\tau)}(x, \mathcal{D})$. Our goal is to choose this parameter in a data-driven way such that overfitting of the training data is avoided while symmetry in $(X_i)_{i \in \mathcal{G}_\theta}$ is preserved.

The following criterion turned out to be quite useful: In view of our specific construction of the p values, $T_\theta^{(\tau)}(X, \mathcal{D}(X, \theta))$ should take relatively big values whenever $Y \neq \theta$. Hence we compute for all training observations with $Y_i \neq \theta$ the test statistic

$$T_\theta^{(\tau)}(X_i, \mathcal{D}_i(X_i, X, \theta)),$$

where $\mathcal{D}_i(X_i, X, \theta)$ denotes the training data after adding the observation (X, θ) and setting the class label of observation X_i to θ. Then we take the sum of these values,

$$S(\tau, X, \theta) := \sum_{i\colon Y_i \neq \theta} T_\theta^{(\tau)}(X_i, \mathcal{D}_i(X_i, X, \theta)),$$

and search for a parameter $\tau^*(X, \theta, \mathcal{D})$ maximizing $S(\tau, X, \theta)$ over a certain set of candidates for τ.

Implementations in pvclass. The functions for the nearest neighbor methods accept vectors for the parameters k and τ, respectively. If a vector is provided, the function searches for the best of the given parameters. If no parameter is provided (`k = NULL` or `tau = 0`) certain default vectors are used.

In connection with penalized logistic regression, note that to determine the optimal tuning parameters for a new observation X and all potential class memberships, the test statistic has to be computed $(L-1) \cdot n \cdot l$ times, where l is the number of tuning parameters from which we want to choose the optimal one. This can be very computer-intensive, especially for penalized logistic regression in high dimensions. For the latter method we observed in simulated and real data examples that the p values do not depend too sensitively on the choice of the penalty parameter and $S(\tau_o, X, \theta)$ is unimodal in τ_o. Therefore if `find.tau == TRUE`, we only consider few values for τ_o on a logarithmic grid as follows: We start with a certain value τ_o (default 10) and compare $S(\tau_o, X, \theta)$ with $S(\tau_o \cdot \delta, X, \theta)$ for some $\delta > 1$ (default 2). While $S(\tau_o \cdot \delta, X, \theta) > S(\tau_o, X, \theta)$ and τ_o is smaller than a certain threshold $\tau_{\max}$ (default 100) we replace τ_o with $\tau_o \cdot \delta$. If in the very beginning $S(\tau_o, X, \theta) \geq S(\tau_o \cdot \delta, X, \theta)$, we compare $S(\tau_o, X, \theta)$ with $S(\tau_o/\delta, X, \theta)$. While $S(\tau_o/\delta, X, \theta) > S(\tau_o, X, \theta)$ and τ_o is larger than a certain threshold $\tau_{\min}$ (default 1) we replace τ_o with τ_o/δ.

4.1. Numerical examples

We illustrate the procedure with two simulated data sets.

Example 1. Consider $L = 2$ classes with $P_\theta = \mathcal{N}_{100}(\mu_\theta, I_{100})$, where $\mu_1 = (1, 0.5, 0.25, 0.125, 0, \ldots, 0)^\top$ and $\mu_2 = -\mu_1$. We simulated 100 training sets with $N_1 = N_2 = 100$ observations per class. For each training set we simulated a test set again with $N_1 = N_2 = 100$ observations per class and computed $\pi_\theta^{(\tau)}(X, \mathcal{D}(X, \theta))$ for each test observation and $\tau = 1, 2, 4, 8, 16, 32, 64$. Here $\pi_\theta^{(\tau)}$ denotes the p value based on penalized logistic regression. Additionally we computed τ^* for 20 training sets.

Figure 7 shows boxplots of the distributions of the rates of uniquely correct classified test observations for different values of τ. The misclassification rates depend heavily on the training set. Nevertheless our method is very stable and chooses for 99.8% of the test observations $\tau^* = 16$, which has the highest median rate of uniquely correct classification.

Example 2. Next we consider an example with $L = 5$ classes, $P_\theta = \mathcal{N}_{100}(\mu_\theta, I_{100})$ and $\mu_\theta = (0_{4(\theta-1)}, 2, 1, 0.5, 0.25, 0, \ldots, 0)^\top$, where 0_j denotes the row vector with j zeros. We simulated 100 training sets with $N_\theta = 100$ per class. For each training set we simulated a test set with $N_\theta = 40$ test observations per class and computed $\pi_\theta^{(\tau)}(X, \mathcal{D}(X, \theta))$ for each test observation and $\tau = 1, 2, 4, 8, 16, 32, 64$. Additionally we computed τ^* for 5 training sets.

Figure 8 shows boxplots of the distributions of the rates of uniquely correct classified test observations for different values of τ. As in the previous example the misclassification rates depend heavily on the training set. Again our method is very stable and chooses for all test observations $\tau^* = 16$, which has the highest median rate of uniquely correct classification.

Example 3 (Buerk)**.** For the hospital data described in Section 3 we computed p values for $\tau = 1, 2, \ldots, 100$. The amount of regularization had no influence on the misclassification rates in this example.

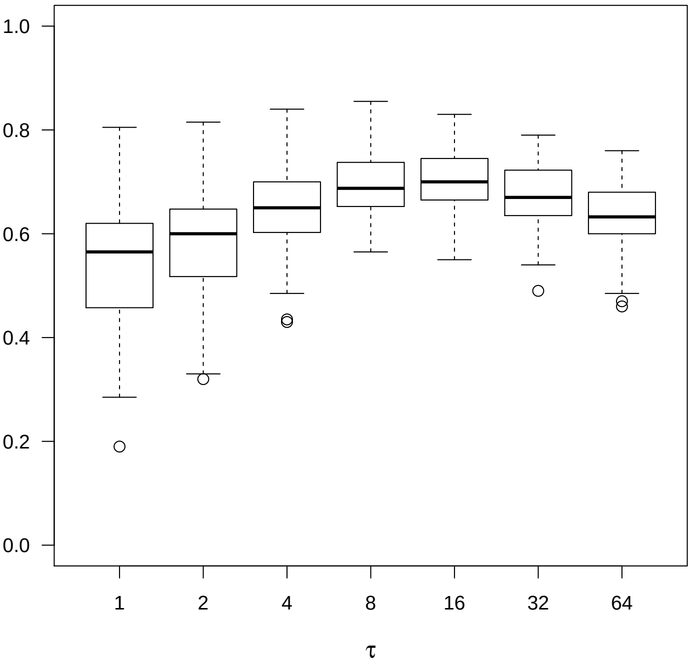

Figure 7: Distributions of the rates of uniquely correct classified test observations for different values of τ for Example 1.

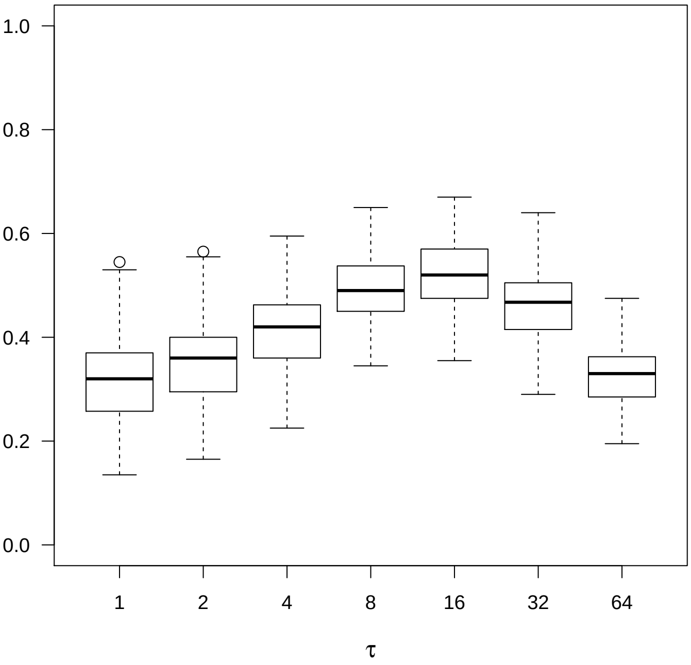

Figure 8: Distributions of the rates of uniquely correct classified test observations for different values of τ for Example 2.

5. Relation to other classifiers and packages

There are numerous software packages for classification. It should be stressed that our package does *not* provide just another classifier. Our aim is to provide a first open-source implementation of nonparametric p values for classification. Indeed one can can easily combine our package with one's favorite classifier, e.g. based on neural nets or support vector machines, as long as it involves some plausibility or implausibility measures for class memberships. Precisely, suppose that for given training data $\mathcal{D}$ your favorite classifier provides explicitly or implicitly a matrix

$$\begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1L} \\ T_{21} & T_{22} & \cdots & T_{2L} \\ \vdots & \vdots & & \vdots \\ T_{n1} & T_{n2} & \cdots & T_{nL} \end{bmatrix} \tag{8}$$

where T_{iy} measures the implausibility of "$Y_i = y$" for the feature vector X_i. (If T_{iy} measures the plausibility of "$Y_i = y$", just replace T_{iy} with $-T_{iy}$.) To compute a p value $\pi_\theta(X, \mathcal{D})$ for a new observation, one has to compute this matrix with the augmented data matrix $\mathcal{D}(X, \theta)$ in place of $\mathcal{D}$. This results in a matrix with an additional first row $[T_{01}, T_{02}, \ldots, T_{0L}]$ corresponding to the additional observation $(X_0, Y_0) = (X, \theta)$. Note that in general the other n rows will be different from the original n rows of (8). Then the p value $\pi_\theta(X, \mathcal{D})$ is given by

$$\pi_\theta(X, \mathcal{D}) \;=\; (N_\theta + 1)^{-1} \sum_{i=0}^{n} 1\{Y_i = \theta, T_{i\theta} \geq T_{0\theta}\}.$$

To analyze the training data itself by means of cross-validated p values, one has to compute the matrix (8) $1 + n(L-1)$ times: For $i = 1, 2, \ldots, n$, the original data set $\mathcal{D}$ and the resulting matrix (8) yield the cross-validated p value

$$\pi_\theta(X_i, \mathcal{D}_i) \;=\; N_\theta^{-1} \sum_{j=1}^{n} 1\{Y_j = \theta, T_{j\theta} \geq T_{i\theta}\} \quad \text{for } \theta = Y_i.$$

For $\theta \neq Y_i$ one may use the same formula, but one has to recalculate the matrix (8) after replacing (X_i, Y_i) in $\mathcal{D}$ with (X_i, θ). Note that this affects the group sizes N_θ and N_{Y_i}, too. The resulting matrix $\big(\pi_\theta(X_i, \mathcal{D}_i)\big)_{i,\theta} \in [0,1]^{n \times L}$ of crossvalidated p values may be analyzed by means of `analyze.pvs`.

These remarks indicate that the computation of our p values is necessarily more involved than the computation of a traditional classifier. Roughly saying, to evaluate one future feature vector, one has to compute the underlying classifier L times, and for the computation of all cross-validated p values one needs nL computations of the classifier. On the other hand, the underlying data sets $\mathcal{D}$, $\mathcal{D}(X, \theta)$, $\mathcal{D}_i$ etc. are very similar, and in our implementation in **pvclass** we utilize various tricks to exploit these redundancies.

Acknowledgments

This work was supported by the research group FOR916 of the Swiss National Science Foundation (SNF) and the Deutsche Forschungsgemeinschaft (DFG).

References

Altman DG, Bland JM (1994). "Statistics Notes: Diagnostic Tests 1: Sensitivity and Specificity." *British Medical Journal*, **308**(6943), 1552. ISSN 0959-8138. `doi:10.1136/bmj.308.6943.1552`.

Bates D, Mächler M (2017). ***Matrix**: Sparse and Dense Matrix Classes and Methods.* R package version 1.2-10, URL `https://CRAN.R-project.org/package=Matrix`.

Dümbgen L (1998). "On Tyler's M-Functional of Scatter in High Dimension." *Annals of the Institute of Statistical Mathematics*, **50**(3), 471–491. `doi:10.1023/a:1003573311481`.

Dümbgen L, Igl BW, Munk A (2008). "p-Values for Classification." *Electronic Journal of Statistics*, **2**, 468–493. `doi:10.1214/08-ejs245`.

Dümbgen L, Nordhausen K, Schuhmacher H (2016). "New Algorithms for M-Estimation of Multivariate Scatter and Location." *Journal of Multivariate Analysis*, **144**, 200–217. `doi:10.1016/j.jmva.2015.11.009`.

Dümbgen L, Pauly M, Schweizer T (2015). "M-Functionals of Multivariate Scatter." *Statistics Surveys*, **9**, 32–105. `doi:10.1214/15-ss109`.

Fawcett T (2006). "An Introduction to ROC Analysis." *Pattern Recognition Letters*, **27**(8), 861–874. `doi:10.1016/j.patrec.2005.10.010`.

Friedman JH, Hastie T, Tibshirani R (2010). "Regularization Paths for Generalized Linear Models via Coordinate Descent." *Journal of Statistical Software*, **33**(1), 1–22. `doi:10.18637/jss.v033.i01`.

R Core Team (2017). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Scott C (2007). "Performance Measures for Neyman-Pearson Classification." *IEEE Transactions on Information Theory*, **53**, 2852–2863. `doi:10.1109/tit.2007.901152`.

Tibshirani R (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society B*, **58**(1), 267–288.

Zhao A, Feng Y, Wang L, Tong X (2015). "Neyman-Pearson Classification under High-Dimensional Settings." Preprint, URL `http://arxiv.org/abs/1508.03106`.

Zhu J, Hastie T (2004). "Classification of Gene Microarrays by Penalized Logistic Regression." *Biostatistics*, **5(14)**, 427–443. `doi:10.1093/biostatistics/5.3.427`.

Zumbrunnen N (2009). *p-Values for Weighted Nearest-Neighbour Classifiers.* Master Thesis, University of Bern.

Zumbrunnen N (2014). *p-Values for Classification – Computational Aspects and Asymptotics.* Ph.D. thesis, University of Bern. URL `http://boris.unibe.ch/id/eprint/53585`.

Zumbrunnen N, Dümbgen L (2017). ***pvclass**: p-Values for Classification.* R package version 1.4, URL `https://CRAN.R-project.org/package=pvclass`.

Affiliation:

Niki Zumbrunnen, Lutz Dümbgen
Institute for Mathematical Statistics and Actuarial Science
University of Bern
Alpeneggstrasse 22
3012 Bern, Switzerland
E-mail: `niki.zumbrunnen@gmail.com`, `lutz.duembgen@stat.unibe.ch`
URL: `http://www.stat.unibe.ch/duembgen`

Journal of Statistical Software `http://www.jstatsoft.org/`
published by the Foundation for Open Access Statistics `http://www.foastat.org/`

June 2017, Volume 78, Issue 4 *Submitted:* 2014-07-04
`doi:10.18637/jss.v078.i04` *Accepted:* 2016-04-03