Fitting Aggregation Functions to Data: Part II – Idempotization

Maciej Bartoszuk[1], Gleb Beliakov[2], Marek Gagolewski*[3,1], and Simon James[2]

[1] Faculty of Mathematics and Information Science, Warsaw University of Technology, ul. Koszykowa 75, 00-662 Warsaw, Poland, bartoszukm@mini.pw.edu.pl
[2] School of Information Technology, Deakin University, 221 Burwood Hwy, Burwood, Victoria 3125, Australia, {gleb,sjames}@deakin.edu.au
[3] Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland, gagolews@ibspan.waw.pl

Abstract. The use of supervised learning techniques for fitting weights and/or generator functions of weighted quasi-arithmetic means – a special class of idempotent and nondecreasing aggregation functions – to empirical data has already been considered in a number of papers. Nevertheless, there are still some important issues that have not been discussed in the literature yet. In the second part of this two-part contribution we deal with a quite common situation in which we have inputs coming from different sources, describing a similar phenomenon, but which have not been properly normalized. In such a case, idempotent and nondecreasing functions cannot be used to aggregate them unless proper pre-processing is performed. The proposed idempotization method, based on the notion of B-splines, allows for an automatic calibration of independent variables. The introduced technique is applied in an R source code plagiarism detection system.

Keywords: Aggregation functions, weighted quasi-arithmetic means, least squares fitting, idempotence

1 Introduction

Idempotent aggregation functions – mappings like $\mathsf{F} : [0,1]^n \to [0,1]$ being nondecreasing in each variable and fulfilling $\mathsf{F}(x,\dots,x) = x$ for all $x \in [0,1]$ – have numerous applications, including areas like decision making, pattern recognition, and data analysis, compare, e.g., [8,11].

For a fixed $n \ge 2$, let $\mathbf{w} \in [0,1]^n$ be a weighting vector, i.e., one with $\sum_{i=1}^n w_i = 1$. In the first unit [1] of this two-part contribution we dealt with two important practical issues concerning supervised learning of weights of weighted quasi-arithmetic means with a known continuous and strictly monotone generator $\varphi : [0,1] \to \bar{\mathbb{R}}$, that is idempotent aggregation functions given for arbitrary

* Corresponding author.

$\mathbf{x} \in [0,1]^n$ by the formula:

$$\mathsf{WQAMean}_{\varphi,\mathbf{w}}(\mathbf{x}) = \varphi^{-1}\left(\sum_{i=1}^{n} w_i \varphi(x_i)\right).$$

First of all, we observed that most often researchers considered an approximate version of weight learning tasks and relied on a linearization of input variables, compare, e.g., [7]. Therefore, we discussed possible implementations of the exact fitting procedure and identified some cases where linearization leads to solutions of significantly worse quality in terms of the squared error between the desired and generated outputs. Secondly, we noted that the computed models may overfit a training data set and perform weakly on test and validation samples. Thus, some regularization methods were proposed to overcome this limitation. We indicated that due to the typical constraints on the weighting vector (nonnegative coefficients summing up to 1), not all the regularization techniques known from machine learning [13] can be applied, but – on the other hand – we may consider new, quite different ones instead.

Assume that we are given $m \geq n$ input vectors in a form $\mathbf{X} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}] \in [0,1]^{n\times m}$ together with m desired output values $\mathbf{Y} = [y^{(1)}, \ldots, y^{(m)}] \in [0,1]^{1\times m}$. For simplicity, we shall focus only on fitting weighted arithmetic means to $(\mathbf{X}, \mathbf{Y})$ using the least squared error criterion, noting that the key ideas presented further on can be extrapolated to other settings. And so, we aim to:

$$\text{minimize } \sum_{j=1}^{m}\left(\sum_{i=1}^{n} w_i x_i^{(j)} - y^{(j)}\right)^2 \quad \text{w.r.t. } \mathbf{w},$$

under the constraints that $\mathbf{1}^T\mathbf{w} = 1$ and $\mathbf{w} \geq \mathbf{0}$, compare, e.g., [5].

However, let us presume that the input values represent m measurements of the same phenomenon done via n different methods which output numeric values that cannot be directly compared: each of them is defined up to a strictly increasing and continuous transformation and a proper input data idempotization scheme has to be applied prior to fitting a model.

Example 1. In the R [15] language plagiarism detection system described in [2,3], the similarity of a source code chunk pair is assessed via $n = 4$ diverse methods. Each of them reflects quite different ideas behind what plagiarism really is in its nature:

- $x_1^{(j)}$ – is based on the so-called program dependence graphs (PDGs),
- $x_2^{(j)}$ – simply computes the Levenshtein distance between source texts,
- $x_3^{(j)}$ – determines the longest common subsequence of two corresponding token strings,
- $x_4^{(j)}$ – compares the number of common R function calls.

Each of the four variables is a real number in $[0,1]$, but the way they have been mapped to the unit interval is quite arbitrary – in fact, initially, we should treat them as values on an ordinal – and not interval – scale. Most common machine learning methods (e.g., regression and classification) should work quite well on such data, but the construction of aggregation functions does not make much sense on raw inputs of this kind. However, if appropriate strictly monotone transformations $\varphi_1, \ldots, \varphi_4 : [0,1] \to [0,1]$ were determined, we would have the values normalized in such a way that they can be compared to each other ($x_1^{(j)} = 0.5$ would denote the same similarity level as $x_2^{(j)} = 0.5$, hence we could expect – by idempotence – the aggregated similarity to be equal to 0.5 too). Moreover – by nondecreasingness – we could be sure that any increase of a similarity level never leads to a decrease in the aggregated similarity – a constraint not guaranteed by any classical machine learning method.

Therefore, in this part of the contribution, we deal with the problem which aims to construct an *idempotized* model for a given data set, that is we are going to:

$$\text{minimize} \sum_{j=1}^{m} \left(\sum_{i=1}^{n} w_i \tilde{x}_i^{(j)} - y^{(j)} \right)^2 \quad \text{w.r.t. } \mathbf{w},$$

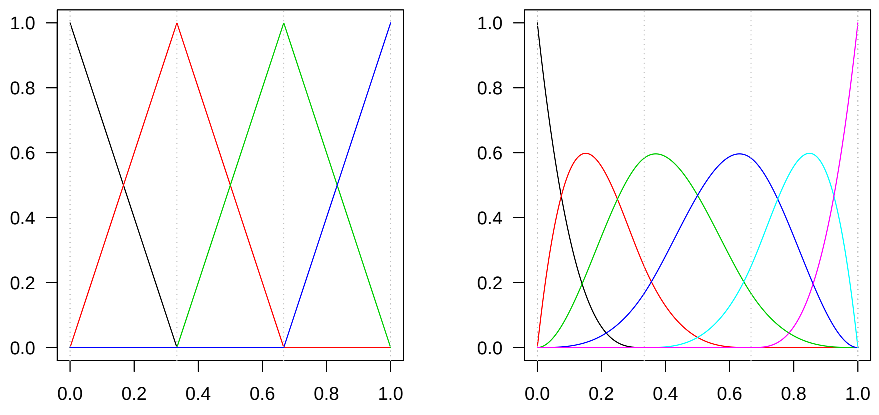
under the standard constraints on a weighting vector $\mathbf{w}$, where $\tilde{x}_i^{(j)} = \varphi_i(x_i^{(j)})$ for some automatically generated monotone and continuous $\varphi_1, \ldots, \varphi_n : [0,1]^n \to [0,1]$. This enables us to develop idempotent aggregation functions-based regression (and, as a by-product, binary classification) models [13], which – by construction – fulfill some desired algebraic properties, and hence posses a better, more intuitive interpretation than classical approaches on data sets similar to the one in Example 1.

The paper is set out as follows. In the next section we recall the notion of B-splines, which we shall use for modeling the φ_i functions. Section 3 discusses the proposed idempotization and aggregation function fitting procedure, together with some key implementation details. Note that in order to increase its performance on test samples, the model employs a regularization term which we discussed in the first part of this contribution [1]. Section 4 discusses the results of an experimental study conducted on the aforementioned plagiarism detection system data. Finally, Section 5 summarizes the paper and indicates future research directions.

2 B-splines

In a quasi-arithmetic mean fitting task, Beliakov et al. [4,6,9,10] rely on the notion of B-splines to model an unknown generator function.

Let $p \geq 1$ and $\mathbf{t} = (t_1, \ldots, t_k)$ be an increasingly ordered *knot vector* of length k for some $k \geq 0$ such that $0 < t_i < t_{i+1} < 1$ for all $i = 1, \ldots, k$. For simplicity, we presume that $t_i = 0$ for $i < 1$ and $t_i = 1$ whenever $i > k$.



(a) $p = 1, k = 2$ (b) $p = 3, k = 2$

Fig. 1. Exemplary B-spline basis functions $N^{\mathbf{t}}_{j-p,p}(x)$ as a function of x, $j = 1, \ldots\ldots, p + k + 1$; $\mathbf{t}$ is a vector of equidistant knots, k is the number of the internal knots, while p is the polynomial degree.

Definition 1. B-spline basis functions *for* $j = 0, \ldots, p$ *and* $x \in [0, 1]$ *are defined recursively as:*

$$N^{\mathbf{t}}_{i,j}(x) = \begin{cases} 1 \text{ if } x \in [t_{i-1}, t_i[, \\ 0 \text{ otherwise}, \end{cases} \qquad (j = 0)$$

$$N^{\mathbf{t}}_{i,j}(x) = \frac{x - t_{i-1}}{t_{i+j-1} - t_{i-1}} N^{\mathbf{t}}_{i,j-1}(x) + \frac{t_{i+j} - x}{t_{i+j} - t_i} N^{\mathbf{t}}_{i+1,j-1}(x), \qquad (j > 0)$$

with convention $\cdot/0 = 0$.

Note that a vector of equidistant knots is a quite common setting. Figure 1 depicts exemplary B-spline basis functions.

Additionally, let $\mathbf{v} \in [0, 1]^\eta$ be a vector of *control points*, where $\eta = p + k + 1$.

Definition 2. *A function* $B^{\mathbf{t}}_{\mathbf{v}} : [0, 1] \to [0, 1]$ *given by:*

$$B^{\mathbf{t}}_{\mathbf{v}}(x) = \sum_{i=1}^{\eta} v_i N^{\mathbf{t}}_{i-p,p}(x) \tag{1}$$

is called a nonperiodic B-spline of degree p *based on a knot vector* $\mathbf{t}$ *and generated by a control points vector* $\mathbf{v}$, *see, e.g., [16].*

In particular, for $p = 1$ we get a piecewise linear function interpolating $(0, v_1), (t_1, v_2), \ldots, (t_k, v_{\eta-1}), (1, v_\eta)$. On the other hand, for $p = 3$ we get a cubic B-spline.

3 Proposed Idempotization Method

The proposed idempotization and weighted arithmetic mean fitting task seeks an OMA operator-like [14] function:

$$\mathsf{F}_{\boldsymbol{\varphi},\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i \varphi_i(x_i),$$

that minimizes the total squared differences between $\mathbf{Y}$ and $\mathsf{F}_{\boldsymbol{\varphi},\mathbf{w}}(\mathbf{X})$. Here, $\mathbf{w}$ stands for a weighting vector and $\varphi_1, \ldots, \varphi_n : [0,1] \to [0,1]$ are some strictly increasing continuous bijections.

Of course, as there are uncountably many functions that can be used in the model we are looking for, we should restrict the feature space in order to make the task solvable on a computer. In our case, for a fixed p and a knot vector $\mathbf{t}$ of length k, we assume that φ_i – used to normalize the i-th variable, $i = 1, \ldots, n$ – is a nonperiodic B-spline:

$$\varphi_i(x) = \sum_{j=1}^{\eta} c_i N^{\mathbf{t}}_{j-p,p}(x)$$

for some vector of $\eta = p + k + 1$ control points $\mathbf{c}$ ordered increasingly. Note that the condition $0 = c_1 < c_2 < \cdots < c_{\eta-1} < c_\eta = 1$ guarantees that φ_i is strictly increasing, continuous, and onto $[0, 1]$.

Therefore, the feasible set consists of functions:

$$\mathsf{F}_{\mathbf{c},\mathbf{w}}(\mathbf{x}) = \sum_{i=1}^{n} w_i \tilde{x}_i = \sum_{i=1}^{n} w_i \sum_{j=1}^{\eta} c_j^{(i)} N^{\mathbf{t}}_{j-p,p}(x_i),$$

where $w_1, \ldots, w_n \geq 0$, $\sum_{i=1}^{n} w_i = 1$, $c_1^{(i)} = 0$, $c_\eta^{(i)} = 1$, $c_2^{(i)} - c_1^{(i)} > 0$, $\ldots$, $c_\eta^{(i)} - c_{\eta-1}^{(i)} > 0$ for all $i = 1, \ldots, n$. Please observe that $\mathsf{F}_{\mathbf{c},\mathbf{w}}$ is an idempotent and nondecreasing in each variable function of each $\tilde{\mathbf{x}}^{(j)} = (\varphi_1(x_1^{(j)}), \ldots, \varphi_n(x_n^{(j)}))$.

Also, as in the first part of our contribution [1], we would like to prevent overfitting to the training data set, so we should consider some form of the model regularization.

To sum up, for some fixed Tiknohov regularization coefficient $\lambda_w \in \mathbb{R}$, in this paper we are interested in the following optimization task:

$$\text{minimize} \sum_{l=1}^{m} \left(\left(\sum_{i=1}^{n} w_i \sum_{j=1}^{\eta} c_j^{(i)} N^{\mathbf{t}}_{j-p,p}\left(x_i^{(l)}\right) \right) - y^{(l)} \right)^2 + \lambda_w \sum_{i=1}^{n} w_i^2 \quad \text{w.r.t. } \mathbf{w}, \mathbf{c},$$

under the above-mentioned constraints.

As far as computer implementation is concerned, we can rewrite the above equation in terms of a bi-level minimization procedure. The inner-level part, for a fixed $\mathbf{w}$, optimizes for $\mathbf{c}$ and in fact can be written in the form of a standard quadratic programming task (with linear constraints, note that we may pre-compute the values of B-spline basis functions for each element in $\mathbf{X}$ and store them for further reference). The outer-level component, optimizing for $\mathbf{w}$, can be solved via some non-linear solver – in our case we propose to rely on the CMA-ES [12] algorithm and logarithmic barrier functions that enable us to ensure that the constraints on $\mathbf{w}$ are met.

4 Experimental Results

In this section we apply the proposed method on the data set from Example 1, that is, four different similarity measures for each (unordered) pair of R functions in the benchmark data set discussed in [2,3]. The number of unique observations equals to $m = 30628$. The benchmark data set is of the following form:

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**...**
$x_1^{(j)}$	0.82	0.58	0.15	0.37	0.17	0.22	0.69	0.87	...
$x_2^{(j)}$	0.73	0.41	0.25	0.26	0.02	0.13	0.90	0.70	...
$x_3^{(j)}$	0.63	0.84	0.38	0.40	0.11	0.46	0.72	0.83	...
$x_4^{(j)}$	0.92	0.75	0.48	0.39	0.12	0.28	0.80	0.92	...
$y^{(j)}$	1.00	0.75	0.50	0.25	0.00	0.25	0.75	1.00	...

The meaning of the four variables has been explained in Example 1. The output variable, y, is a value in the set $\{0.0, 0.25, 0.5, 0.75, 1.0\}$ and reflects an expert's assessment of a similarity degree originally provided on a linguistic scale, one of "totally dissimilar", "dissimilar", "hard to say", "similar", or "very similar". We can conceive the y variable as a kind of censored data – it would of course be impossible for an expert to provide a precise similarity degree assessment in a form of a real number in the $[0, 1]$ interval. At a design stage, an (ordered) linguistic scale seemed a much more user-friendly choice.

We may observe that, as far as raw data are concerned, there is no weighted arithmetic mean which returns the value of 1.00 for input values like $(0.82, 0.73, 0.63, 0.92)$ or $(0.87, 0.70, 0.83, 0.92)$.

We split the data set into two parts: 80% randomly chosen observations are used for training, while the remaining 20% is left for testing purposes. Let us verify the usefulness of the proposed method in the two following scenarios:

- we treat the fitted function as a regression model which describes the relationship between the explanatory variables and the dependent variable,
- we partition the range of the predicted y into intervals and label the values in y according to which interval they fall; as a result, we obtain a binary classifier.

For simplicity, we assume that the B-splines' knots are equidistant.

In the first scenario, we are simply interested in the sum of squared differences (denoted with $\mathfrak{d}_2^2$) between the predicted and desired ys. For the sake of comparison, we chose a classical linear regression model (to recall, there are no constraints on the form of coefficients in its case).

In the second case, we marked the y values greater or equal to 0.5 as cases of plagiarism (class 1) and the other ones as non-suspicious ones (class 2). In order to be able to use the considered regression models in such a task (i.e., the proposed method as well as linear regression), after finding the optimal fit we also seek for the parameter α that splits the predicted y range into two subintervals in such a way that the two naturally obtained classes maximize the F-measure on the training data set. To recall, the *F-measure* is the harmonic mean of precision and recall. These two classifiers are compared with logistic regression and the random forest algorithm.

The sum-of-squares error and the F-measure are negatively correlated, although the correspondence between them is not a monotone function. There are cases, where increasing $\mathfrak{d}_2^2$ leads to an increase in the F-measure and oppositely.

Firstly, let us study the influence of the B-splines degrees and the number of internal knots used on the model performance. We examined the polynomials with corresponding parameters ranging from $p = 1$ and $k = 1$ up to $p = 5$ and $k = 5$ (25 cases in total). Surprisingly, it turns out that the impact is relatively small. What is more, we observe that higher degree polynomials may also lead to a decreased model performance. The difference between the minimal and the maximal F-measure value is equal to ca. 0.02. For the $\mathfrak{d}_2^2$ error, the model of the lowest quality has been obtained for $p = 1$, $k = 1$. As can be seen in Figure 2, fitted polynomials of higher degrees of course have shapes similar to those of lower degrees.

Moreover, let us consider the effect of using the λ_w regularization coefficient. The F-measure and the $\mathfrak{d}_2^2$ error as a function of λ_w is depicted in Figure 3 ($p = 3$, $k = 1$ was used for F-measure and $p = 1$, $k = 4$ for $\mathfrak{d}_2^2$ error). The highest value of the F-measure was obtained for $p = 3$, $k = 1$, $\lambda_w = 33$, while the smallest $\mathfrak{d}_2^2$ error – for $p = 1$, $k = 4$, $\lambda_w = 30$.

Table 1 summarizes the performance measures of the four considered algorithms. The proposed method gives a higher F-measure than linear and logistic regression as well as a lower $\mathfrak{d}_2^2$ error than linear regression. Even though we get a lower F-measure than in the random forest case, please note that our method comes with important algebraic properties of the resulting aggregation function (such as idempotence and nondecreasingness in each variable), as well as a nice model interpretability. The random forest algorithm does not posses such advantages.

Finally, let us study the impact of introducing idempotization to a weighted arithmetic mean-based model, by comparing the performance of the model on the raw data set and on the version transformed with optimal B-splines. Table 2 summarizes the model quality measures for a fixed λ_w equal to 0. We observe that relying on a single feature does not lead to particularly good performance. The same happens if the weighting vector is optimized for but the idempotization scheme is not applied at all. Therefore, there is a positive impact of both factors. Similar observations can be done for other λ_w coefficients.

5 Conclusion

We have discussed a supervised learning method for weights of weighted arithmetic means in cases where data come from an ordinal scale and they have to be properly mapped to the $[0, 1]$ interval prior to computing an optimal fit.

The introduced method has been applied on a real-world data set consisting of data on similarity degrees of pairs of R functions. It has many advantages:

– determining the φ_i functions enables us to normalize the input values so that they become mutually comparable and easily interpretable to the plagiarism detection system's users; other machine learning methods can be applied on the transformed sample too;
– the fitted weighted arithmetic mean serves as a regression model for our data and explains the relationship between the individual similarity degrees and the aggregated similarity assessment; as the weights are by definition nonnegative and they sum up to one, we have a clear intuition of which of the four methods has the highest impact;
– the fitted model fulfills two important properties: it is idempotent and nondecreasing in each variable; thus, its behavior is much more natural and understandable to end-users;
– the obtained regression model can be easily transformed in such a way that it is suitable for using in, e.g., binary classification tasks.

Let us note that the introduced method can be easily extended to the case of fitting arbitrary weighted quasi-arithmetic means, with or without known generator functions.

For future work, inspecting different regularization schemes could lead to models of increased quality. In particular, one may think of introducing a regularization component for the vector of control points, e.g., for some $\lambda_c \in \mathbb{R}$, of the form $\lambda_c \sum_{i=1}^{n} \sum_{j=2}^{\eta} \left(c_j^{(i)} - c_{j-1}^{(i)} \right)^2$.

Acknowledgments. This study was supported by the National Science Center, Poland, research project 2014/13/D/HS4/01700.

References

1. Bartoszuk, M., Beliakov, G., Gagolewski, M., James, S.: Fitting aggregation functions to data: Part I – Linearization and regularization. In: Proc. IPMU'16. Springer (2016), in press

2. Bartoszuk, M., Gagolewski, M.: A fuzzy R code similarity detection algorithm. In: Laurent, A., et al. (eds.) Information Processing and Management of Uncertainty in Knowledge-Based Systems, Part III. vol. 444, pp. 21–30. Springer (2014)
3. Bartoszuk, M., Gagolewski, M.: Detecting similarity of R functions via a fusion of multiple heuristic methods. In: Alonso, J., Bustince, H., Reformat, M. (eds.) Proc. IFSA/Eusflat 2015. pp. 419–426. Atlantis Press (2015)
4. Beliakov, G.: Monotone approximation of aggregation operators using least squares splines. International Journal of Uncertainty, Fuzziness and Knowledge-based Systems 10, 659–676 (2002)
5. Beliakov, G.: How to build aggregation operators from data. International Journal of Intelligent Systems 18, 903–923 (2003)
6. Beliakov, G.: Learning weights in the generalized OWA operators. Fuzzy Optimization and Decision Making 4, 119–130 (2005)
7. Beliakov, G.: Construction of aggregation functions from data using linear programming. Fuzzy Sets and Systems 160, 65–75 (2009)
8. Beliakov, G., Bustince, H., Calvo, T.: A Practical Guide to Averaging Functions. Springer (2016)
9. Beliakov, G., Pradera, A., Calvo, T.: Aggregation functions: A guide for practitioners. Springer-Verlag (2007)
10. Beliakov, G., Warren, J.: Appropriate choice of aggregation operators in fuzzy decision support systems. IEEE Transactions on fuzzy systems 9(6), 773–784 (2001)
11. Gagolewski, M.: Data Fusion: Theory, Methods, and Applications. Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland (2015)
12. Hansen, N.: The CMA evolution strategy: A comparing review. In: Lozano, J., Larranga, P., Inza, I., Bengoetxea, E. (eds.) Towards a new evolutionary computation. Advances in estimation of distribution algorithms. pp. 75–102. Springer (2006)
13. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag (2013)
14. Mesiar, R., Mesiarová-Zemánková, A.: The ordered modular averages. IEEE Transactions on Fuzzy Systems 19(1), 42–50 (2011)
15. R Development Core Team: R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria (2016), http://www.R-project.org
16. Schumaker, L.: Spline Functions: Basic Theory. Cambridge University Press (2007)

Table 1. Performance of the fitted models (accuracy, precision, recall, F-measures, squared L_2 error). The proposed method is based on $\lambda_w = 33$, $w_1 = 0.35$, $w_2 = 0.15$, $w_3 = 0.15$, $w_4 = 0.35$, $p = 3$, $k = 1$ for optimizing F-measure (a) and $\lambda_w = 30$, $w_1 = 0.30$, $w_2 = 0.16$, $w_3 = 0.15$, $w_4 = 0.39$, $p = 1$, $k = 4$ for optimizing $\mathfrak{d}_2^2$ (b).

method	**accuracy**	**precision**	**recall**	F	$\mathfrak{d}_2^2$
Proposed method (a)	0.997	0.921	0.933	0.927	106.62
Proposed method (b)	0.997	0.900	0.920	0.910	95.85
Linear regression	0.995	0.810	0.969	0.883	103.53
Logistic regression	0.997	0.885	0.960	0.921	—
Random forest	0.998	0.927	0.956	0.941	—

Table 2. Performance measures as functions of different weighting vectors; $p = 3$, $k = 1$, $\lambda_w = 0$, with and without idempotization.

w_1	w_2	w_3	w_4	**accuracy**	**precision**	**recall**	F	$\mathfrak{d}_2^2$	**idempot.**
1	0	0	0	0.992	0.848	0.693	0.763	186.30	Yes
0	1	0	0	0.995	0.927	0.787	0.851	208.74	Yes
0	0	1	0	0.994	0.803	0.853	0.828	316.04	Yes
0	0	0	1	0.996	0.904	0.840	0.871	136.67	Yes
0.27	0.06	0.38	0.29	0.996	0.952	0.800	0.870	137.34	No
0.41	0.12	0.07	0.40	0.997	0.919	0.907	0.913	107.69	Yes

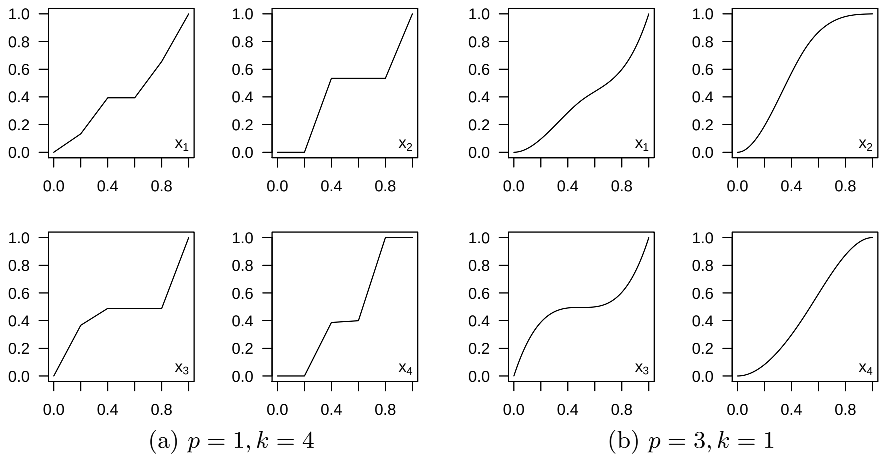


Fig. 2. Best B-splines of different degrees fit to the training sample.

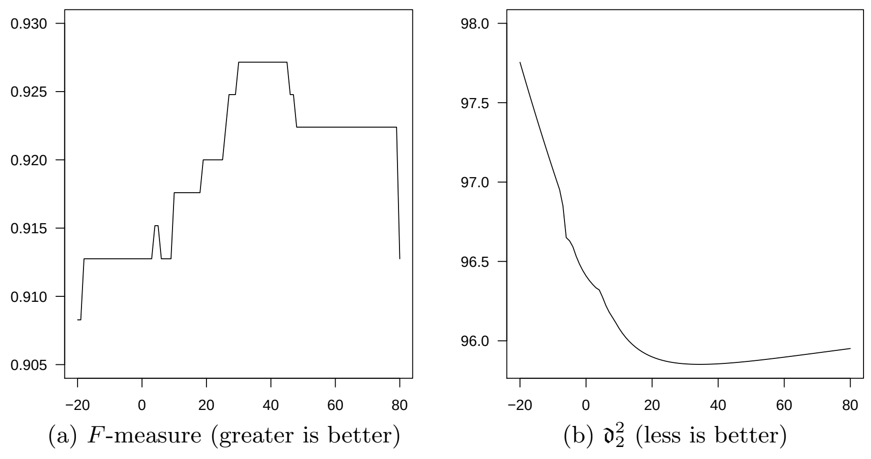


Fig. 3. F-measure and squared error as a function of the λ_w regularization coefficient.