Noname manuscript No.
(will be inserted by the editor)

Supplementary Material for : Nonparametric Regression and Classification with Functional, Categorical, and Mixed Covariates

the date of receipt and acceptance should be inserted later

Further Numerical Experiments

In this supplementary material we present additional simulation results for the regression and the classification case. In the classification case we compare our results with a random forest. For further comparison of our method with other procedures see Section "Numerical experiments" in the main paper.

1 Regression Problems

1.1 Set-up

To further investigate the finite sample performance of our procedure we generate data according to two additional yet simpler models, namely

(CatR) For $i = 1, \dots, n$, the observations are generated as
$Y_i = 2(X_{i1} + \ldots + X_{iq}) + \varepsilon_i$, with iid $X_{i1}, \dots, X_{ip} \sim\mathcal{B}(0.5)$, $q \leq p$.

(FunR) For $i = 1, \dots, n$ and $j = 1, \dots, p$, the functional observations $X_{ij}(t)$, $t \in [0, T]$, are generated according to

$$\tilde{X}_{ij}(t) = \sum_{l=1}^{5} \left(B_{ij,l} \sin\left(\frac{t}{T}(5 - B_{ij,l})2\pi \right) - M_{ij,l} \right),$$

where $B_{ij,l} \sim \mathcal{U}[0, 5]$ and $M_{ij,l} \sim \mathcal{U}[0, 2\pi]$ for $l = 1, \dots, 5$, $j = 1, \dots, p$, $i = 1, \dots, n$, and $T = 300$. $\mathcal{U}$ stands for the (continuous) uniform distribution. Then, $X_{ij}(t)$ is calculated from $\tilde{X}_{ij}(t)$ by scaling it in direction i and then dividing each value by 10. The regression is built by the functional linear model

$$Y_i = 5 \sum_{j=1}^{q} \int X_{ij}(t) \gamma_{3,\frac{1}{3}}(t/10) dt + \varepsilon_i, \quad q \leq p,$$

where the coefficient function $\gamma_{a,b}(t) = b^a/\Gamma(a)t^{a-1}e^{-bt}I\{t > 0\}$ is the density of the Gamma distribution. See Ramsay and Silverman (2005) Chapter 15 or Kokoszka and Reimherr (2017) Chapter 4 for an introduction to functional linear models. Furthermore, we assume that each X_{ij} is observed on a dense, equidistant grid of 300 evaluation points.

The errors ε_i are iid standard normal in all models.

In both scenarios we investigate 'minimal' and 'sparse' cases. Specifically, for CatR and FunR, we compare the cases $q = 1$, $p = 2$ (minimal: CatR.m and FunR.m, respectively) and $q = 3$, $p = 18$ (sparse: (*.s)),. For all generated data sets we use a one-sided Picard kernel $K(u) = e^{-u}I\{u \geq 0\}$ and the results shown are based on 500 replications each.

The weights are determined in the same way as described in the main paper for model MixR.

1.2 Results

The minimizing weights for our six different scenarios (including MixR.m and MixR.s) and sample size $n = 100, 500, 1000$ are shown in Figure 1. We present all results for MixR here although some of them are already displayed in the main paper for the sake of completeness. To increase comparability between the different models we display normed weights $\frac{\hat{\omega}_j}{\sum_{k=1}^{p} \hat{\omega}_k}$. This can also be interpreted as separating the estimation of the weights (normed weights) and optimization of the bandwidth ($h_{\text{opt}} = \frac{h_n^{\text{fun/cat}}}{\sum_{k=1}^{p} \hat{\omega}_k}$). It can be seen that the selection of relevant predictors works well, as the covariates with influence on the response get distinctly higher weights than those without in all scenarios. The sum over the weights for relevant covariates should be approximately one whereas the weights for irrelevant covariates should be close to zero. Both is visible for the simulated data.

As explained in the main paper we also compute the minimizer of Q under the restrictions

(i) $\omega_1 = \omega_2 = \ldots = \omega_p$,
(ii) $\omega_j = 0$ for all covariates with no influence on the response.

In Figure 2 the squared estimation error of $\hat{f}$ is shown, where we display the average over 100 (minimal case) and 10000 (sparse case) $\mathbf{x}$-values, respectively. In scenario CatR with $p = 2$, we only use the 4 possible $\mathbf{x}$-values. In all other scenarios, the $\mathbf{x}$-values are generated randomly in the same way as the covariates in the respective scenario. In each of the 500 replications, new $\mathbf{x}$-values are generated. In all cases the results for our procedure are comparable to those under restriction (ii) and better than those under restriction (i), as expected. To get an insight in the influence of the $\mathbf{x}$-values on the estimation error we ran the simulations also with $\mathbf{x}$-values that are the same for each replication. Like for scenario MixR in the main paper the results are almost identical to those with varying $\mathbf{x}$-values shown in Figure 2. Only the variance of the estimation errors is slightly larger with varying $\mathbf{x}$-values (as could be expected).

2 Classification Problems

2.1 Set-up

Similar to the regression case, we generate data according to the models

(CatC) For $i = 1, \ldots, n$, the observations are generated with iid errors $\varepsilon_i \sim \mathcal{U}[0, 1]$ as

$$Y_i \begin{cases} = X_{i1} + \ldots, X_{iq} + 1 & \text{if } \varepsilon_i \leq 0.7 \\ \sim \mathcal{U}\{1, \ldots, G\} & \text{else,} \end{cases}$$

with $X_{i1}, \ldots, X_{ip}$ $\sim B(0.5)$, $q \leq p$; $\mathcal{U}$ stands for the (discrete) uniform distribution. We run the simulations with $G = 5$.

(FunC) The functional observations are based on those built in model FunR, see Section 1. Let's call them $X_{ij}^{(\mathrm{Fun})}$. Then the functional observations for this classification model are $X_{ij}(t) = X_{ij}^{(\mathrm{Fun})}(t) + c \cdot C_{ij}$ for some constant $c > 0$ with $C_{ij} \sim \mathcal{U}\{0, 1, 2\}$, and the outcome is $Y_i = C_{i1} + \ldots + C_{iq} + 1$, $q \leq p$. Thus we have $G = 2q + 1$ response classes in this scenario. We simulate this setup for different values of $c \in \{0.1, 0.3, 0.7\}$. In Figure 3 examples for the functional observations with different c are shown to highlight the effect of the size of c. It can be seen that for $c = 0.7$ and $q = 1$ classes are distinctly separated, and the classification task could even be done manually/visually. In what follows, we will hence focus on $c = 0.3$.

As before we compare minimal (*.m) and sparse (*.s) cases in all scenarios, i. e., $q = 1$, $p = 2$ (*.m) and $q = 3$, $p = 18$ (*.s) for CatC and FunC, and $q_{\mathrm{fun}} = q_{\mathrm{cat}} = 1$, $p_{\mathrm{fun}} = p_{\mathrm{cat}} = 2$ (*.m) . The results are based on 500 replications. We use again the one-sided Picard kernel as described in Section 1. In contrast to the regression case, however, we use a pre-estimator for the weights instead of a starting value for the bandwidth, as explained in the main paper.

2.2 Results

In Figure 4 the minimizing normed weights for model CatC, FunC (with $c = 0.3$) and MixC for $n = 100, 500, 1000$ are displayed. Again we repeat some results for scenario MixC for the sake of completeness.The performance regarding the variable selection is very encouraging. The prediction performance of our procedure is shown in Figure 5, where we display the squared error of $\hat{P}_g$ and compare it to the results under restriction (i) and (ii) as described in Section 1.2. Additionally we compare the results to those of a random forest, as a benchmark apart from kernel-based, nonparametric prediction. After applying a functional principal component analysis (R package

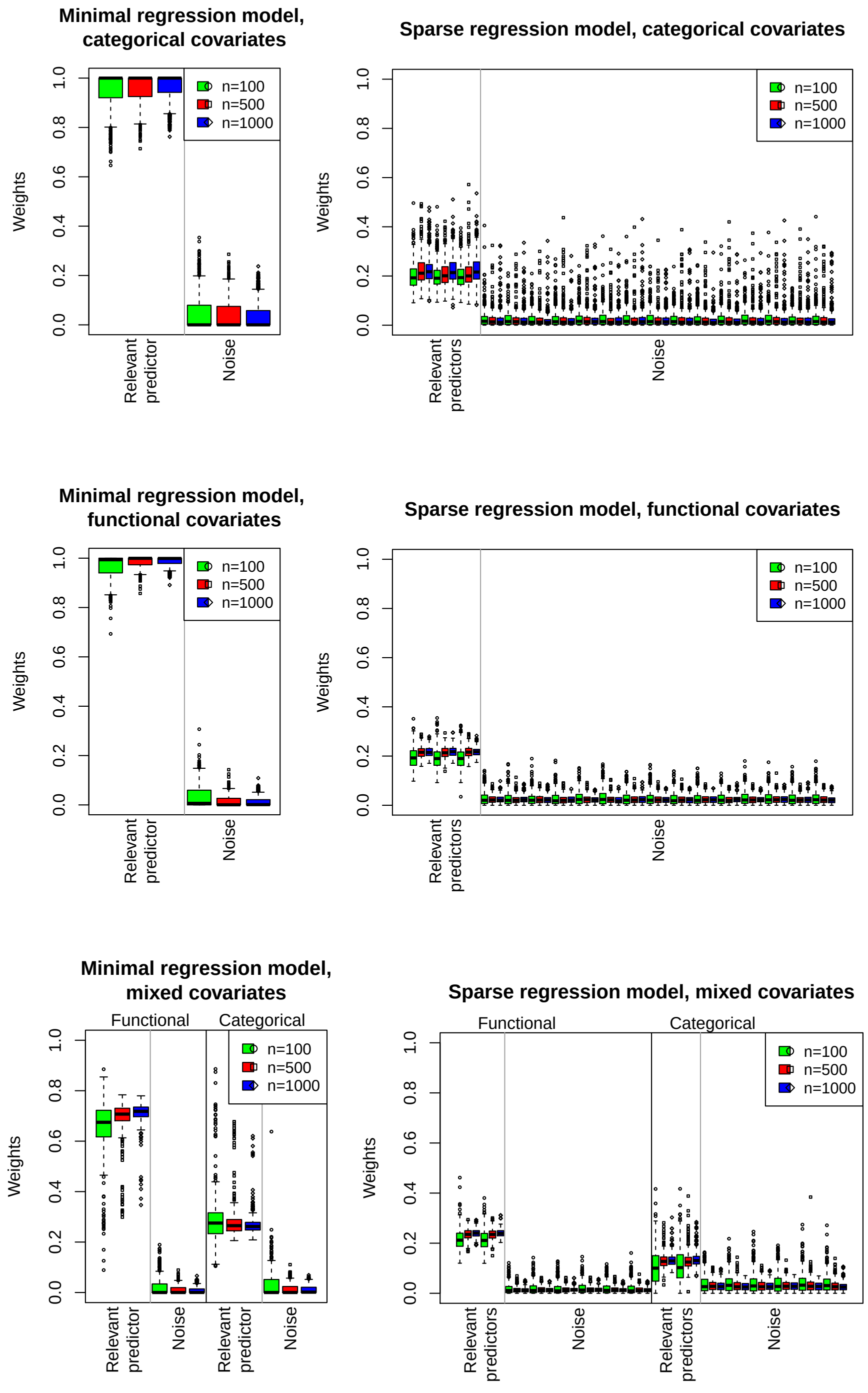

Fig. 1 Normed minimizing weights $\frac{\hat{\omega}_j}{\sum_{k=1}^{p} \hat{\omega}_k}$ for models CatR (top), FunR (middle) and MixR (bottom) in the minimal and sparse case, respectively.

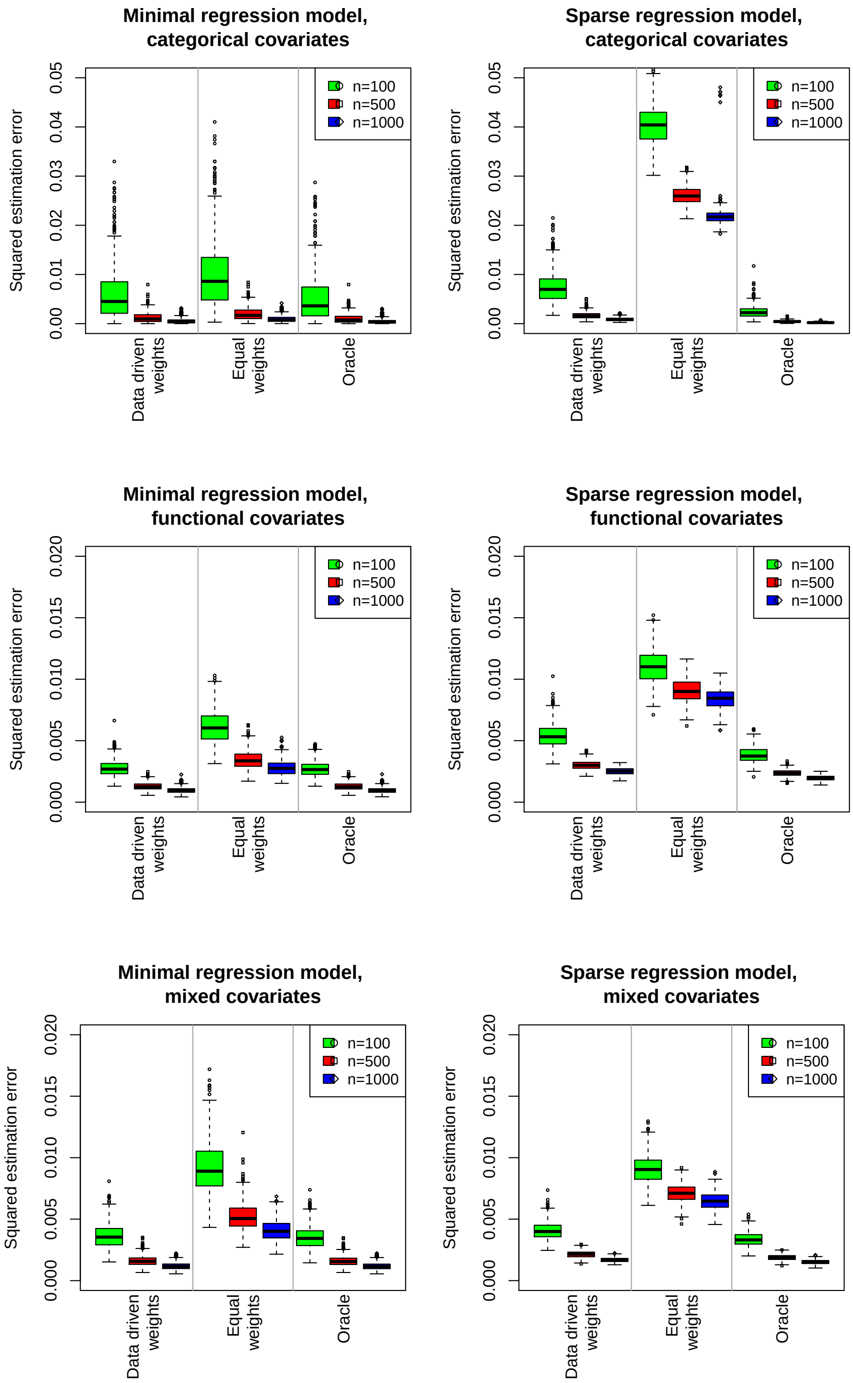

Fig. 2 Prediction results for models CatR (top), FunR (middle) and MixR (bottom) in the minimal (left) and sparse (right) case with no restriction ('data driven weights'), restriction (i, 'equal weights') and (ii, 'oracle'), respectively.

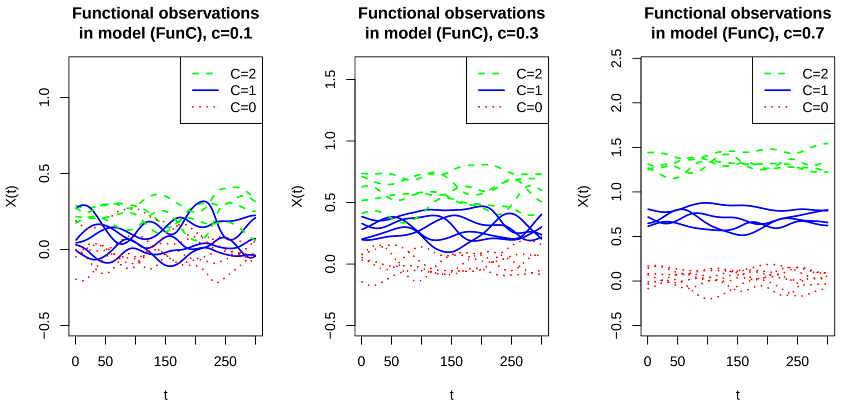

Fig. 3 Examples for functional observations from model FunC for $c = 0.1, 0.3, 0.7$ (from left to right). The green dashed lines represent X_{ij} with $C_{ij} = 2$, the blue solid lines those with $C_{ij} = 1$ and the red dotted lines are realizations of X_{ij} with $C_{ij} = 0$.

refund by Goldsmith et al. (2021)) on the functional observations we build a random forest using the R function *randomForest* (Liaw and Wiener (2002)), as also explained in the main paper. For new $\mathbf{x}$-values that are generated in the same way as the observations from the training set, we predict the posterior probability with the random forest and with our $\hat{P}_g$ with the estimated weights, respectively. The data for the boxplots is calculated on test sets with $N = 100$ (minimal case) and $N = 1000$ (sparse case) as the Brier Score In scenario CatC the response is calculated as $y(\mathbf{x}) = x_1 + \ldots + x_q + 1$ and in FunC and MixC $y(\mathbf{x})$ are built in the same way as for the training observations. Similar to the regression case, the results achieved with new $\mathbf{x}$-values for each replication and those with the same $\mathbf{x}$-values in all replications are comparable. We display the results with varying $\mathbf{x}$-values. It can be seen that the prediction works well and in almost all cases clearly better than the random forest. Further in the sparse cases, the results with data driven weights are much better than those with equal weights, which confirms the good variable selection/weighting performance.

As additional information we display the missclassification rate. The results are summed up in Table 1. They confirm and extend the good performance shown in Figure 5, especially that our procedure works much better than the random forest in most of the settings considered. For model FunC with different values of c, we see that classification becomes much easier with growing c as expected.

To gain some further insight into the performance of our procedure in the classification of functional data, we simulate another model with a purely nonparametric concept of classification, that is

(FunC.2) The functional observations are generated in exactly the same way as in model FunR. The classification is then based on the maximizing argument

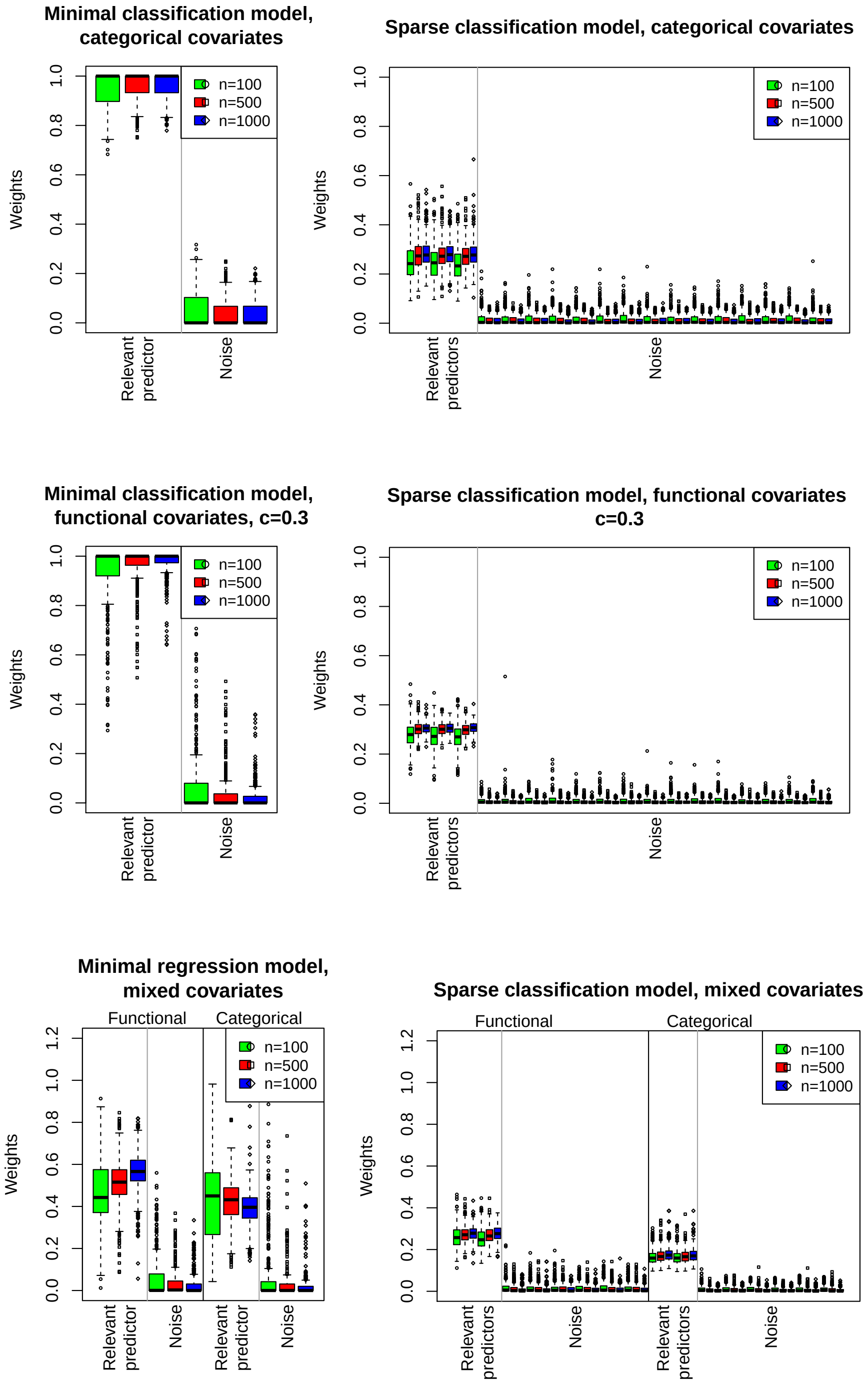

Fig. 4 Normed minimizing weights $\frac{\hat{\omega}_j}{\sum_{k=1}^{p} \hat{\omega}_k}$ for models CatC (top), FunC (middle) with $c = 0.3$ and MixC (bottom) in the minimal (left) and sparse (right) case, respectively.

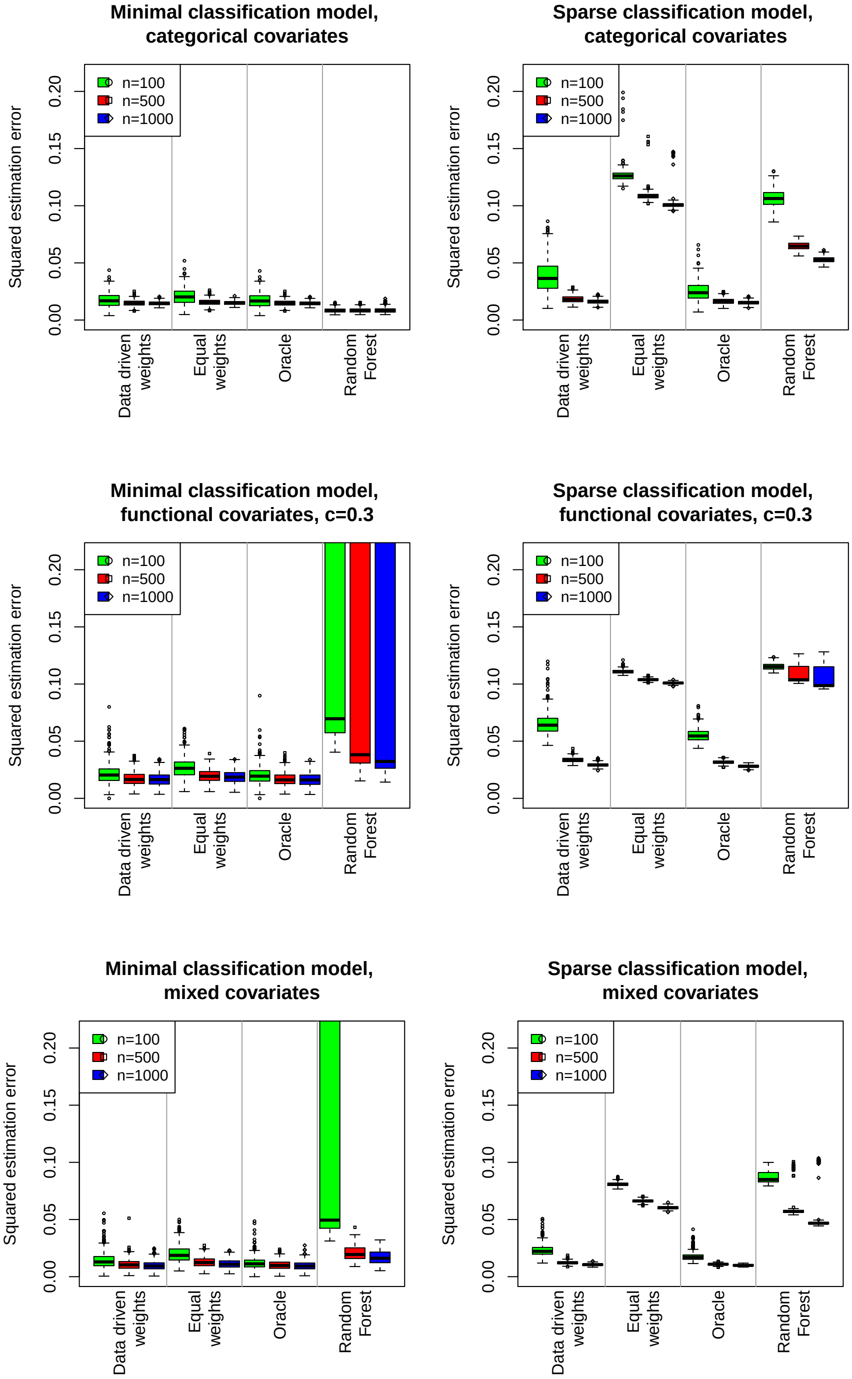

Fig. 5 Prediction results for models CatC (top), FunC (middle) with $c = 0.3$, and MixC (bottom) in the minimal (left) and sparse (right) case with no restriction (‘data driven weights’), restriction (i, ‘equal weights’) and (ii, ‘oracle’), and with a random forest, respectively.

Model	n	Data driven w.	Equal weights	Oracle	Random forest
(CatC.m)	100	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
	500	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
	1000	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
(CatC.s)	100	0.09 (0.07)	0.50 (0.03)	0.00 (0.02)	0.34 (0.06)
	500	0.00 (0.00)	0.36 (0.02)	0.00 (0.00)	0.05 (0.02)
	1000	0.00 (0.00)	0.30 (0.02)	0.00 (0.00)	0.02 (0.01)
(FunC.m)	100	0.38 (0.05)	0.40 (0.05)	0.38 (0.05)	0.50 (0.18)
	500	0.36 (0.04)	0.37 (0.04)	0.35 (0.04)	0.44 (0.17)
$c = 0.1$	1000	0.35 (0.04)	0.36 (0.05)	0.35 (0.05)	0.42 (0.16)
(FunC.s)	100	0.73 (0.02)	0.74 (0.02)	0.69 (0.02)	0.75 (0.02)
	500	0.73 (0.02)	0.74 (0.02)	0.69 (0.02)	0.71 (0.02)
$c = 0.1$	1000	0.65 (0.02)	0.70 (0.02)	0.63 (0.02)	0.69 (0.02)
(FunC.m)	100	0.04 (0.02)	0.05 (0.02)	0.04 (0.02)	0.28 (0.31)
	500	0.03 (0.02)	0.04 (0.02)	0.03 (0.02)	0.23 (0.30)
$c = 0.3$	1000	0.03 (0.02)	0.04 (0.02)	0.03 (0.02)	0.21 (0.28)
(FunC.s)	100	0.31 (0.05)	0.69 (0.02)	0.27 (0.03)	0.74 (0.03)
	500	0.31 (0.05)	0.69 (0.02)	0.27 (0.03)	0.65 (0.09)
$c = 0.3$	1000	0.14 (0.01)	0.60 (0.02)	0.13 (0.01)	0.58 (0.12)
(FunC.m)	100	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.29 (0.33)
	500	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.26 (0.33)
$c = 0.7$	1000	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.25 (0.32)
(FunC.s)	100	0.08 (0.14)	0.68 (0.04)	0.03 (0.03)	0.75 (0.05)
	500	0.00 (0.00)	0.78 (0.18)	0.00 (0.00)	0.67 (0.17)
$c = 0.7$	1000	0.00 (0.00)	0.81 (0.19)	0.00 (0.00)	0.55 (0.24)
(FunC.2.m)	100	0.46 (0.05)	0.57 (0.06)	0.46 (0.05)	0.78 (0.12)
	500	0.34 (0.05)	0.47 (0.05)	0.34 (0.05)	0.77 (0.14)
	1000	0.30 (0.04)	0.43 (0.05)	0.30 (0.04)	0.76 (0.15)
(FunC.2.s)	100	0.74 (0.03)	0.77 (0.02)	0.70 (0.02)	0.78 (0.03)
	500	0.62 (0.02)	0.72 (0.02)	0.60 (0.02)	0.75 (0.03)
	1000	0.58 (0.02)	0.71 (0.02)	0.56 (0.02)	0.74 (0.05)
(MixC.m)	100	0.04 (0.02)	0.06 (0.07)	0.03 (0.02)	0.32 (0.43)
	500	0.03 (0.02)	0.03 (0.03)	0.03 (0.02)	0.20 (0.37)
	1000	0.03 (0.01)	0.03 (0.02)	0.03 (0.01)	0.18 (0.35)
(MixC.s)	100	0.13 (0.03)	0.59 (0.02)	0.10 (0.02)	0.63 (0.11)
	500	0.07 (0.01)	0.44 (0.02)	0.06 (0.01)	0.21 (0.25)
	1000	0.06 (0.01)	0.39 (0.02)	0.06 (0.01)	0.13 (0.24)

Table 1 Missclassification rates as arithmetic mean (and standard deviation) with no restriction ('Data driven weights'), restriction (i) ('Equal weights'), restriction (ii) ('Oracle') and with a random forest respectively. The values in teal are the lowest and the values in violet the second to lowest in each row.

of each functional observation following the set-up in Fuchs et al. (2015). Let $j_{i,\max}$ be the index such that
$\max_t X_{ij_{i,\max}}(t) = \max(\max_t X_{i1}(t), \dots, \max_t X_{iq}(t))$, $q \leq p$.
Then $Y_i = g \in \{1, \dots, G\}$ if and only if $\arg\max_t X_{ij_{i,\max}}(t) \in (\frac{gT-T}{G}, \frac{gT}{G}]$.

The results for this model are displayed in Figure 6 with $G = 5$ and in Table 1. It can be seen that the prediction for this model is more difficult than for model FunC while the variable selection still works quite well. In comparison to the random forest, however, our procedure is still highly competitive.

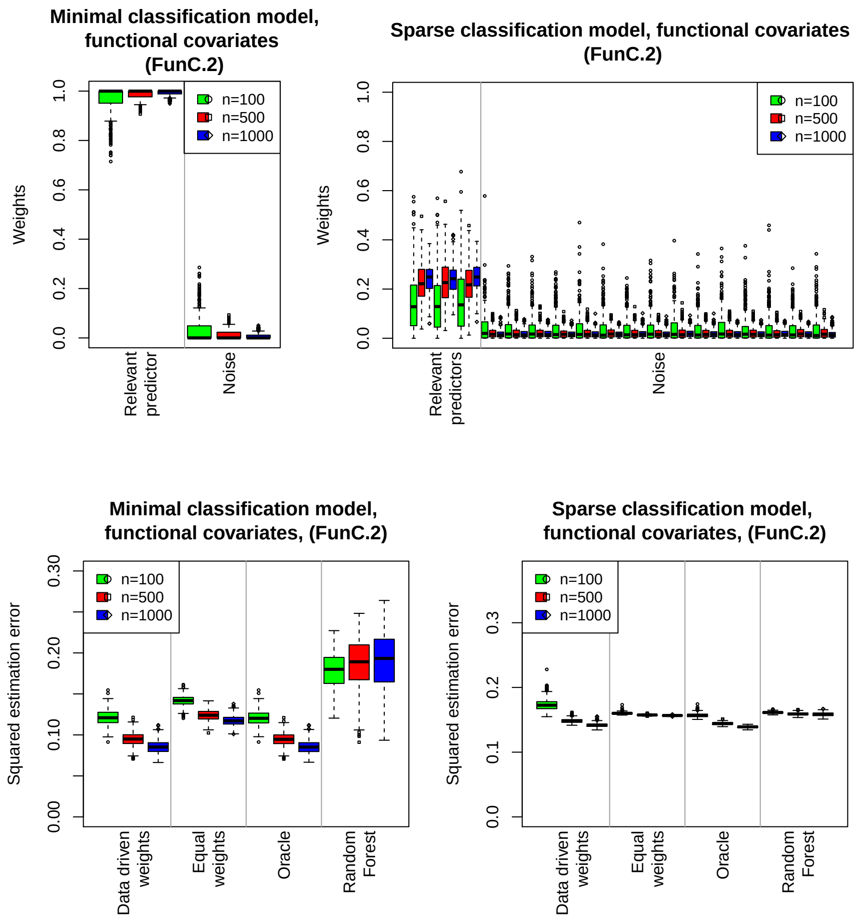

Fig. 6 Normed minimizing weights (top) and prediction resulte (bottom) for model (FunC.2) in the minimal (left) and sparse (right) case, respectively.

References

Fuchs K, Gertheiss J, Tutz G (2015) Nearest neighbor ensembles for functional data with interpretable feature selection. Chemometrics and Intelligent Laboratory Systems 146:186–197

Goldsmith J, Scheipl F, Huang L, Wrobel J, Di C, Gellar J, Harezlak J, McLean MW, Swihart B, Xiao L, Crainiceanu C, Reiss PT (2021) refund: Regression with Functional Data. URL https://CRAN.R-project.org/package=refund, r package version 0.1-24

Kokoszka P, Reimherr M (2017) Introduction to Functional Data Analysis. Texts in Statistical Science, CRC Press

Liaw A, Wiener M (2002) Classification and regression by randomforest. R News 2(3):18–22, URL https://CRAN.R-project.org/doc/Rnews/

Ramsay J, Silverman B (2005) Functional Data Analysis. Springer Series in Statistics, Springer New York