COBRA : une stratégie d'agrégation non linéaire

Benjamin Guedj

LSTA, UPMC & LTCI, Telecom ParisTech

http://www.lsta.upmc.fr/doct/guedj/

En collaboration avec Gérard Biau, Aurélie Fischer et James D. Malley

Keywords


Prediction
Analysis
Trees
Density
Markov-Chain
PAC-Bayesian
Vapnik-Cervonenkis
Splines
Posterior
Algorithm
MCMC
Learning
Sparsity
Unsupervised
Parameters
Models
Neighbors
Oracle
Dimension
Kernel
Bayesian
Forests
Regression
AIC
Lasso
Nonasymptotic
Machine
Monte-Carlo
BIC
High-dimensional
Linear
Network
Aggregation
Classification
Supervised
Nonparametric
Estimation
Inference
k-means
ABC
Group-Lasso
Shrinkage
Nonlinear
Logistic
Selection
Clustering
Random
Data
Additive
Gradient

Outline

The prediction problem

- The present talk focuses on the prediction problem in a regression setting.

The prediction problem

- The present talk focuses on the prediction problem in a regression setting.

- On the basis of a training sample $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ of i.i.d. replications of a r.v. $(\mathbf{X}, Y) \in \mathbb{R}^d \times \mathbb{R}$, how could we learn the relationship between Y and $\mathbf{X}$?

  (*e.g.*, estimation of the regression function $\mathbf{x} \mapsto \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$)

The prediction problem

- The present talk focuses on the prediction problem in a regression setting.

- On the basis of a training sample $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ of i.i.d. replications of a r.v. $(\mathbf{X}, Y) \in \mathbb{R}^d \times \mathbb{R}$, how could we learn the relationship between Y and $\mathbf{X}$?

  (*e.g.*, estimation of the regression function $\mathbf{x} \mapsto \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$)

- Countless methods: Least squares regression, Lasso, ridge regression, Elastic net, nearest neighbors, neural networks, (PAC-)bayesian methods, random trees and forests, ...

The prediction problem

- The present talk focuses on the prediction problem in a regression setting.

- On the basis of a training sample $\mathcal{D}_n = \{(\mathbf{X}_i, Y_i)\}_{i=1}^n$ of i.i.d. replications of a r.v. $(\mathbf{X}, Y) \in \mathbb{R}^d \times \mathbb{R}$, how could we learn the relationship between Y and $\mathbf{X}$?

  (*e.g.*, estimation of the regression function $\mathbf{x} \mapsto \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$)

- Countless methods: Least squares regression, Lasso, ridge regression, Elastic net, nearest neighbors, neural networks, (PAC-)bayesian methods, random trees and forests, ...

- How should we decide which method to use?

The aggregation approach

- Aggregation approach: From a known dictionary $\mathbb{D} = \{\phi_1, \phi_2, \ldots, \phi_M\}$, we consider estimators of the form $f_\theta = \theta^\top \mathbb{D} = \sum_{k=1}^{M} \theta_k \phi_k$ where:

The aggregation approach

- Aggregation approach: From a known dictionary $\mathbb{D} = \{\phi_1, \phi_2, \ldots, \phi_M\}$, we consider estimators of the form $f_\theta = \theta^\top \mathbb{D} = \sum_{k=1}^M \theta_k \phi_k$ where:
  - $\theta \in \{e_1, \ldots, e_M\}$ (selectors),

The aggregation approach

- Aggregation approach: From a known dictionary $\mathbb{D} = \{\phi_1, \phi_2, \ldots, \phi_M\}$, we consider estimators of the form $f_\theta = \theta^\top \mathbb{D} = \sum_{k=1}^M \theta_k \phi_k$ where:
  - $\theta \in \{e_1, \ldots, e_M\}$ (selectors),
  - $\theta \in \Lambda^M = \{\lambda \in \mathbb{R}_+^M : \sum_{k=1}^M \lambda_k = 1\}$ (convex aggregation),

The aggregation approach

- Aggregation approach: From a known dictionary $\mathbb{D} = \{\phi_1, \phi_2, \ldots, \phi_M\}$, we consider estimators of the form $f_\theta = \theta^\top \mathbb{D} = \sum_{k=1}^M \theta_k \phi_k$ where:
  - $\theta \in \{e_1, \ldots, e_M\}$ (selectors),
  - $\theta \in \Lambda^M = \{\lambda \in \mathbb{R}_+^M : \sum_{k=1}^M \lambda_k = 1\}$ (convex aggregation),
  - $\theta \in \mathbb{R}^M$ (linear aggregation),

The aggregation approach

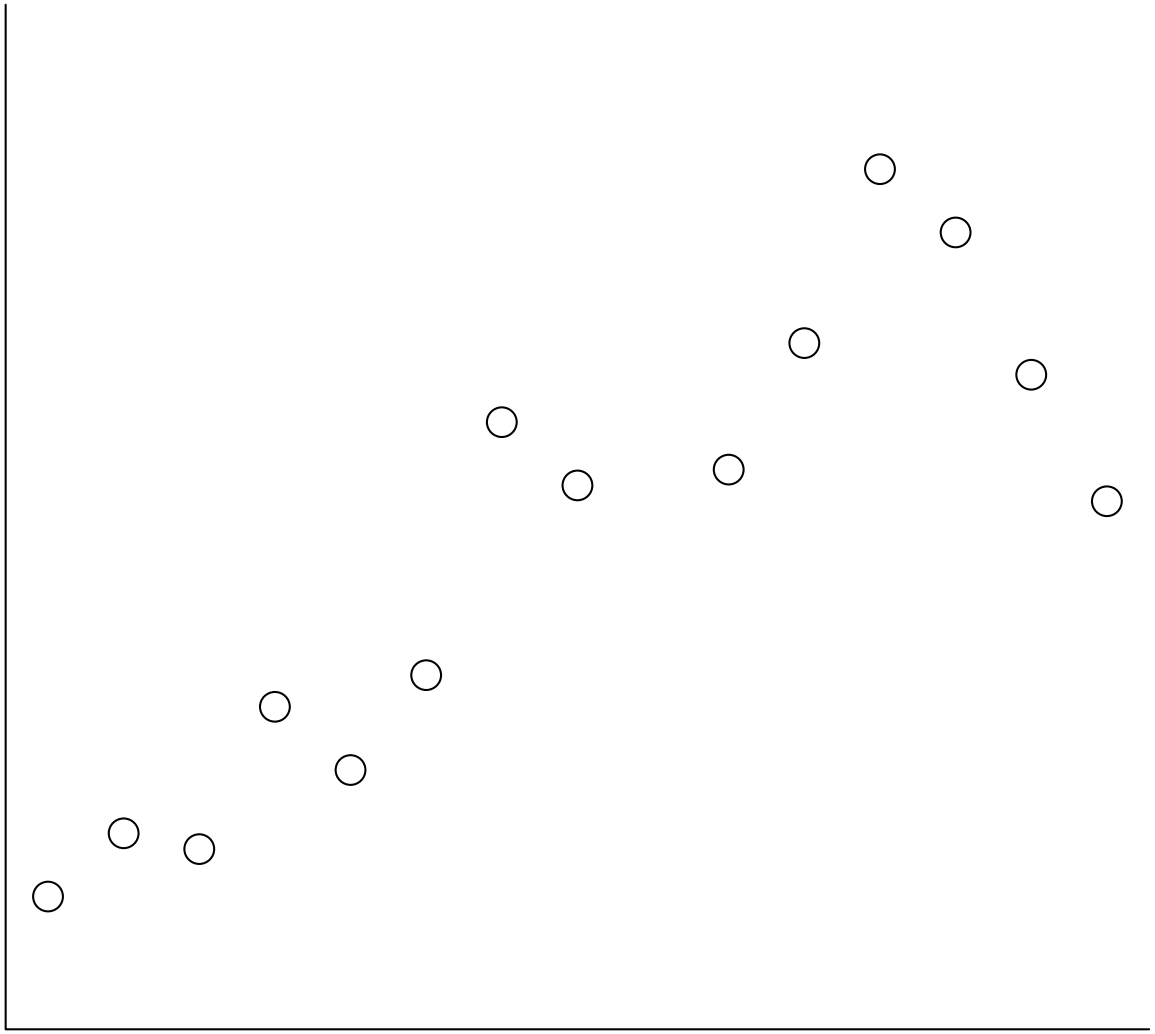
- Aggregation approach: From a known dictionary $\mathbb{D} = \{\phi_1, \phi_2, \ldots, \phi_M\}$, we consider estimators of the form $f_\theta = \theta^\top \mathbb{D} = \sum_{k=1}^M \theta_k \phi_k$ where:
  - $\theta \in \{e_1, \ldots, e_M\}$ (selectors),
  - $\theta \in \Lambda^M = \{\lambda \in \mathbb{R}_+^M : \sum_{k=1}^M \lambda_k = 1\}$ (convex aggregation),
  - $\theta \in \mathbb{R}^M$ (linear aggregation),
  - ...

The aggregation approach

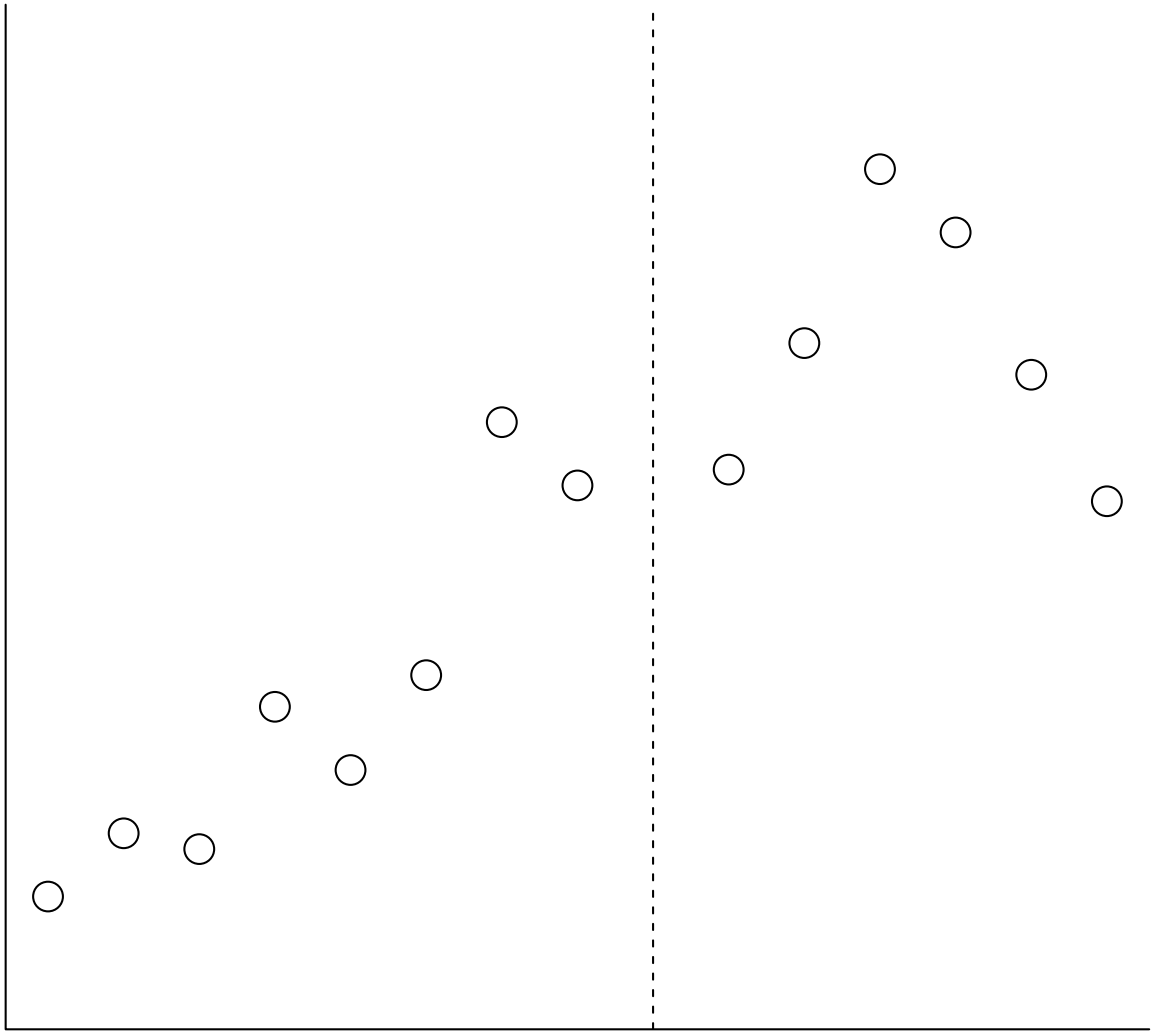
- Aggregation approach: From a known dictionary $\mathbb{D} = \{\phi_1, \phi_2, \ldots, \phi_M\}$, we consider estimators of the form $f_\theta = \theta^\top \mathbb{D} = \sum_{k=1}^M \theta_k \phi_k$ where:
  - $\theta \in \{e_1, \ldots, e_M\}$ (selectors),
  - $\theta \in \Lambda^M = \{\lambda \in \mathbb{R}_+^M : \sum_{k=1}^M \lambda_k = 1\}$ (convex aggregation),
  - $\theta \in \mathbb{R}^M$ (linear aggregation),
  - ...
- The present talk investigates a fairly different point of view: We propose an original nonlinear method for combining preliminary predictors. Inspiration: Mojirsheibani (1999).

Outline

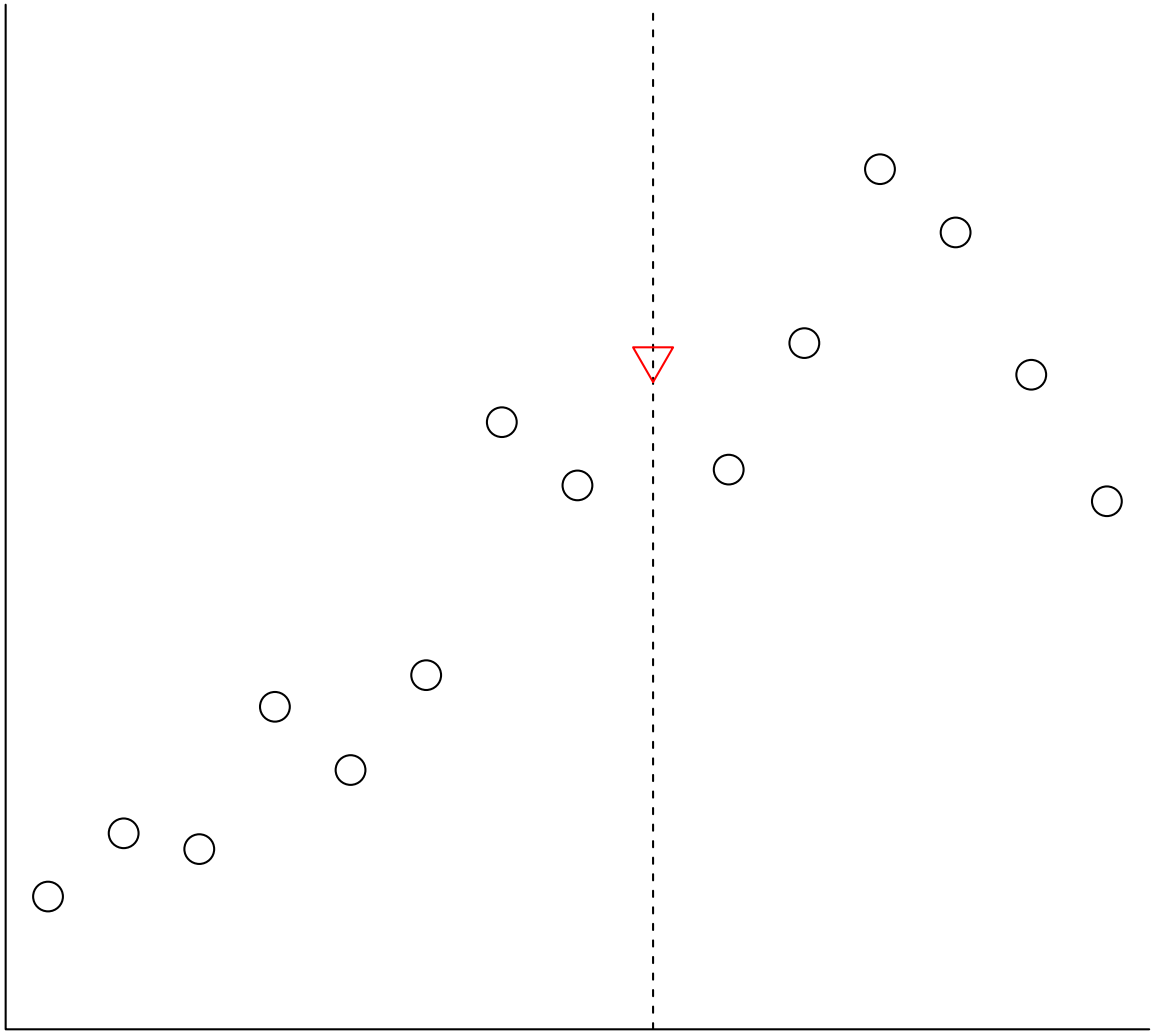
Example



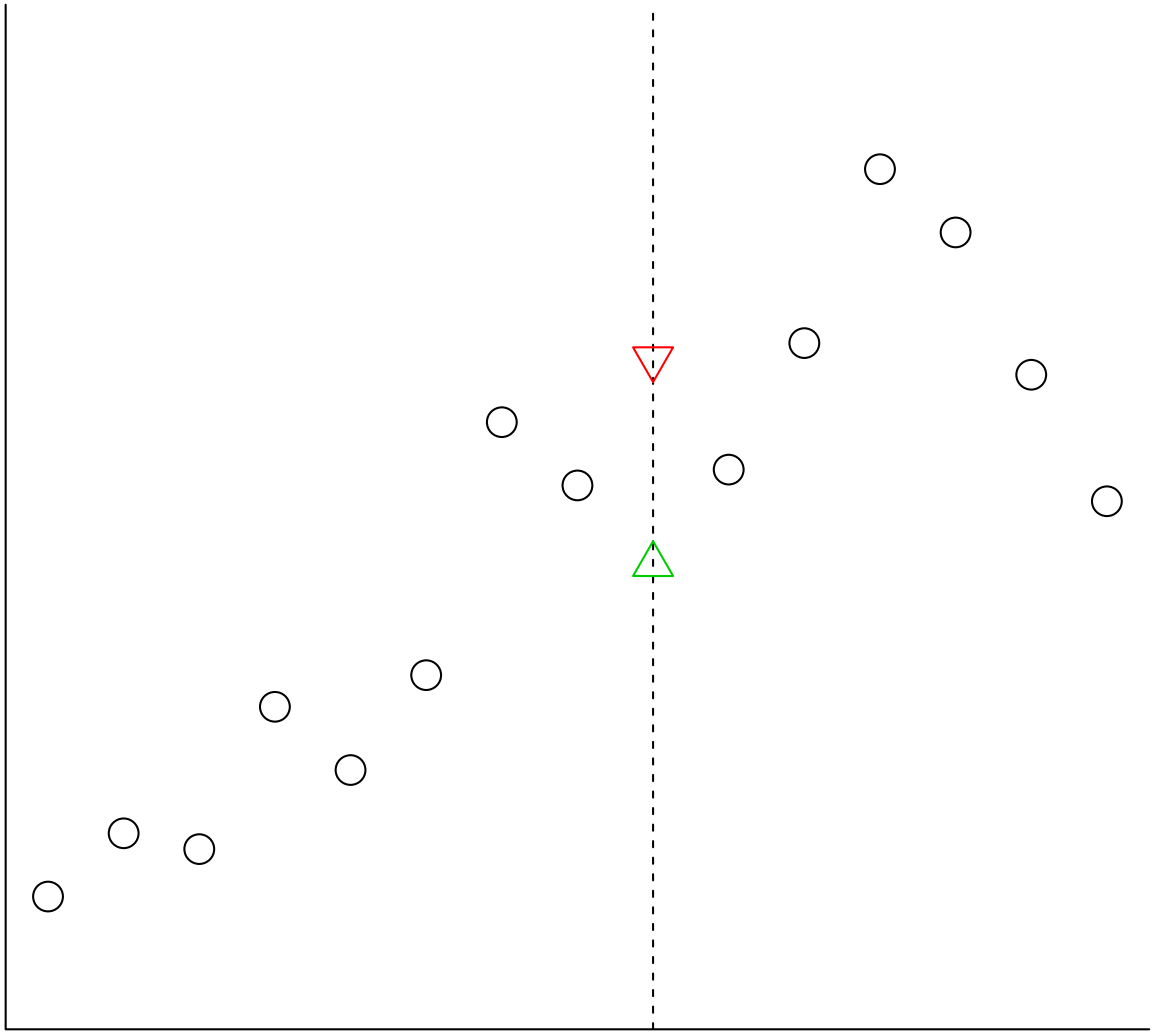
Example



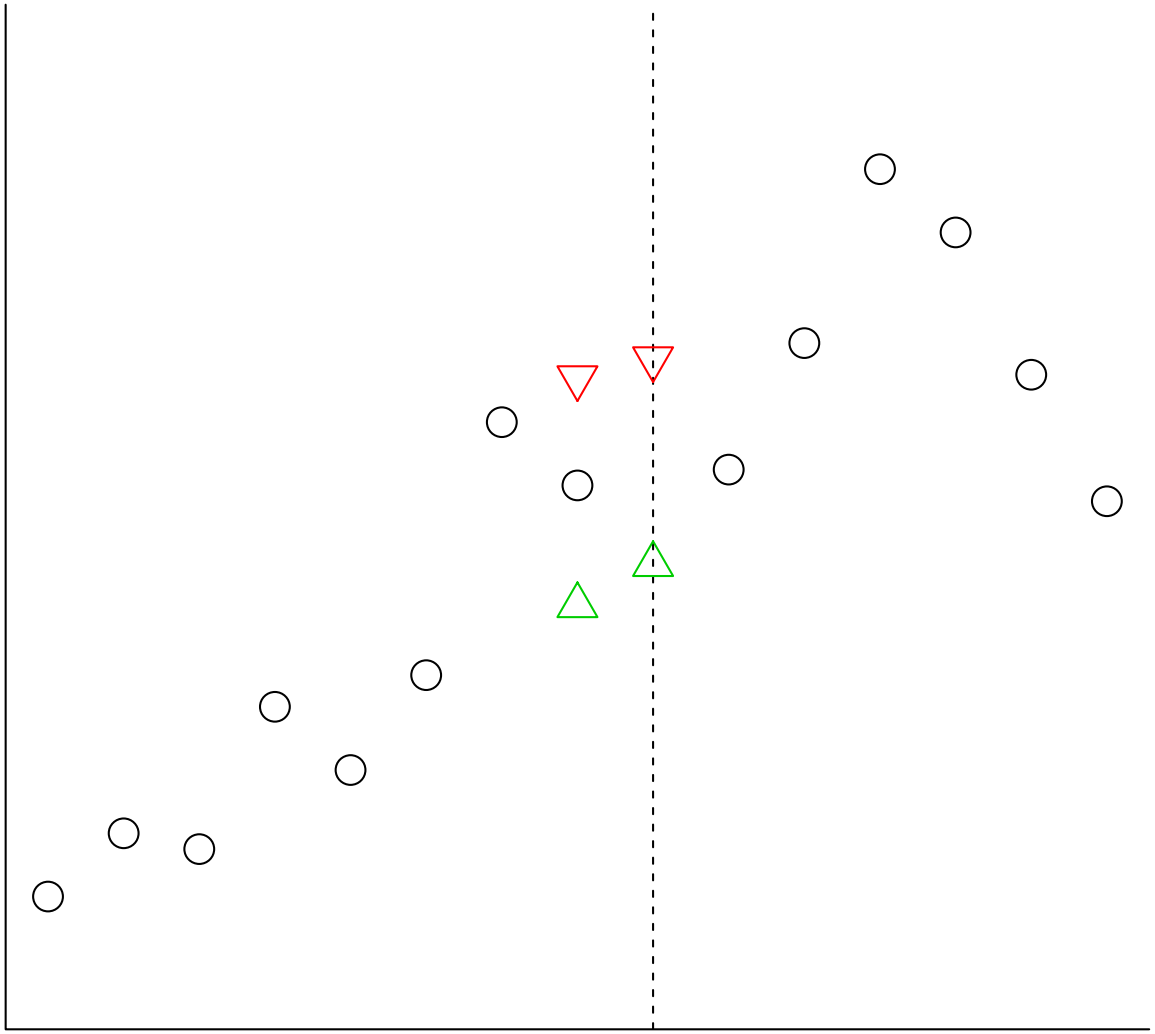
Example



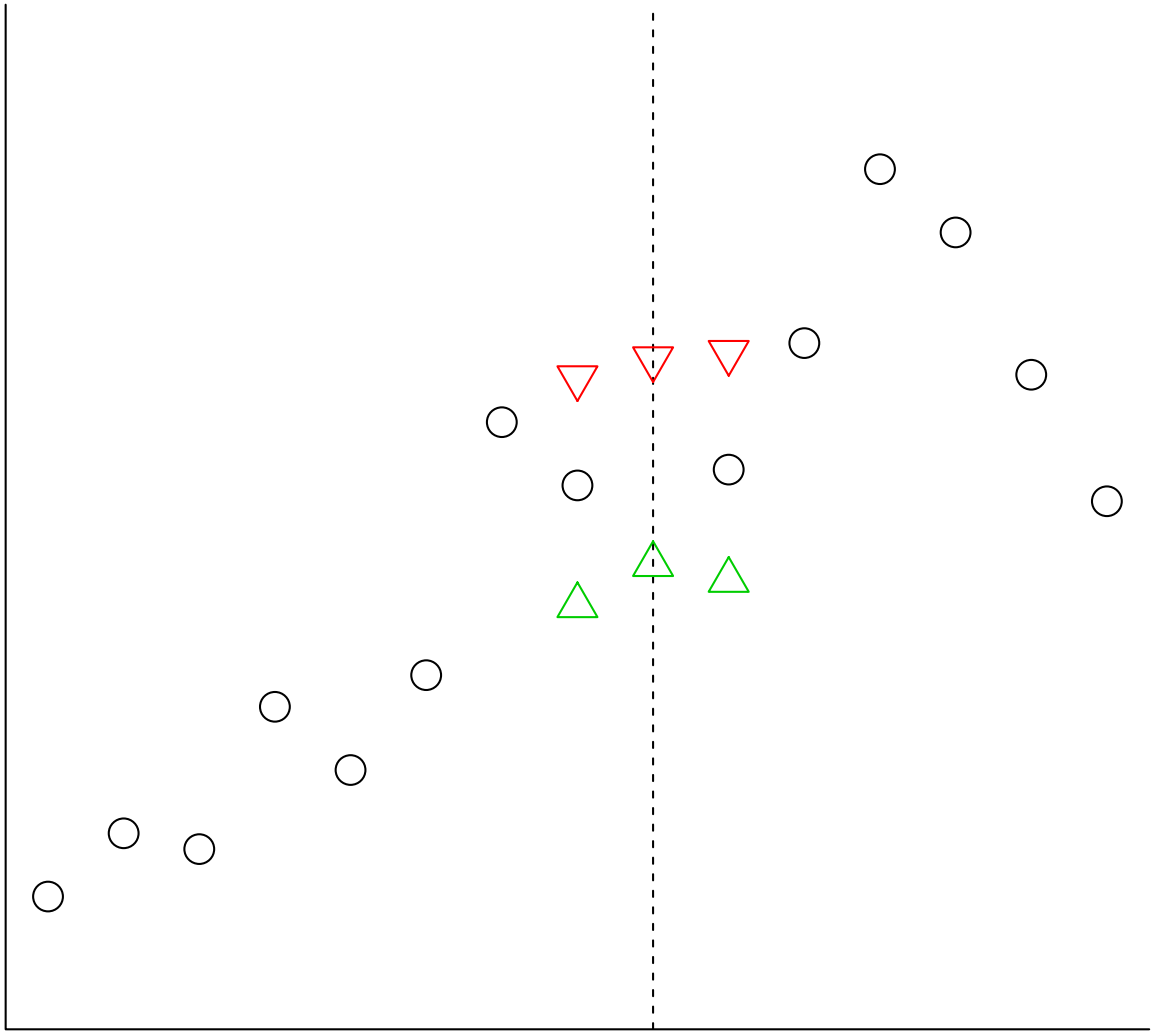
Example



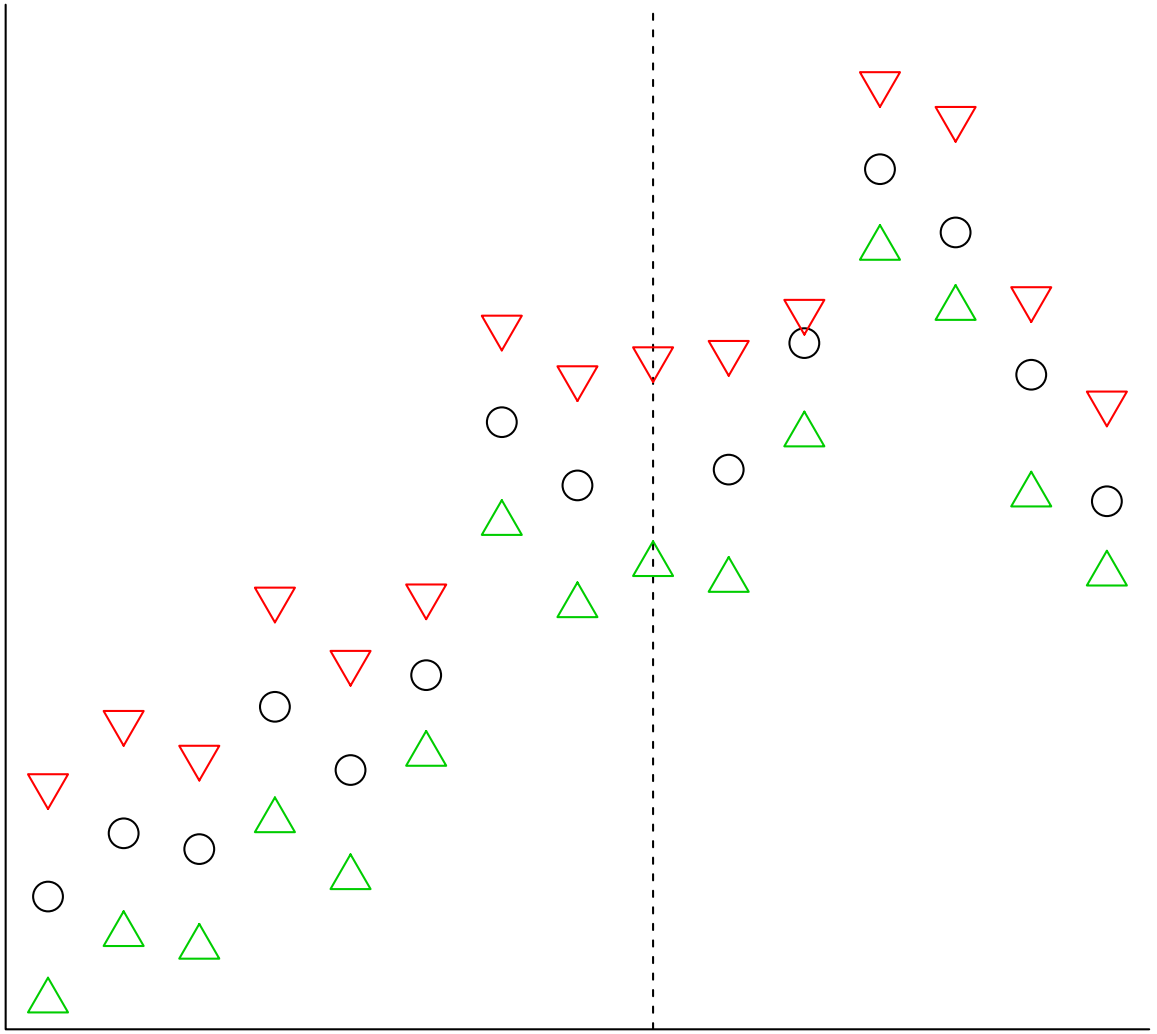
Example



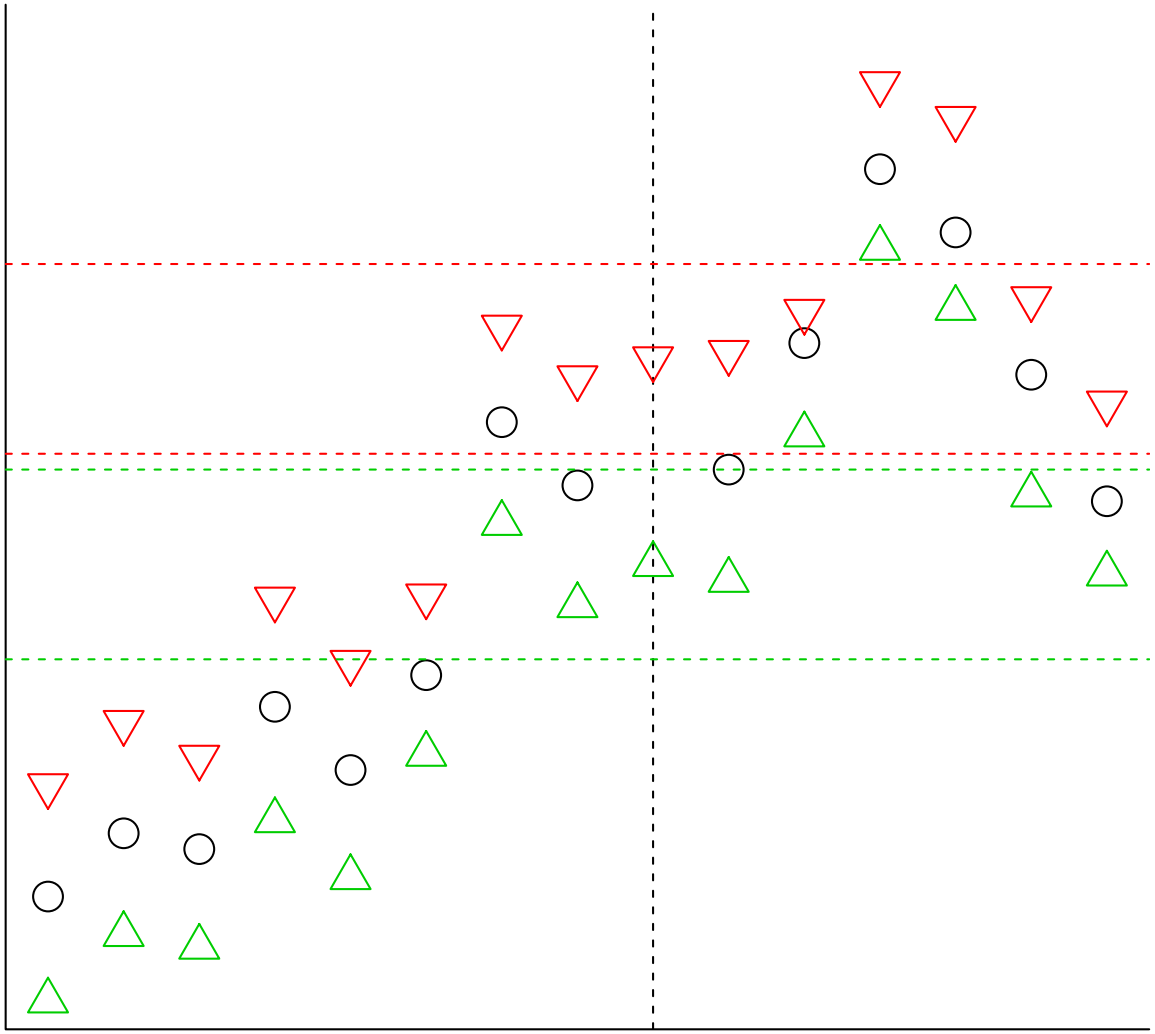
Example



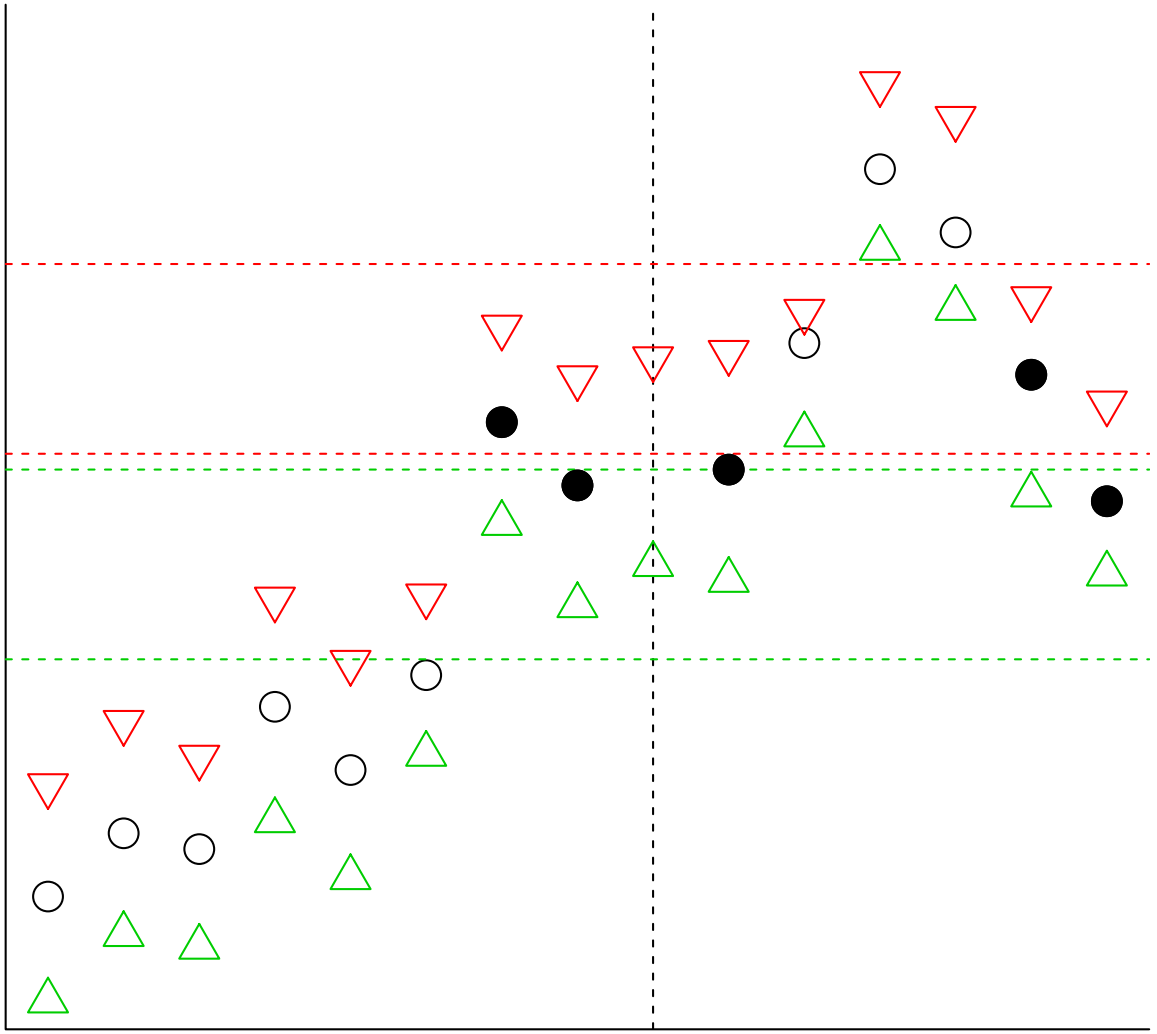
Example



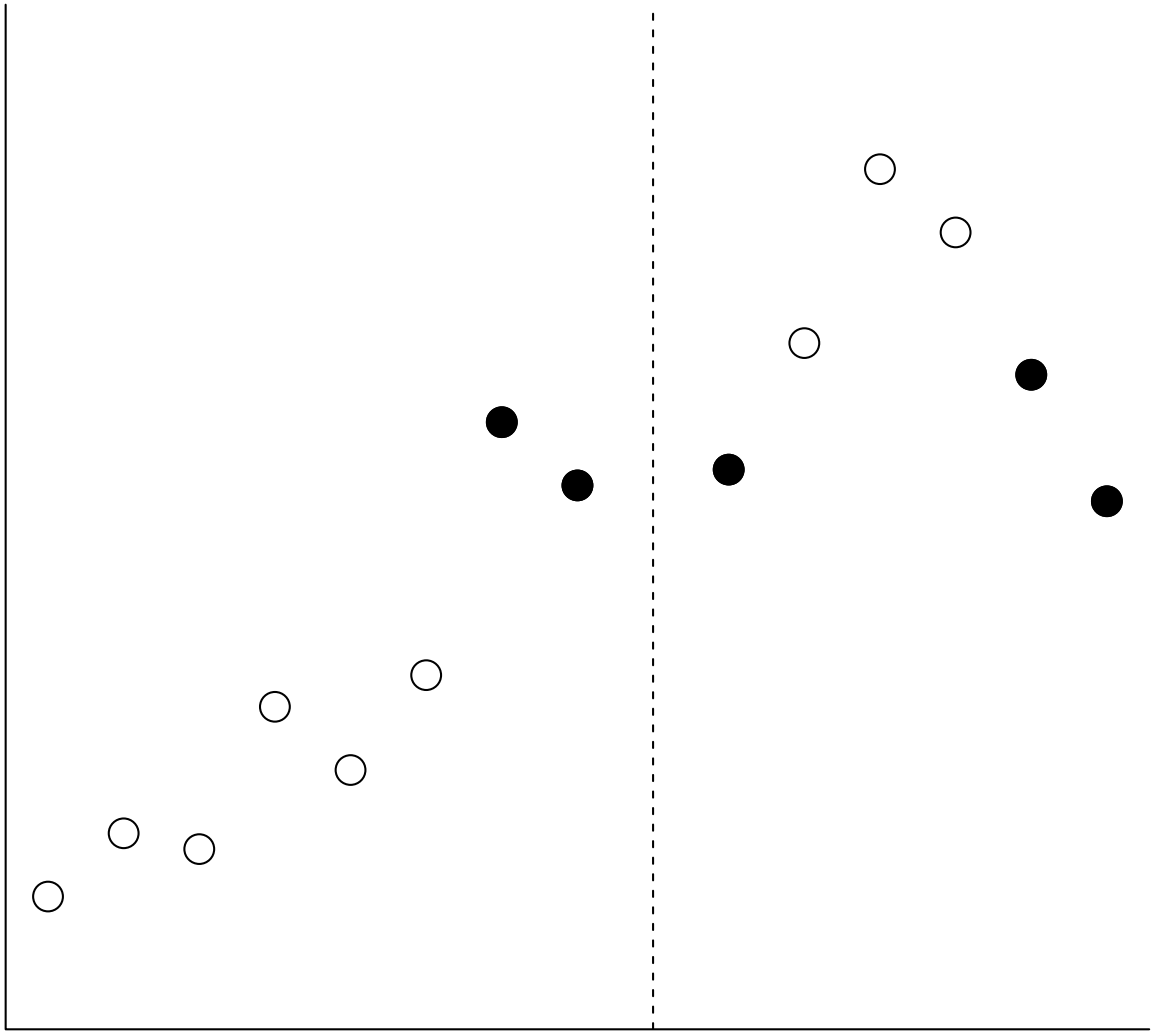
Example



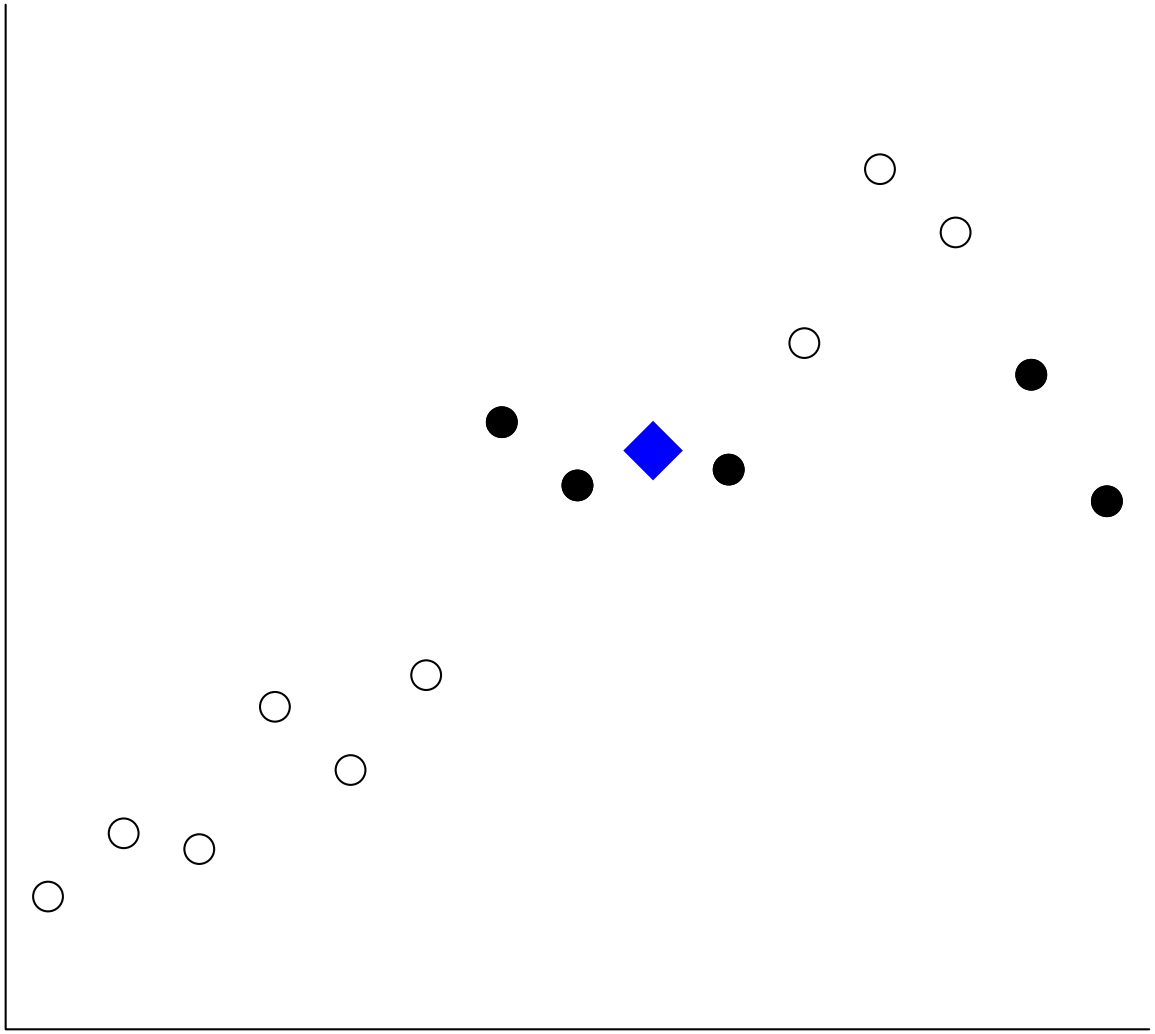
Example



Example



Example



Notation

- Training sample: $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$, i.i.d. replications of $(\mathbf{X}, Y) \in \mathbb{R}^d \times \mathbb{R}$.

- Goal: Estimate the regression function $r^\star : \mathbf{x} \mapsto \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$.

Notation

- Training sample: $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$, i.i.d. replications of $(\mathbf{X}, Y) \in \mathbb{R}^d \times \mathbb{R}$.

- Goal: Estimate the regression function $r^\star : \mathbf{x} \mapsto \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$.

- Splitting:

$$\mathcal{D}_n = \mathcal{D}_k \cup \mathcal{D}_\ell.$$

Notation

- Training sample: $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$, i.i.d. replications of $(\mathbf{X}, Y) \in \mathbb{R}^d \times \mathbb{R}$.
- Goal: Estimate the regression function $r^\star : \mathbf{x} \mapsto \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$.
- Splitting:
$$\mathcal{D}_n = \mathcal{D}_k \cup \mathcal{D}_\ell.$$
- M basic {$\emptyset$,semi,non}parametric machines: $\mathbf{r}_k = (r_{k,1}, \ldots, r_{k,M})$.
- Sole requirement: They must deliver an estimation of $r^\star$ on the basis of $\mathcal{D}_k$ only.

The regression collective

- For any query point $\mathbf{x} \in \mathbb{R}^d$, define

$$T_n\left(\mathbf{r}_k(\mathbf{x})\right) = \sum_{i=1}^{\ell} W_{n,i}(\mathbf{x}) Y_i,$$

The regression collective

- For any query point $\mathbf{x} \in \mathbb{R}^d$, define

$$T_n\left(\mathbf{r}_k(\mathbf{x})\right) = \sum_{i=1}^{\ell} W_{n,i}(\mathbf{x}) Y_i,$$

where

$$W_{n,i}(\mathbf{x}) = \frac{\mathbf{1}_{\bigcap_{m=1}^{M}\{|r_{k,m}(\mathbf{x}) - r_{k,m}(\mathbf{X}_i)| \leq \varepsilon_\ell\}}}{\sum_{j=1}^{\ell} \mathbf{1}_{\bigcap_{m=1}^{M}\{|r_{k,m}(\mathbf{x}) - r_{k,m}(\mathbf{X}_j)| \leq \varepsilon_\ell\}}}.$$

- Key idea: Closeness is measured by the basic machines, used as a metric indicator over the data.

Theoretical performance

- The performance of T_n is assessed through its quadratic risk:

$$\mathbb{E}\left|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})\right|^2.$$

- The oracle is

$$T(\mathbf{r}_k(\mathbf{X})) = \mathbb{E}[Y|\mathbf{r}_k(\mathbf{X})].$$

- Technical regularity assumption: For any $m = 1, \ldots, M$,

$$r_{k,m}^{-1}((t,+\infty)) \underset{t\uparrow+\infty}{\searrow} \emptyset \quad \text{and} \quad r_{k,m}^{-1}((-\infty,t)) \underset{t\downarrow-\infty}{\searrow} \emptyset.$$

A nonasymptotic result

Theorem

For all distributions of $(\mathbf{X}, Y)$ *with* $\mathbb{E}Y^2 < \infty$,

$$\mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 \leq \inf_f \ \mathbb{E}|f(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 + \mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - T(\mathbf{r}_k(\mathbf{X}))|^2.$$

A nonasymptotic result

Theorem

For all distributions of $(\mathbf{X}, Y)$ *with* $\mathbb{E}Y^2 < \infty$,

$$\mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 \leq \inf_f \ \mathbb{E}|f(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 + \mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - T(\mathbf{r}_k(\mathbf{X}))|^2.$$

- Selectors:

$$\mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 \leq \min_{m=1,\dots,M} \ \mathbb{E}|r_{k,m}(\mathbf{X}) - r^\star(\mathbf{X})|^2 + \mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - T(\mathbf{r}_k(\mathbf{X}))|^2.$$

A nonasymptotic result

Theorem

For all distributions of $(\mathbf{X}, Y)$ *with* $\mathbb{E}Y^2 < \infty$,

$$\mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 \leq \inf_f \ \mathbb{E}|f(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 + \mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - T(\mathbf{r}_k(\mathbf{X}))|^2.$$

- Selectors:

$$\mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 \leq \min_{m=1,\dots,M} \mathbb{E}|r_{k,m}(\mathbf{X}) - r^\star(\mathbf{X})|^2 + \mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - T(\mathbf{r}_k(\mathbf{X}))|^2.$$

- Linear and convex combinations:

$$\mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})|^2 \leq \inf_{\theta \in \mathbb{R}^M} \mathbb{E}\left|\sum_{j=1}^M \theta_j r_{k,j}(\mathbf{X}) - r^\star(\mathbf{X})\right|^2 + \mathbb{E}|T_n(\mathbf{r}_k(\mathbf{X})) - T(\mathbf{r}_k(\mathbf{X}))|^2.$$

Asymptotic result

Proposition

Assume that $\varepsilon_\ell \to 0$ *and* $\ell \varepsilon_\ell^M \to \infty$ *as* $\ell \to \infty$. *Then*

$$\mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - T\left(\mathbf{r}_k(\mathbf{X})\right)\right|^2 \to 0 \quad \text{as } \ell \to \infty,$$

Asymptotic result

Proposition

Assume that $\varepsilon_\ell \to 0$ *and* $\ell \varepsilon_\ell^M \to \infty$ *as* $\ell \to \infty$. *Then*

$$\mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - T\left(\mathbf{r}_k(\mathbf{X})\right)\right|^2 \to 0 \quad \text{as } \ell \to \infty,$$

Corollary

$$\limsup_{\ell \to \infty} \mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - r^\star(\mathbf{X})\right|^2 \leq \inf_f \ \mathbb{E}\left|f(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})\right|^2 .$$

Main theoretical result

Theorem

Assume that Y and the basic machines $\mathbf{r}_k$ are bounded by some constant R. Assume moreover that there exists a constant $L \geq 0$ such that, for any $k \geq 1$,

$$|T(\mathbf{r}_k(\mathbf{x})) - T(\mathbf{r}_k(\mathbf{y}))| \leq L|\mathbf{r}_k(\mathbf{x}) - \mathbf{r}_k(\mathbf{y})|, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

Then, with the choice $\varepsilon_\ell \propto \ell^{-\frac{1}{M+2}}$, one has

$$\mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - r^\star(\mathbf{X})\right|^2 \leq \inf_f \ \mathbb{E}\left|f(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})\right|^2 + C\ell^{-\frac{2}{M+2}},$$

for some positive constant $C = C(R, L)$, independent of k.

Main theoretical result

Theorem

Assume that Y and the basic machines $\mathbf{r}_k$ are bounded by some constant R. Assume moreover that there exists a constant $L \geq 0$ such that, for any $k \geq 1$,

$$|T(\mathbf{r}_k(\mathbf{x})) - T(\mathbf{r}_k(\mathbf{y}))| \leq L|\mathbf{r}_k(\mathbf{x}) - \mathbf{r}_k(\mathbf{y})|, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

Then, with the choice $\varepsilon_\ell \propto \ell^{-\frac{1}{M+2}}$, one has

$$\mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - r^\star(\mathbf{X})\right|^2 \leq \inf_f \ \mathbb{E}\left|f(\mathbf{r}_k(\mathbf{X})) - r^\star(\mathbf{X})\right|^2 + C\ell^{-\frac{2}{M+2}},$$

for some positive constant $C = C(R, L)$, independent of k.

In particular:

$$\mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - r^\star(\mathbf{X})\right|^2 \leq \min_{m=1,\ldots,M} \mathbb{E}\left|r_{k,m}(\mathbf{X}) - r^\star(\mathbf{X})\right|^2 + C\ell^{-\frac{2}{M+2}}.$$

Comments on the rate of convergence

- M is fixed and rather small: Our procedure is designed to take advantage of various heterogeneous basic estimators, not aggregate thousands of very close estimators to select a tuning parameter, for example.

- Our price to pay for aggregating M primal estimators is of the order of $\ell^{-2/(2+M)}$, which is prominently faster than the usual nonparametric rate $\ell^{-2/(2+d)}$.

- Our results are nearly universal in the sense that no strong assumption is made on the distribution of $(\mathbf{X}, Y)$.

Consistency

If at least one machine is consistent, the regression collective inherits this property.

Consistency

If at least one machine is consistent, the regression collective inherits this property.

Corollary

Assume that at least one of the original estimators, say r_{k,m_0}*, satisfies*

$$\mathbb{E}\left|r_{k,m_0}(\mathbf{X}) - r^\star(\mathbf{X})\right|^2 \to 0 \quad \text{as } k \to \infty,$$

for all distribution of $(\mathbf{X}, Y)$ *in some class* $\mathcal{M}$*. Then*

$$\mathbb{E}\left|T_n\left(\mathbf{r}_k(\mathbf{X})\right) - r^\star(\mathbf{X})\right|^2 \to 0 \quad \text{as } k, \ell \to \infty.$$

Extension

- All original estimators are asked to have the same opinion. This can be ruinous if the pool of machines is heterogeneously good.

Extension

- All original estimators are asked to have the same opinion. This can be ruinous if the pool of machines is heterogeneously good.

- This unanimity constraint may be relaxed.

- More sophisticated form: For some $\alpha \in \{1/M, 2/M, \ldots, 1\}$,

$$W_{n,i}(\mathbf{x}) = \frac{\mathbf{1}_{\left\{\sum_{m=1}^{M} \mathbf{1}_{\{|r_{k,m}(\mathbf{x}) - r_{k,m}(\mathbf{x}_i)| \leq \varepsilon_\ell\}} \geq M\alpha\right\}}}{\sum_{j=1}^{\ell} \mathbf{1}_{\left\{\sum_{m=1}^{M} \mathbf{1}_{\{|r_{k,m}(\mathbf{x}) - r_{k,m}(\mathbf{x}_j)| \leq \varepsilon_\ell\}} \geq M\alpha\right\}}}.$$

- Both parameters ε_ℓ and α have a critical impact.

Outline

The R package COBRA

- COBRA: COmBined Regression Alternative.
  Freely available on the CRAN website.

The R package COBRA

- COBRA: COmBined Regression Alternative.
  Freely available on the CRAN website.

- Natively takes advantage of multi-core CPUs (thanks to the `snowfall` package).

- COBRA is written in C, the R package is just a wrapper.

The R package COBRA

- COBRA: COmBined Regression Alternative.
  Freely available on the CRAN website.
- Natively takes advantage of multi-core CPUs (thanks to the `snowfall` package).
- COBRA is written in C, the R package is just a wrapper.
- Input: training sample + (machines) + testing sample.

The R package COBRA

- COBRA: COmBined Regression Alternative.
  Freely available on the CRAN website.
- Natively takes advantage of multi-core CPUs (thanks to the `snowfall` package).
- COBRA is written in C, the R package is just a wrapper.
- Input: training sample + (machines) + testing sample.
- Default machines: `lars`, `ridge`, `FNN`, `tree` and `randomForest`.

The R package COBRA

- COBRA: COmBined Regression Alternative.
  Freely available on the CRAN website.
- Natively takes advantage of multi-core CPUs (thanks to the `snowfall` package).
- COBRA is written in C, the R package is just a wrapper.
- Input: training sample + (machines) + testing sample.
- Default machines: `lars`, `ridge`, `FNN`, `tree` and `randomForest`.
- Better: The user can feed COBRA with her/his own preferred machines. Format: $(n_{\text{train}} + n_{\text{test}}) \times M$ matrix, whose entries are the scalars $\{r_{k,j}(\mathbf{X}_i)\}_{i=1,\ldots,n_{\text{train}}+n_{\text{test}}}^{j=1,\ldots,M}$.

Automatic calibration of ε_ℓ, α, k

- Since the scale of the data may be unknown to the statistician, we devised a data-dependent procedure to select optimal values for ε_ℓ and α.

Automatic calibration of ε_ℓ, α, k

- Since the scale of the data may be unknown to the statistician, we devised a data-dependent procedure to select optimal values for ε_ℓ and α.

- Using a small subsample of $\mathcal{D}_\ell$, we minimize the empirical risk of the COBRA estimator on the grid $\{\varepsilon_{\ell,\min}, \ldots, \varepsilon_{\ell,\max}\} \times \{1/M, 2/M, \ldots, 1\}$, where $\varepsilon_{\ell,\min} = 10^{-300}$, and $\varepsilon_{\ell,\max}$ is proportional to the largest difference between the predictions of the basic machines.

Automatic calibration of ε_ℓ, α, k

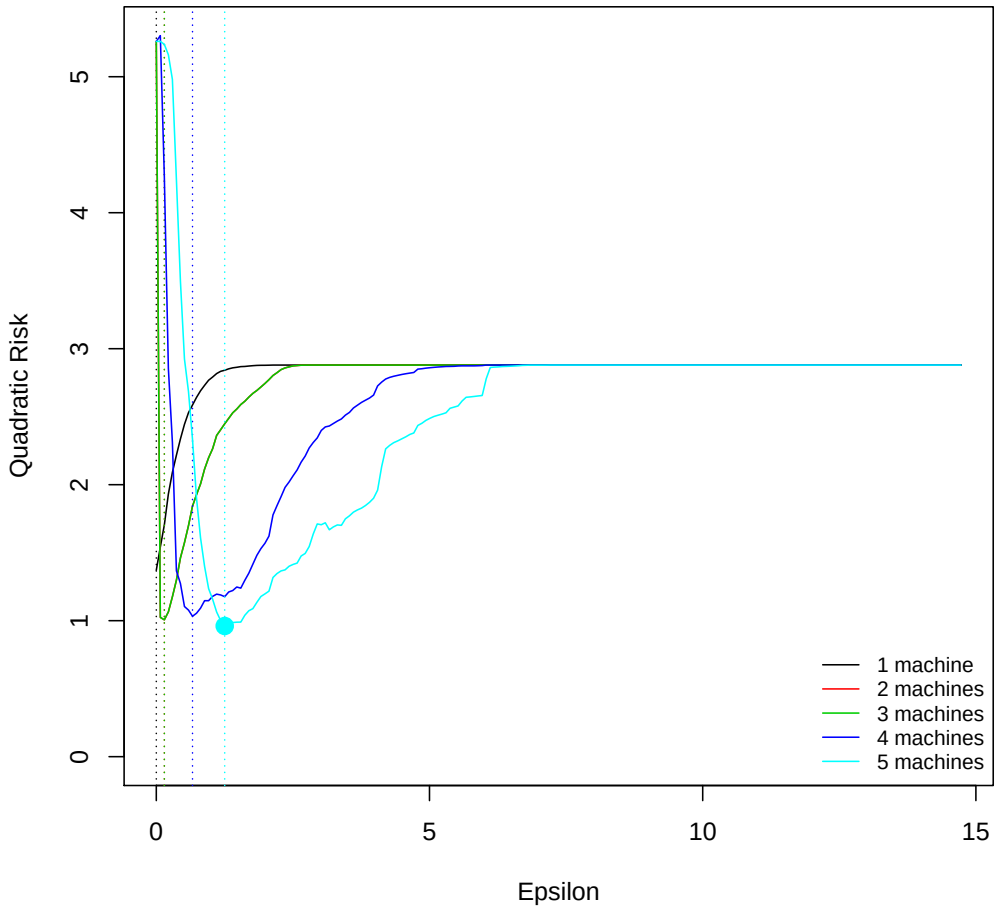
- Since the scale of the data may be unknown to the statistician, we devised a data-dependent procedure to select optimal values for ε_ℓ and α.

- Using a small subsample of $\mathcal{D}_\ell$, we minimize the empirical risk of the COBRA estimator on the grid $\{\varepsilon_{\ell,\min}, \ldots, \varepsilon_{\ell,\max}\} \times \{1/M, 2/M, \ldots, 1\}$, where $\varepsilon_{\ell,\min} = 10^{-300}$, and $\varepsilon_{\ell,\max}$ is proportional to the largest difference between the predictions of the basic machines.

- For large values of n, it appears reasonable to cut the sample in two equal halfs ($k = n/2$). However, when confronted to small sample sizes, we advise to adopt a random cut scheme, compute a few COBRA estimators, then take their mean or median.

Choice of ε_ℓ and α, 1/3

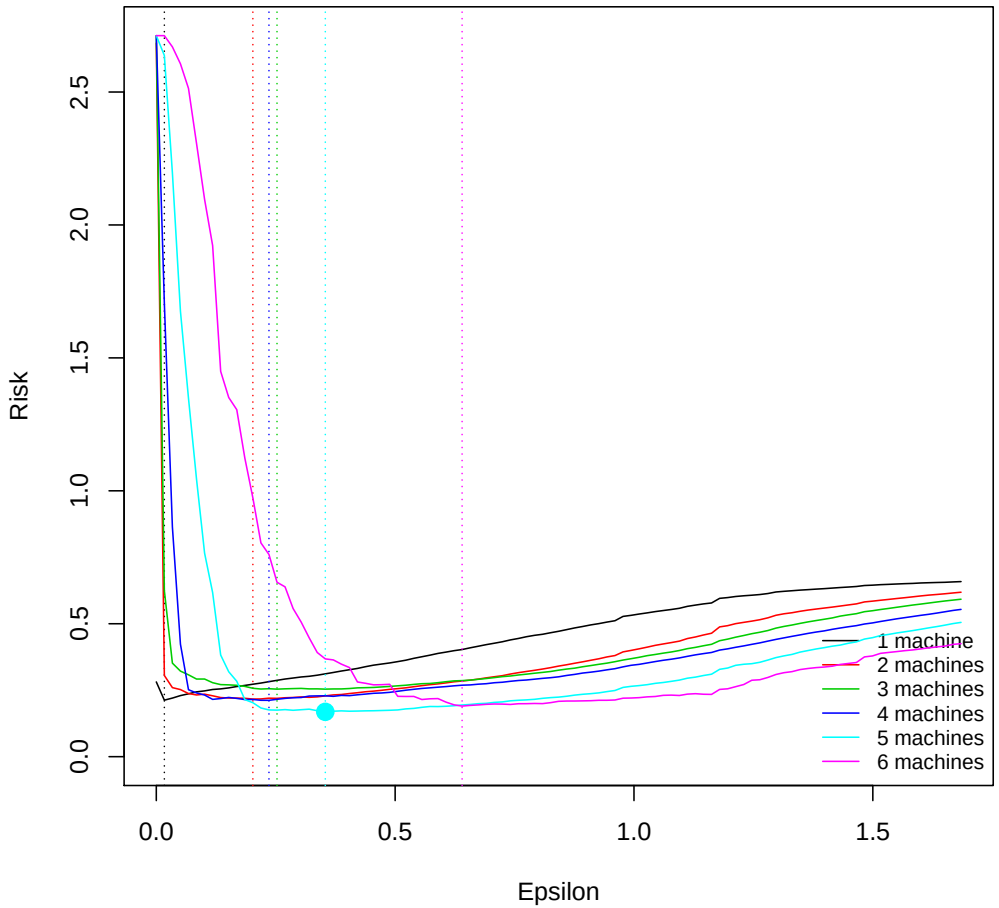
$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 1200,\ d = 10.$$
$$Y = X_1 + 3X_3^2 - 2\exp(-X_5) + X_6.$$



Choice of ε_ℓ and α, 2/3

$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 700,\ d = 20.$$

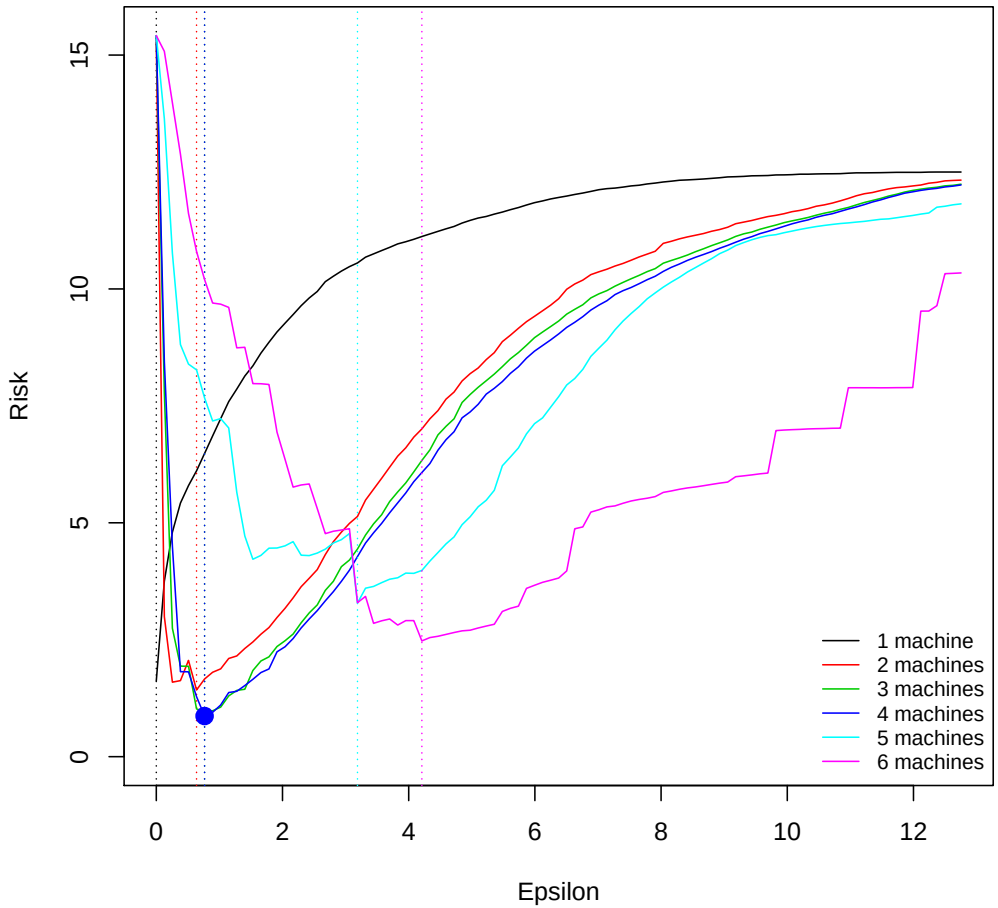
$$Y = \mathbf{1}_{\{X_1 > 0\}} + X_2^3 + \mathbf{1}_{\{X_4 + X_6 - X_8 - X_9 > 1 + X_{14}\}} + \exp(-X_2^2) + \mathcal{N}(0, 0.5).$$



Choice of ε_ℓ and α, 3/3

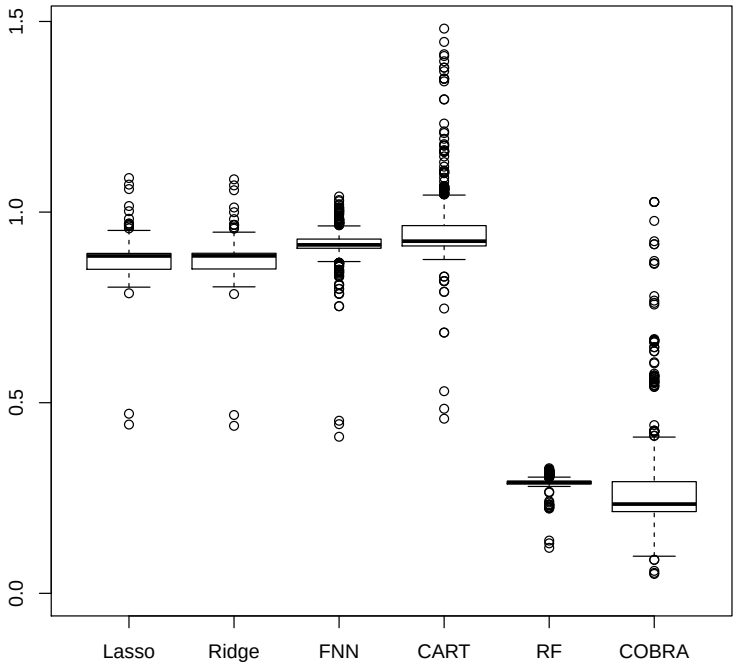
$$\mathbf{X} \sim \mathcal{N}(0, \Sigma), \Sigma_{ij} = 2^{-|i-j|}, \; n = 700, \; d = 20.$$

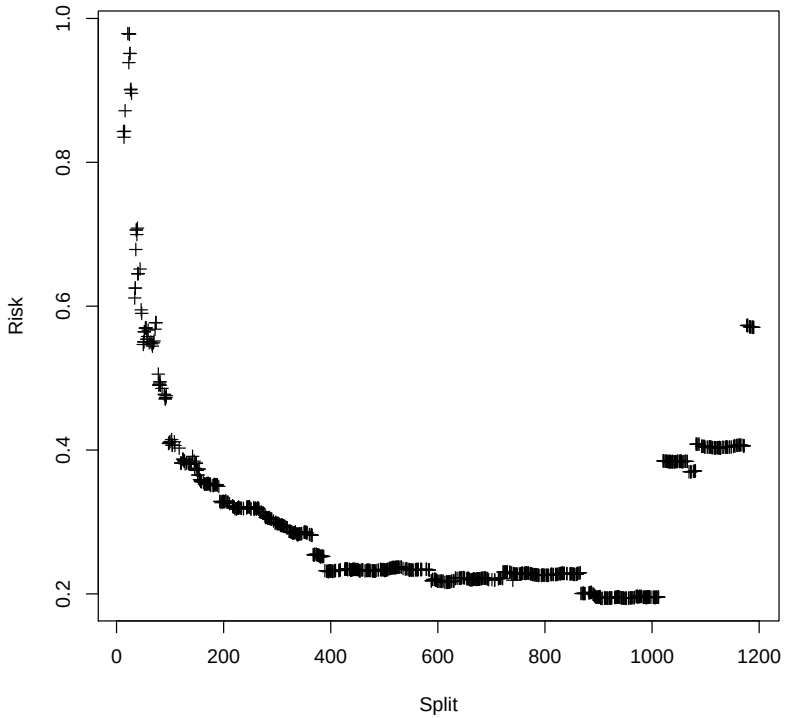
$$Y = \mathbf{1}_{\{X_1 > 0\}} + X_2^3 + \mathbf{1}_{\{X_4 + X_6 - X_8 - X_9 > 1 + X_{14}\}} + \exp(-X_2^2) + \mathcal{N}(0, 0.5).$$



Stability of COBRA: The choice of k

$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 1200,\ d = 10.$$
$$Y = X_1 + 3X_3^2 - 2\exp(-X_5) + X_6.$$

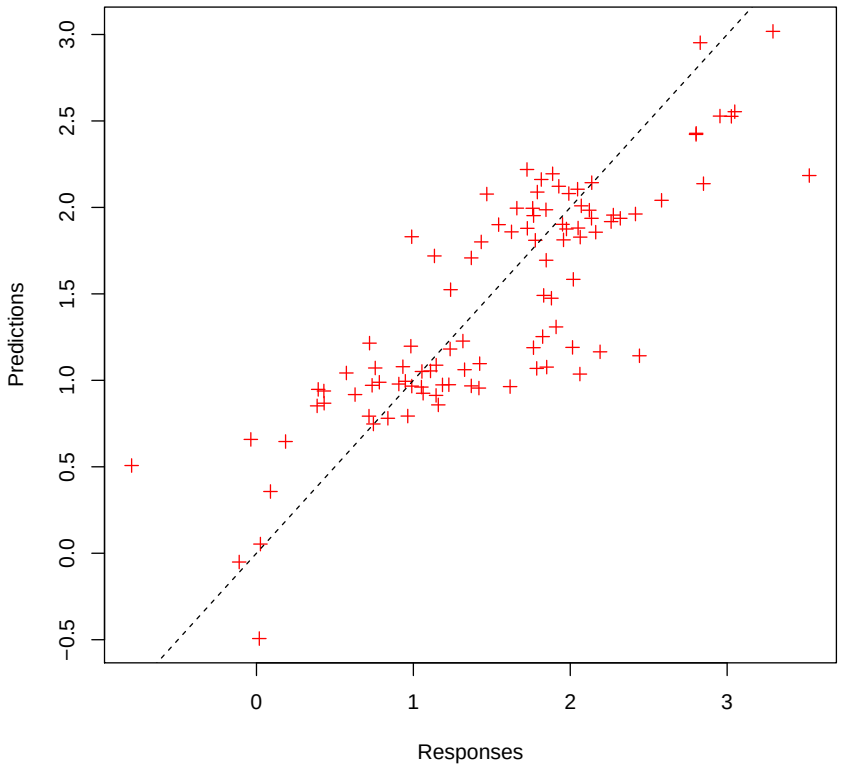


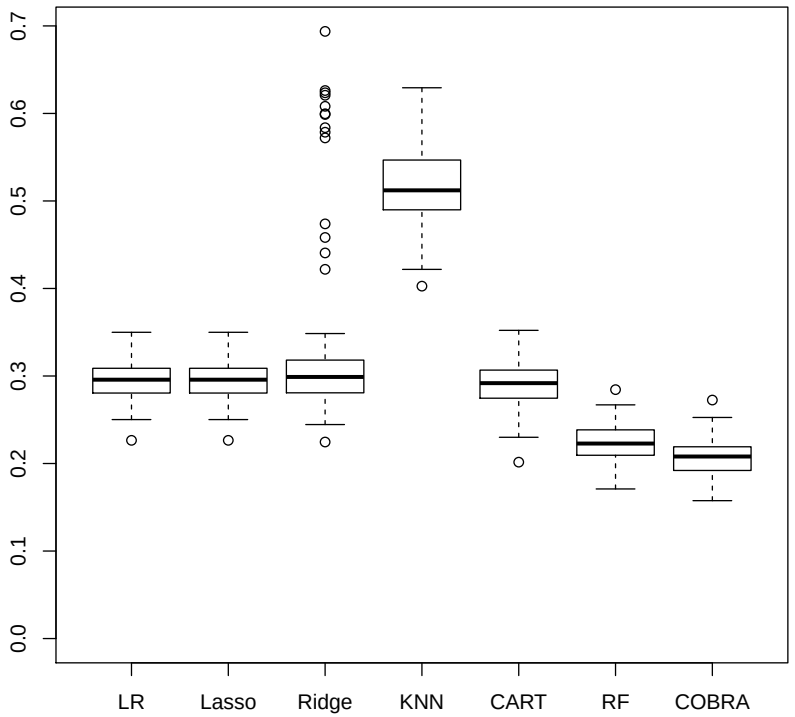


Predictive performance, 1/3

$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 700,\ d = 20.$$

$$Y = \mathbf{1}_{\{X_1>0\}} + X_2^3 + \mathbf{1}_{\{X_4+X_6-X_8-X_9>1+X_{14}\}} + \exp(-X_2^2) + \mathcal{N}(0, 0.5).$$

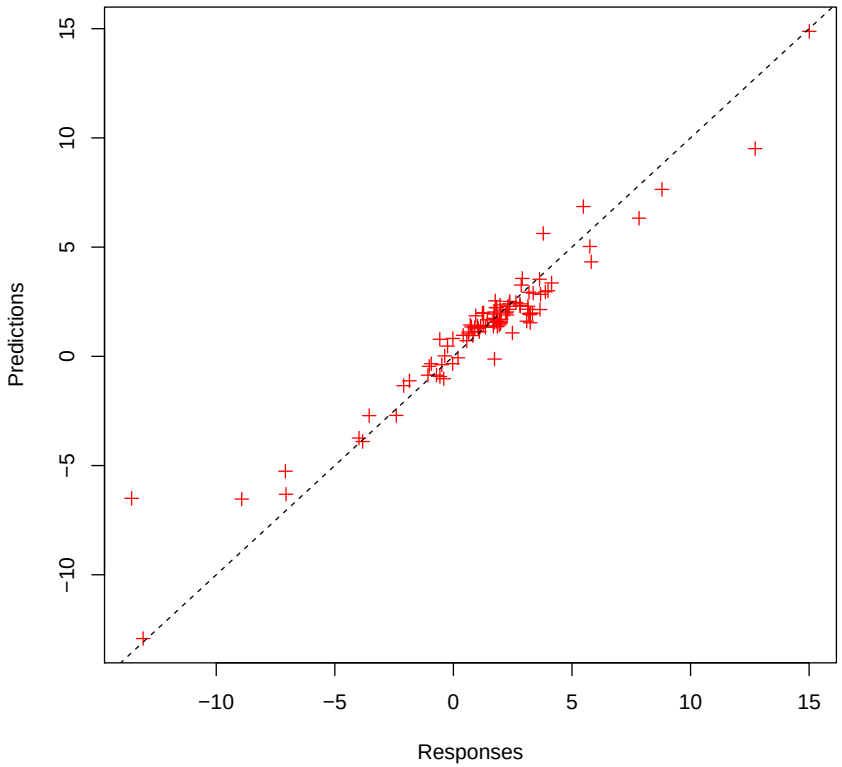


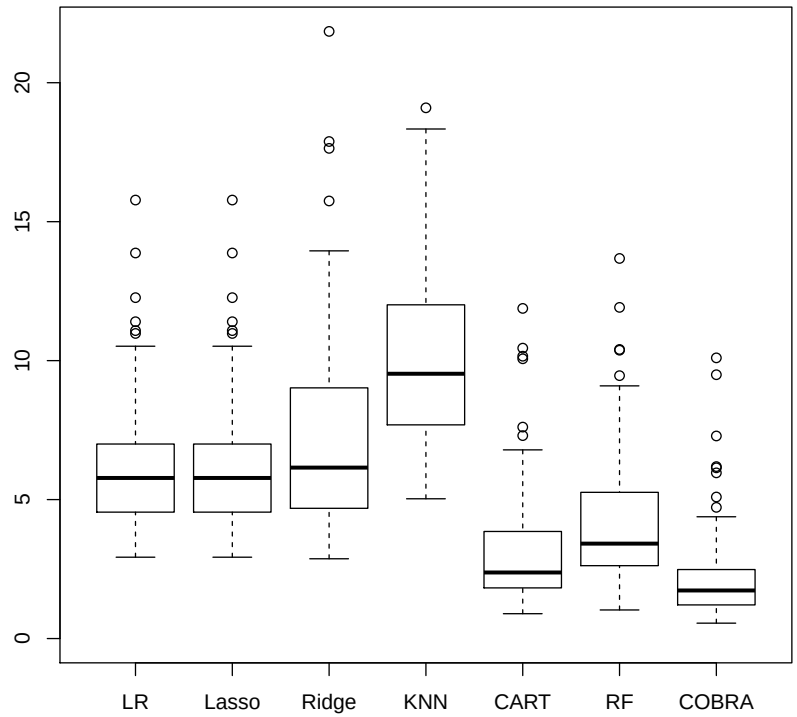


Predictive performance, 2/3

$$\mathbf{X} \sim \mathcal{N}(0, \Sigma), \Sigma_{ij} = 2^{-|i-j|}, \; n = 700, \; d = 20.$$

$$Y = \mathbf{1}_{\{X_1 > 0\}} + X_2^3 + \mathbf{1}_{\{X_4 + X_6 - X_8 - X_9 > 1 + X_{14}\}} + \exp(-X_2^2) + \mathcal{N}(0, 0.5).$$

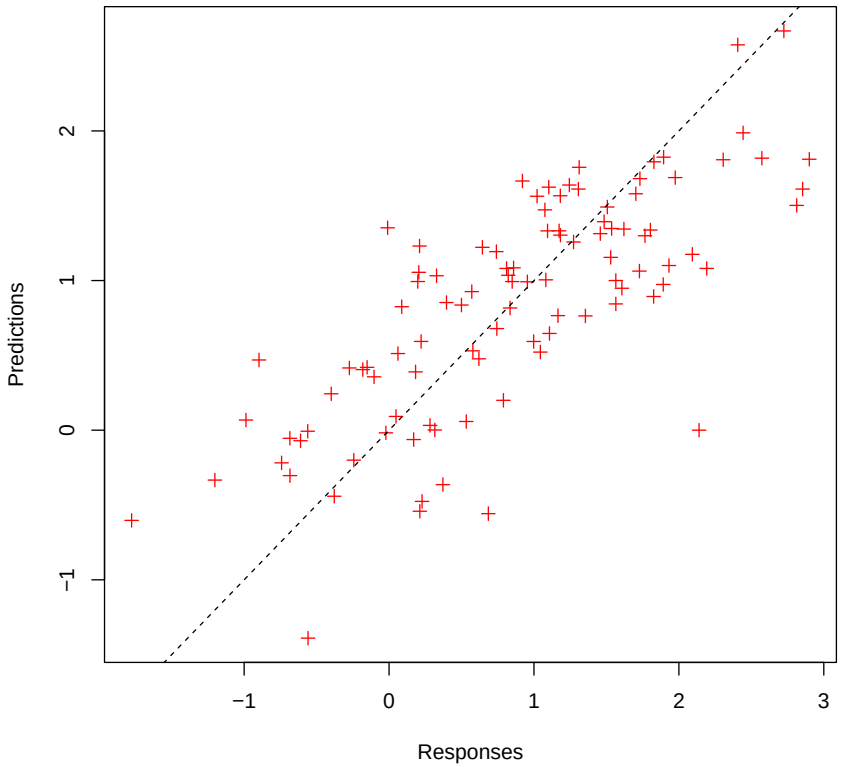


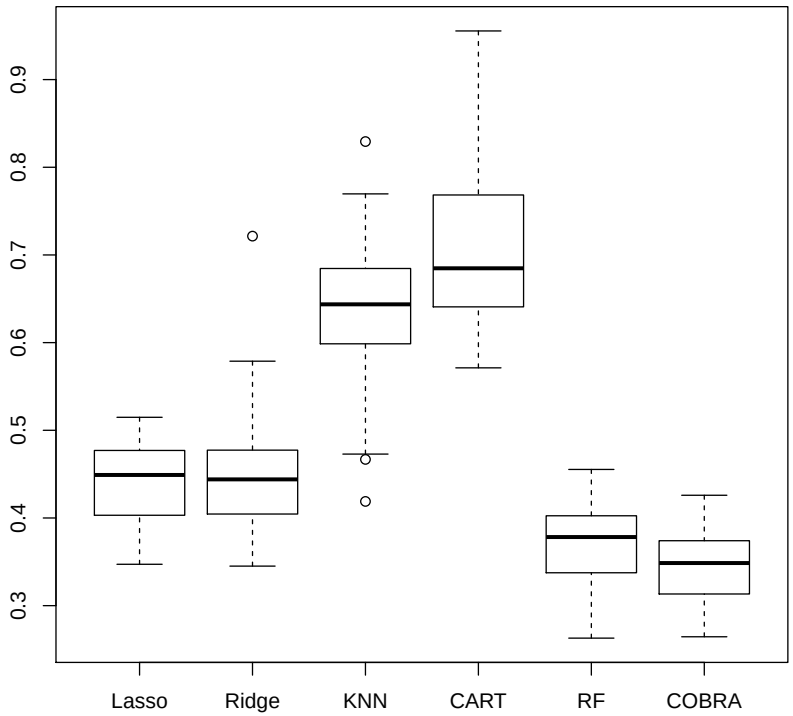


Predictive performance, 3/3

$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 600,\ d = 10.$$

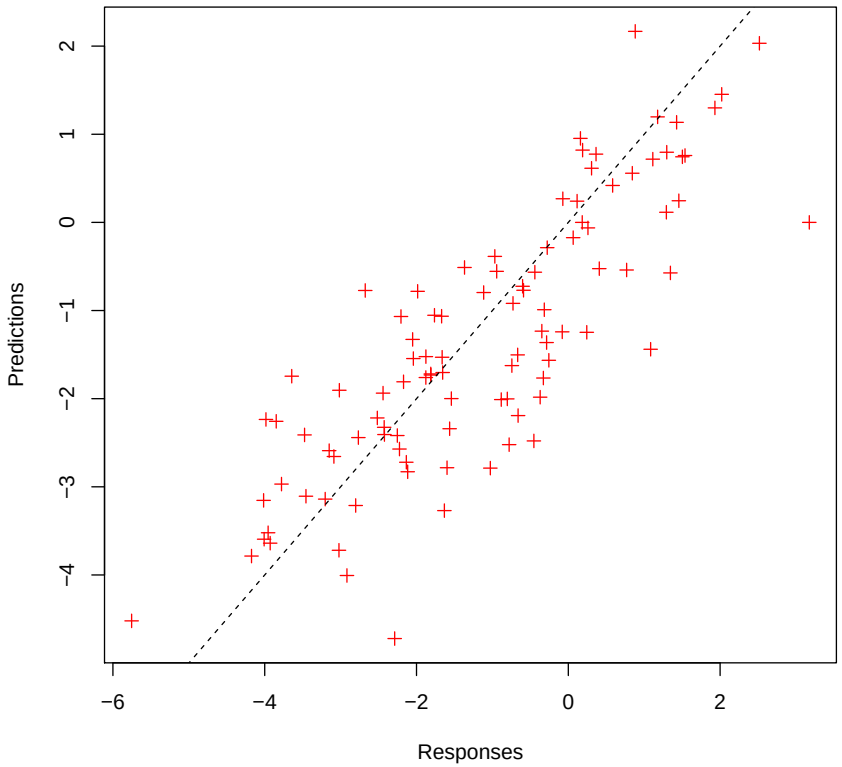
$$Y = \sum_{j=1}^{10} X_j^j + \mathcal{N}(0, 0.1).$$

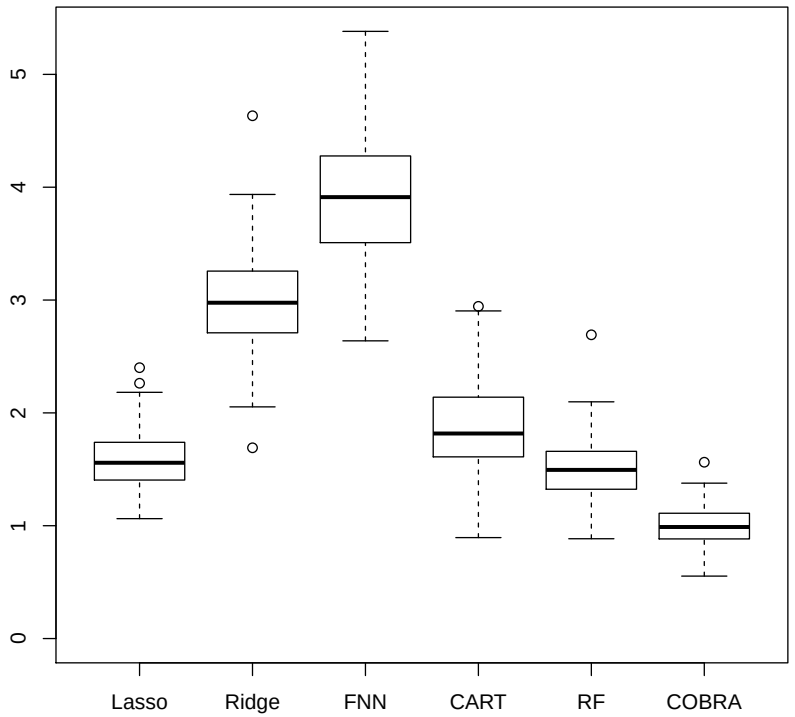




Predictive performance in high dimensions, 1/3

$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 500,\ d = 1000.$$
$$Y = X_1 + 3X_3^2 - 2\exp(-X_5) + X_6.$$

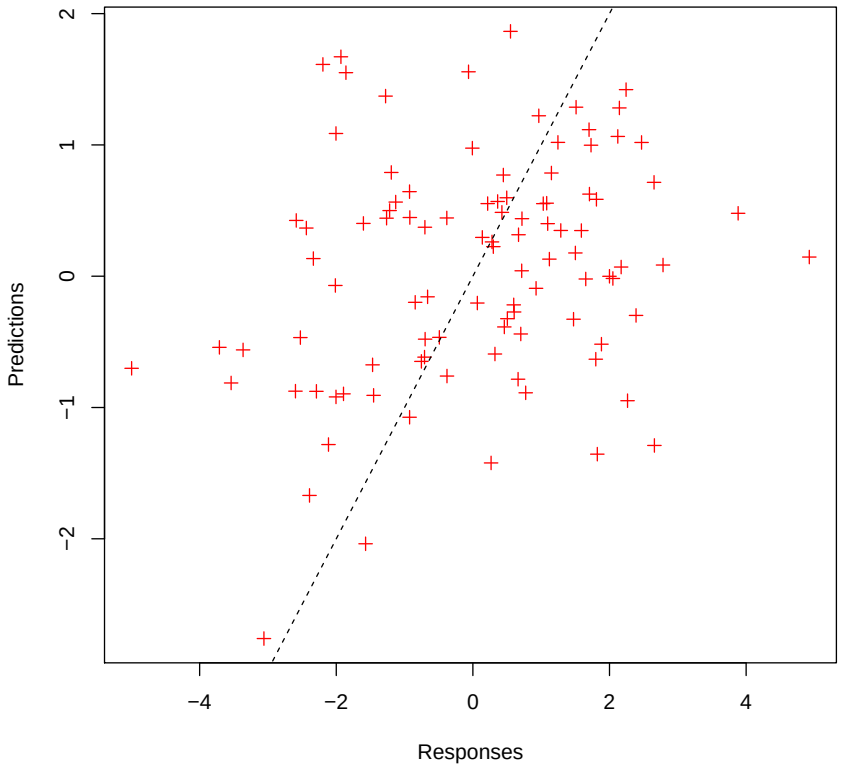


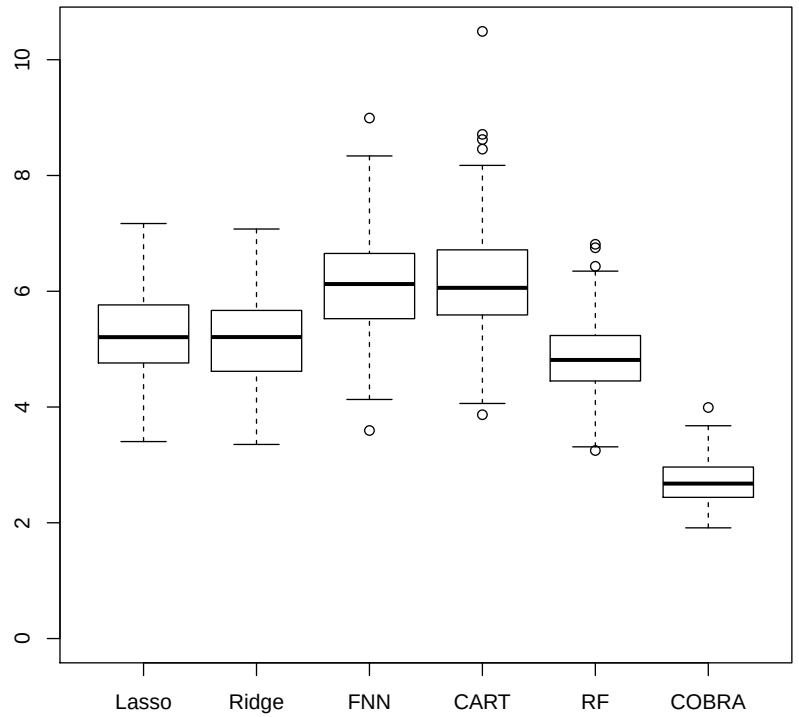


Predictive performance in high dimensions, 2/3

$$\mathbf{X} \sim \mathcal{N}(O, \Sigma), \Sigma_{ij} = 2^{-|i-j|}, \, n = 500, \, d = 1000.$$

$$Y = X_1 + 3X_3^2 - 2\exp(-X_5) + X_6.$$

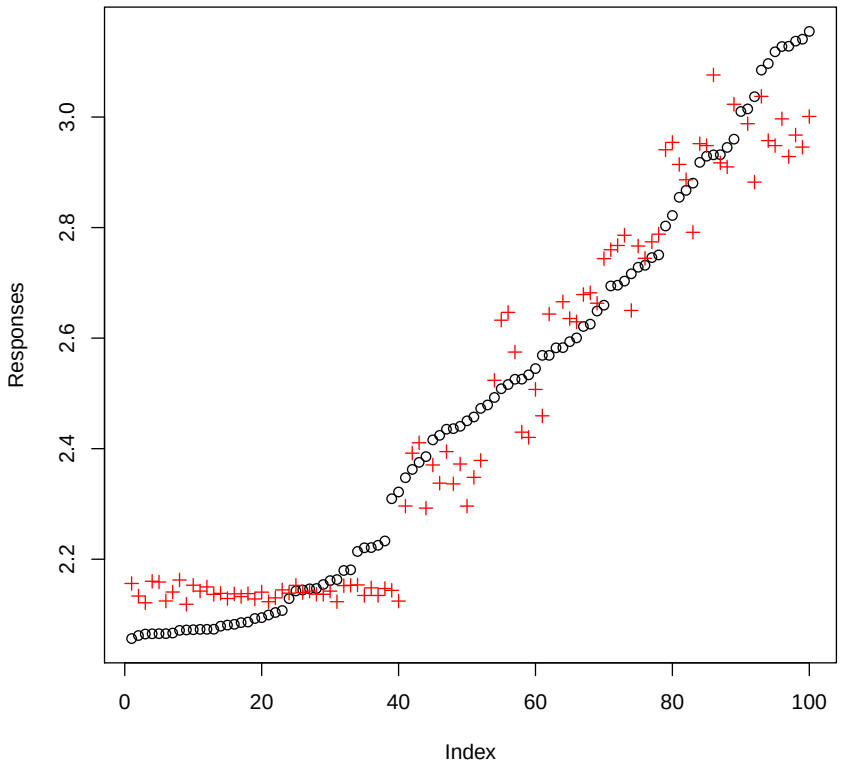


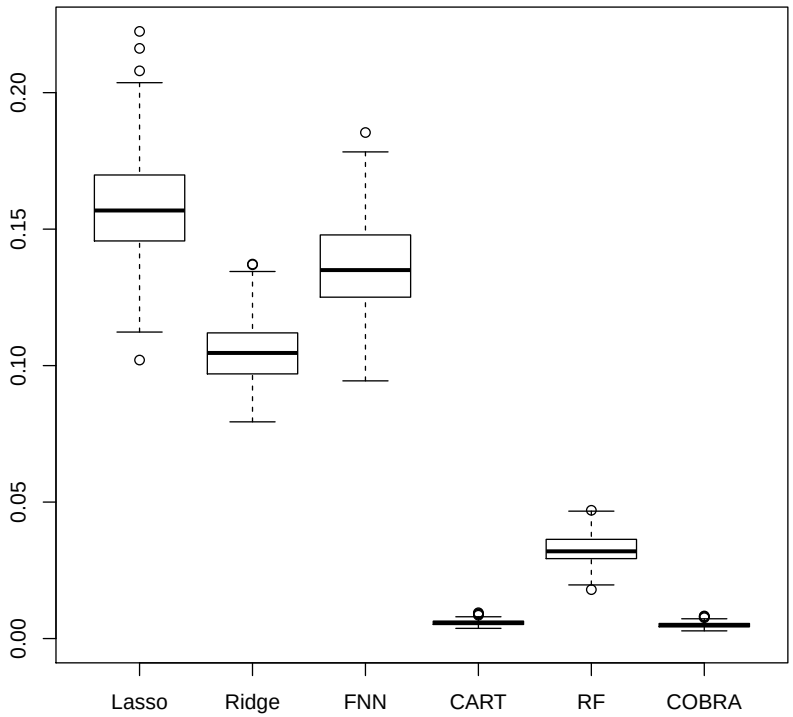


Predictive performance in high dimensions, 3/3

$$\mathbf{X} \sim \mathcal{U}(-1,1)^d,\ n = 500,\ d = 1500.$$

$$Y = \exp(-X_1) + \exp(X_1) + \sum_{j=2}^{d} X_j^{j/100}.$$





Competitors

- Next, we chose two competitors to benchmark COBRA.

Competitors

- Next, we chose two competitors to benchmark COBRA.

- On the theoretical side, exponentially weighted aggregation (EWA) is a popular choice. For all basic estimators $r_{k,1}, \ldots, r_{k,M}$, their empirical risks $\hat{R}_1, \ldots, \hat{R}_M$ are computed on $\mathcal{D}_\ell$ and

$$\mathrm{EWA}_\beta : \mathbf{x} \mapsto \sum_{j=1}^{M} \hat{w}_j r_{k,j}(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^d,$$

where

$$\hat{w}_j = \frac{\exp(-\beta \hat{R}_j)}{\sum_{i=1}^{M} \exp(-\beta \hat{R}_i)}, \quad j = 1, \ldots, M.$$

Competitors

- On the implementation side, we feel close to the philosophy of the Super Learner R package (Polley and van der Laan, 2007). It allows the user to blend various machines together.

Competitors

- On the implementation side, we feel close to the philosophy of the Super Learner R package (Polley and van der Laan, 2007). It allows the user to blend various machines together.
- The Super Learner trains basic machines $r_1, \ldots, r_M$ on the whole sample $\mathcal{D}_n$. Then, following a V-fold cross-validation procedure, Super Learner adopts a V-blocks partition of the set $\{1, \ldots, n\}$ and computes the matrix

$$H = (H_{ij})_{1 \leq i \leq n}^{1 \leq j \leq M},$$

where H_{ij} is the prediction for the query point $\mathbf{X}_i$ made by machine j trained on all remaining $V - 1$ blocks, *i.e.*, excluding the block containing $\mathbf{X}_i$.

Competitors

- On the implementation side, we feel close to the philosophy of the Super Learner R package (Polley and van der Laan, 2007). It allows the user to blend various machines together.
- The Super Learner trains basic machines $r_1, \ldots, r_M$ on the whole sample $\mathcal{D}_n$. Then, following a V-fold cross-validation procedure, Super Learner adopts a V-blocks partition of the set $\{1, \ldots, n\}$ and computes the matrix

$$H = (H_{ij})_{1 \leq i \leq n}^{1 \leq j \leq M},$$

where H_{ij} is the prediction for the query point $\mathbf{X}_i$ made by machine j trained on all remaining $V-1$ blocks, *i.e.*, excluding the block containing $\mathbf{X}_i$. The Super Learner estimator is then

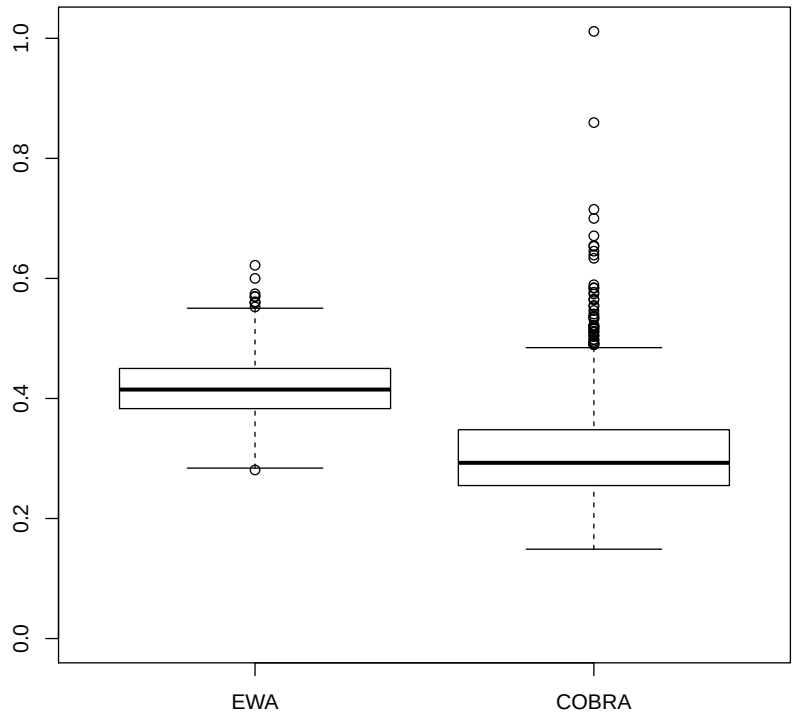
$$SL = \sum_{j=1}^{M} \hat{\alpha}_j r_j, \quad \text{where} \quad \hat{\alpha} \in \operatorname*{arg\,inf}_{\alpha \in \Lambda^M} \sum_{i=1}^{n} |Y_i - (H\alpha)_i|^2.$$

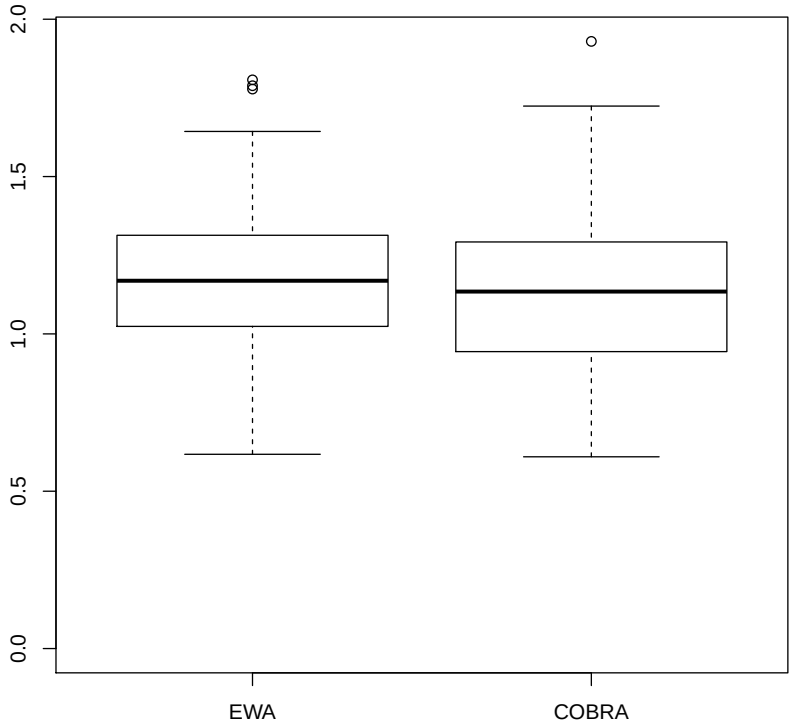
with Λ^M denoting the simplex.

EWA

$n = 500, \mathbf{X} \sim \mathcal{U}(-1,1)^d$, $d = 10$ (left), $\mathbf{X} \sim \mathcal{N}(0,\Sigma)$, $d = 1500$ (right).

$Y = X_1 + 3X_3^2 - 2\exp(-X_5) + X_6.$





Super Learner: Models

Design $\mathbf{X} \sim \mathcal{N}(0, \Sigma)$.

1. $n = 800,\ d = 50,\ Y = X_1^2 + \exp(-X_2^2)$.
2. $n = 600,\ d = 100,$
   $Y = X_1 X_2 + X_3^2 - X_4 X_7 + X_8 X_{10} - X_6^2 + \mathcal{N}(0, 0.5)$.
3. $n = 600,\ d = 100,$
   $Y = -\sin(2X_1) + X_2^2 + X_3 - \exp(-X_4) + \mathcal{N}(0, 0.5)$.
4. $n = 600,\ d = 100,$
   $Y = X_1 + (2X_2 - 1)^2 + \sin(2\pi X_3)/(2 - \sin(2\pi X_3)) + \sin(2\pi X_4) +$
   $2\cos(2\pi X_4) + 3\sin^2(2\pi X_4) + 4\cos^2(2\pi X_4) + \mathcal{N}(0, 0.5)$.
5. $n = 700,\ d = 20,\ Y =$
   $\mathbf{1}_{\{X_1 > 0\}} + X_2^3 + \mathbf{1}_{\{X_4 + X_6 - X_8 - X_9 > 1 + X_{14}\}} + \exp(-X_2^2) + \mathcal{N}(0, 0.5)$.
6. $n = 500,\ d = 30,\ Y = \sum_{k=1}^{10} \mathbf{1}_{\{X_k^3 < 0\}} - \mathbf{1}_{\{\mathcal{N}(0,1) > 1.25\}}$.
7. $n = 600,\ d = 300,$
   $Y = X_1^2 + X_2^2 X_3 \exp(-|X_4|) + X_6 - X_8 + \mathcal{N}(0, 0.5)$.
8. $n = 600,\ d = 50,\ Y = \mathbf{1}_{\{X_1 + X_4^3 + X_9 + \sin(X_{12} X_{18}) + \mathcal{N}(0,0.1) > 0.38\}}$.

COBRA vs. SuperLearner

COBRA

1	2	3	4	5	6	7	8
0.3262	**1.3984**	3.3201	**9.3964**	**4.9990**	1.1988	3.1401	**0.1045**
(0.1242)	(0.3804)	(1.8056)	(2.8953)	(9.3103)	(0.4573)	(1.6097)	(0.0216)
11.96	**14.16**	**11.92**	**13.11**	**5.02**	**4.12**	**41.28**	**6.24**
(0.27)	(0.57)	(0.41)	(0.34)	(0.07)	(0.15)	(2.84)	(0.11)

SuperLearner

1	2	3	4	5	6	7	8
0.8733	2.3391	**3.1885**	25.1073	5.6478	**0.8967**	**3.0367**	0.1116
(0.2740)	(0.4958)	(1.5101)	(7.3179)	(7.7271)	(0.1197)	(1.6225)	(0.0111)
61.92	70.90	59.91	63.58	31.24	24.29	145.18	31.31
(1.85)	(2.47)	(2.06)	(1.21)	(0.86)	(0.82)	(8.97)	(0.73)

Highlights

- Original nonlinear strategy backed up by sharp oracle inequalities and an explicit rate of convergence.

  Biau, Fischer, G. and Malley (2013). COBRA: A Nonlinear Aggregation Strategy. arXiv preprint.

- R package COBRA: extremely fast, flexible and competitive implementation.

  Version 0.99.4 on the CRAN.