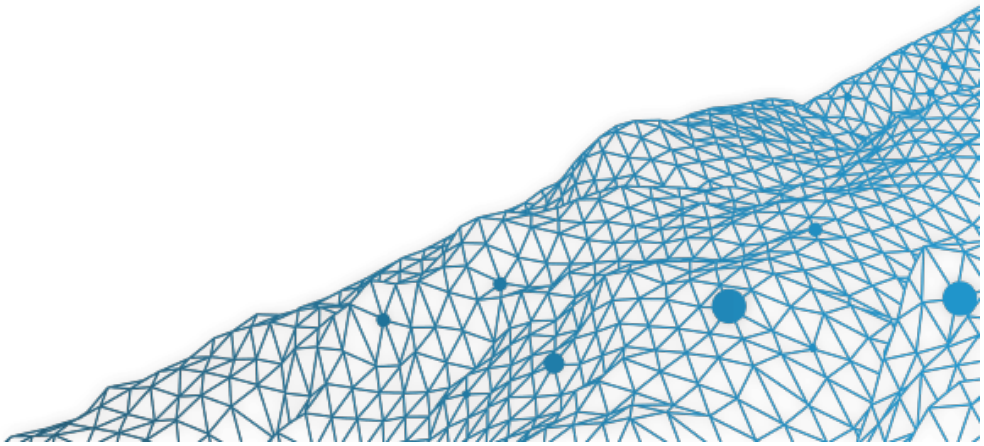
9 Updates & Missing Values

Arthur Charpentier (Université du Québec à Montréal)

Machine Learning & Econometrics

SIDE Summer School - July 2019



Machine Learning, Practical Issues

Two important practical issue :

- what if we cannot access the entire dataset ?
- what if there is an update ? (new observation or new variable)

Consider the case where datasets are located on various servers, and cannot be downloaded (e.g. hospitals), but one can run functions and obtain outputs.
see Wolfson *et al.* (2010, Data Shield)
or http://www.datashield.ac.uk/

Consider a regression model $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$

Machine Learning, Practical Issues

Use the QR decomposition of $\boldsymbol{X}$, $\boldsymbol{X}=\boldsymbol{Q}\boldsymbol{R}$ where $\boldsymbol{Q}$ is an orthogonal matrix $\boldsymbol{Q}^\mathsf{T}\boldsymbol{Q}=\mathbb{I}$. Then

$$\widehat{\boldsymbol{\beta}}=[\boldsymbol{X}^\mathsf{T}\boldsymbol{X}]^{-1}\boldsymbol{X}^\mathsf{T}\boldsymbol{y}=\boldsymbol{R}^{-1}\boldsymbol{Q}^\mathsf{T}\boldsymbol{y}$$

Consider m blocks - map part

$$\boldsymbol{y}=\begin{bmatrix}\boldsymbol{y}_1\\ \boldsymbol{y}_2\\ \vdots\\ \boldsymbol{y}_m\end{bmatrix} \text{ and } \boldsymbol{X}=\begin{bmatrix}\boldsymbol{X}_1\\ \boldsymbol{X}_2\\ \vdots\\ \boldsymbol{X}_m\end{bmatrix}=\begin{bmatrix}\boldsymbol{Q}_1^{(1)}\boldsymbol{R}_1^{(1)}\\ \boldsymbol{Q}_2^{(1)}\boldsymbol{R}_2^{(1)}\\ \vdots\\ \boldsymbol{Q}_m^{(1)}\boldsymbol{R}_m^{(1)}\end{bmatrix}$$

Machine Learning, Practical Issues

Consider the QR decomposition of $\boldsymbol{R}^{(1)}$ - step 1 of reduce part

$$\boldsymbol{R}^{(1)} = \begin{bmatrix} \boldsymbol{R}_1 \\ \boldsymbol{R}_2 \\ \vdots \\ \boldsymbol{R}_m \end{bmatrix} = \boldsymbol{Q}^{(2)}\boldsymbol{R}^{(2)} \text{ where } \boldsymbol{Q}^{(2)} = \begin{bmatrix} \boldsymbol{Q}_1^{(2)} \\ \boldsymbol{Q}_2^{(2)} \\ \vdots \\ \boldsymbol{Q}_m^{(2)} \end{bmatrix}$$

define - step 2 of reduce part

$$\boldsymbol{Q}_j^{(3)} = \boldsymbol{Q}_j^{(2)}\boldsymbol{Q}_j^{(1)} \text{ and } \boldsymbol{V}_j = \boldsymbol{Q}_j^{(3)\mathsf{T}}\boldsymbol{y}_j$$

and finally set - step 3 of reduce part

$$\widehat{\boldsymbol{\beta}} = [\boldsymbol{R}^{(2)}]^{-1}\sum_{j=1}^{m}\boldsymbol{V}_j$$

Online Learning

Let $T_n = \{(y_1, \boldsymbol{x}_1), \cdots, (y_n, \boldsymbol{x}_n)\}$ denote the training dataset, with $y \in \mathcal{Y}$.

Learning

A learning algorithm is a map $A : T_n \to \mathcal{Y}$

Online Learning

A pure online learning algorithm is a sequence of recursive algorithms
(i) m_0 is the initialization
(ii) for $k = 1, 2 \cdots$, $m_k = A(m_{k-1}, (y_n, \boldsymbol{x}_n))$

Recall that the risk is $\mathcal{R}(m) = \mathbb{E}\big[\ell(Y, m\boldsymbol{X})\big]$

As in gradient boosting, consider some approximation of the gradient of $\mathcal{R}(m)$,

$$m_k = m_{k-1} + \gamma_k G(m_{k-1}, (y_n, \boldsymbol{x}_n))$$

• Update with a new observation, as Ridell (1975, Recursive Estimation Algorithms for Economic Research)

Let $\boldsymbol{X}_{1:n}$ denote the matrix of covariates, with n observations (rows), and $\boldsymbol{x}_{n+1}$ denote a new one. Recall that

$$\widehat{\boldsymbol{\beta}}_n = [\boldsymbol{X}_{1:n}^\intercal \boldsymbol{X}_{1:n}]^{-1} \boldsymbol{X}_{1:n}^\intercal \boldsymbol{y}_{1:n} = C_n^{-1} \boldsymbol{X}_{1:n}^\intercal \boldsymbol{y}_{1:n}$$

Since $C_{n+1} = \boldsymbol{X}_{1:n+1}^\top \boldsymbol{X}_{1:n+1} = C_n + \boldsymbol{x}_{n+1}\boldsymbol{x}_{n+1}^\top$, then

$$\widehat{\boldsymbol{\beta}}_{n+1} = \widehat{\boldsymbol{\beta}}_n + C_{n+1}^{-1}\boldsymbol{x}_{n+1}[y_{n+1} - \boldsymbol{x}_{n+1}^\top \widehat{\boldsymbol{\beta}}_n]$$

This updating formation is also called a differential correction, since it is proportional to the prediction error.

Note that the residual sum of squares can also be updated, with

$$S_{n+1} = S_n + \frac{1}{d}[y_{n+1} - \boldsymbol{x}_{n+1}^\intercal \widehat{\boldsymbol{\beta}}_n]^2$$

Online Learning

Online Learning for OLS

$$\widehat{\boldsymbol{\beta}}_{n+1} = \widehat{\boldsymbol{\beta}}_n + C_{n+1}^{-1}\boldsymbol{x}_{n+1}[y_{n+1} - \boldsymbol{x}_{n+1}^\top \widehat{\boldsymbol{\beta}}_n]$$

is a recursive formula, requires storing all the data

(and inverting a matrix at each step).

Good news, $[A+BCD]^{-1} = A^{-1} - A^{-1}B\left[DA^{-1}B + C^{-1}\right]^{-1}DA^{-1}$, so

$$C_{n+1}^{-1} = C_n^{-1} - \frac{C_n^{-1}\boldsymbol{x}_{n+1}\boldsymbol{x}_{n+1}^\top C_n^{-1}}{1+\boldsymbol{x}_{n+1}^\top C_n^{-1}\boldsymbol{x}_{n+1}}$$

We have an algorithm of the form for $k = 1, 2\cdots$, $m_k = A(m_{k-1}, (y_n, C_n, \boldsymbol{x}_n))$

for some matrix C_n

Online Learning

Online Learning for OLS

$$\widehat{\boldsymbol{\beta}}_{n+1} = \widehat{\boldsymbol{\beta}}_n + C_{n+1}^{-1}\boldsymbol{x}_{n+1}[y_{n+1} - \boldsymbol{x}_{n+1}^\top \widehat{\boldsymbol{\beta}}_n]$$

is also a *gradient-type* algorithm, since

$$\nabla\left(y_{n+1} - \boldsymbol{x}_{n+1}^\top\boldsymbol{\beta}\right)^2 = 2\boldsymbol{x}_{n+1}[y_{n+1} - \boldsymbol{x}_{n+1}^\top\boldsymbol{\beta}]$$

One might consider using $\gamma_{n+1} \in \mathbb{R}$ instead of C_{n+1} ($p \times p$ matrix)

Polyak-Ruppert Averaging suggests to use $\gamma_n = n^{-\alpha}$ where $\alpha \in (1/2, 1)$ to ensure convergence

Update Formulas

- Update with a new variable

Let $\boldsymbol{X}_{1:k}$ denote the matrix of covariates, with k explanatory variables (columns), and $\boldsymbol{x}_{k+1}$ denote a new one. Recall that

$$\widehat{\boldsymbol{\beta}}_k = [\boldsymbol{X}_{1:k}^{\mathsf{T}}\boldsymbol{X}_{1:k}]^{-1}\boldsymbol{X}_{1:k}^{\mathsf{T}}\boldsymbol{y}$$

Then $\widehat{\boldsymbol{\beta}}_{k+1} = (\widehat{\boldsymbol{\beta}}_k^{\star}\ ,\ \widehat{\beta}_{k+1}^{\star})^{\mathsf{T}}$ where

$$\widehat{\boldsymbol{\beta}}_k^{\star} = \widehat{\boldsymbol{\beta}}_k - \frac{[\boldsymbol{X}_{1:k}^{\mathsf{T}}\boldsymbol{X}_{1:k}]^{-1}\boldsymbol{X}_{1:k}^{\mathsf{T}}\boldsymbol{x}_{k+1}\boldsymbol{x}_{k+1}P_k^{\perp}\boldsymbol{y}}{\boldsymbol{x}_{k+1}^{\mathsf{T}}P_k^{\perp}\boldsymbol{x}_{k+1}}$$

with $P_k^{\perp} = \mathbb{I} - \boldsymbol{X}_{1:k}(\boldsymbol{X}_{1:k}^{\mathsf{T}}\boldsymbol{X}_{1:k})^{-1}\boldsymbol{X}_{1:k}^{\mathsf{T}}$, while

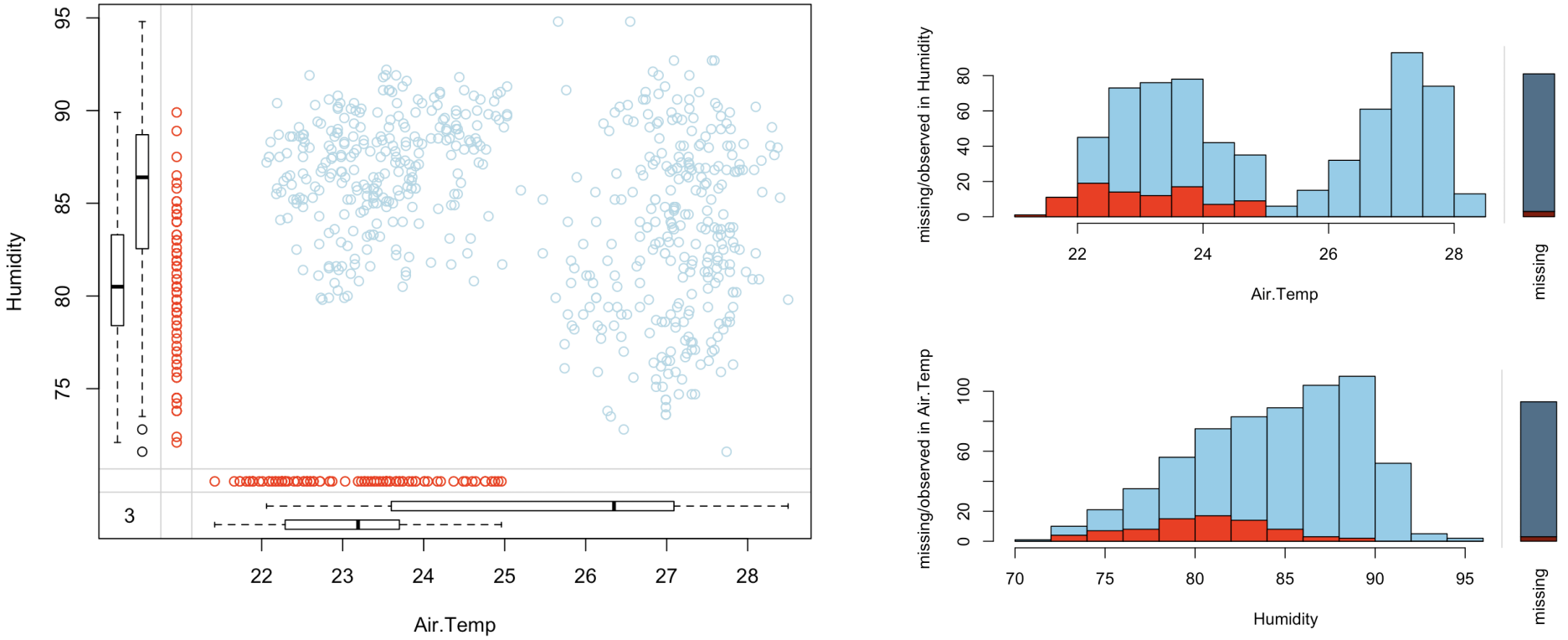
$$\widehat{\beta}_{k+1}^{\star} = \frac{\boldsymbol{x}_{k+1}^{\mathsf{T}}P_k^{\perp}\boldsymbol{y}}{\boldsymbol{x}_{k+1}^{\mathsf{T}}P_k^{\perp}\boldsymbol{x}_{k+1}}$$

If $\boldsymbol{x}_{k+1}$ is orthogonal to previous variables - $\boldsymbol{X}_{1:k}^{\mathsf{T}}\boldsymbol{x}_{k+1} = \boldsymbol{0}$, then $\widehat{\boldsymbol{\beta}}_k^{\star} = \widehat{\boldsymbol{\beta}}_k$.

Observe that $P_k^{\perp}\boldsymbol{y} = \varepsilon_k$.

Missing Values

"There are two kinds of model in the world : those who can extrapolate from incomplete data..."



From Tropical Atmosphere Ocean (TAO) dataset, see `VIM::tao`

Missing Values

With `lm` function, rows with missing values (in y or x) are deleted

To deal with them, one should understand the mechanism leading to missing values

Expectation - Maximization, see Dempster *et al.* (1977, Maximum Likelihood from Incomplete Data via the EM Algorithm)

Consider a mixture model $dF(y) = p_1 dF_{\boldsymbol{\theta}_1}(y) + p_2 dF_{\boldsymbol{\theta}_2}(y)$, i.e. there is $\Theta \in \{1, 2\}$ (with $p_j = \mathbb{P}[\Theta = j]$) such that

$$y_i = \begin{cases} y_{1,i} \text{ with } Y_1 \sim F_{\boldsymbol{\theta}_1}, \text{ if } \Theta = 1 \\ y_{2,i} \text{ with } Y_2 \sim F_{\boldsymbol{\theta}_2}, \text{ if } \Theta = 2 \end{cases}$$

see `mixtools::normalmixEM` for Gaussian mixtures

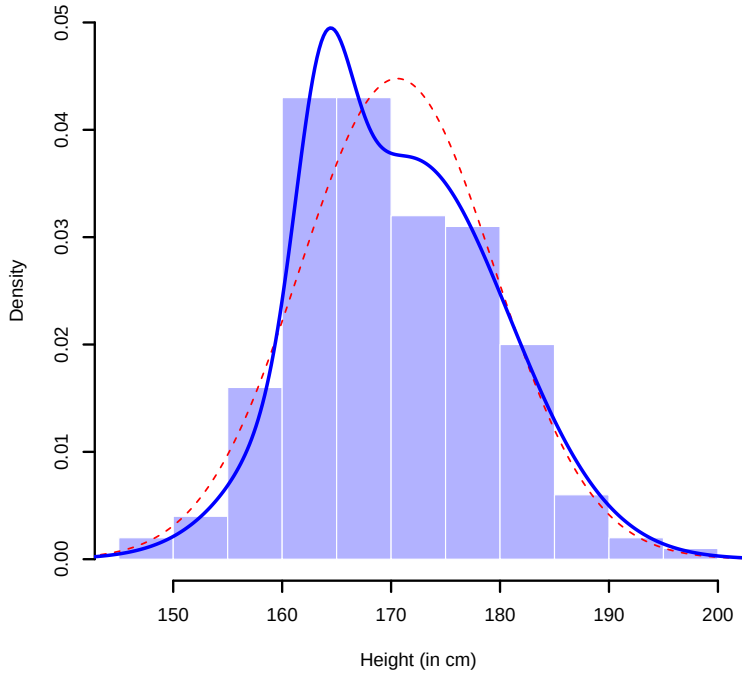
Observable and Non-Obsevable Heterogeneity

Mixture distribution (with two classes) :

- if $\theta = \text{A}$, $Y \sim \mathcal{N}(\mu_A, \sigma_A^2)$
- if $\theta = \text{B}$, $Y \sim \mathcal{N}(\mu_B, \sigma_B^2)$

$$f(y) = p_A f_A(y) + p_B f_B(y)$$

5 parameters to estimate,
no interpretation of the mixture parameter θ



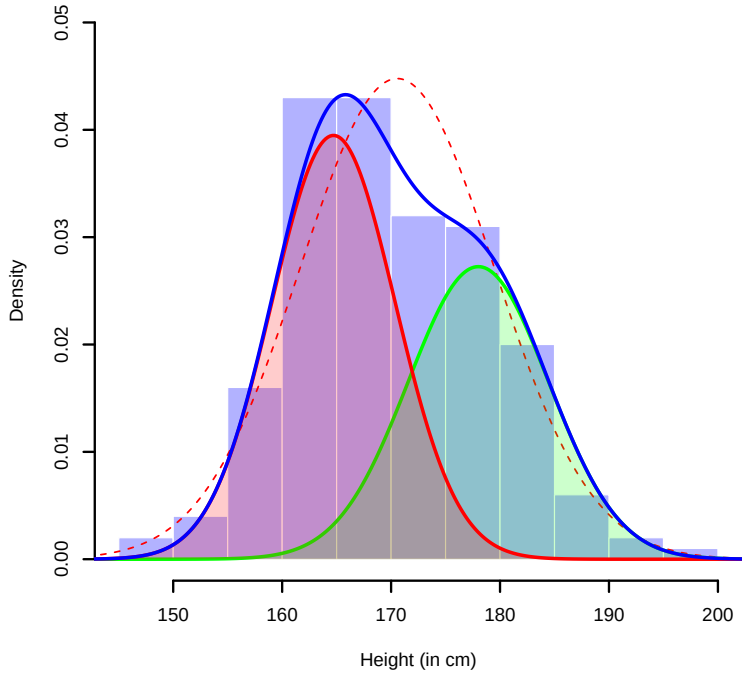
Observable and Non-Observable Heterogeneity

One categorical variable (e.g. gender)

- if gender=`M`, $Y \sim \mathcal{N}(\mu_M, \sigma_M^2)$
- if gender=`F`, $Y \sim \mathcal{N}(\mu_F, \sigma_F^2)$

$$f(y) = p_M f_M(y) + p_F f_F(y)$$

4 parameters to estimate,
(p_M and p_F are known)
clear interpretation of the mixture parameter



Expectation - Maximization

EM for Mixtures

(i) start with initial values $\widehat{\boldsymbol{\theta}}_{1,0}$ and $\widehat{\boldsymbol{\theta}}_{2,0}$, $\widehat{p}_{j,0}$

(ii) for $k = 1, 2, \cdots$

E step : $$\widehat{\gamma}_{k,j,i} = \frac{dF_{\widehat{\boldsymbol{\theta}}_{j,k-1}}(y_i)}{\widehat{p}_{1,k-1} dF_{\widehat{\boldsymbol{\theta}}_{1,k-1}}(y_i) + \widehat{p}_{2,k-1} dF_{\widehat{\boldsymbol{\theta}}_{2,k-1}}(y_i)}$$

M step : use ML techniques with weights $\widehat{\gamma}_{k,j,i}$

M step with a Gaussian mixture, $\widehat{\mu}_{j,k} = \dfrac{\sum \widehat{\gamma}_{k,j,i} y_i}{\sum \widehat{\gamma}_{k,j,i}}$ and $\widehat{\sigma}^2_{j,k} = \dfrac{\sum \widehat{\gamma}_{k,j,i} [y_i - \widehat{\mu}_{j,k}]^2}{\sum \widehat{\gamma}_{k,j,i}}$

Expectation - Maximization

Expectation - Maximization

E step expectation : compute $Q(\theta, \theta^k) = \mathbb{E}\Big[\log f(Y|\theta)|y_{obs}, \theta^k\Big]$

M step maximization : $\theta^{k+1} = \underset{\theta}{\text{argmax}}\Big\{Q(\theta, \theta^k)\Big\}$

Stochastic EM (for Mixtures)

(i) start with initial values $\widehat{\boldsymbol{\theta}}_{1,0}$ and $\widehat{\boldsymbol{\theta}}_{2,0}$, $\widehat{p}_{j,0}$

(ii) for $k = 1, 2, \cdots$

E step : $\widehat{\gamma}_{k,j,i} = \dfrac{dF_{\widehat{\boldsymbol{\theta}}_{j,k-1}}(y_i)}{\widehat{p}_{1,k-1}dF_{\widehat{\boldsymbol{\theta}}_{1,k-1}}(y_i) + \widehat{p}_{2,k-1}dF_{\widehat{\boldsymbol{\theta}}_{2,k-1}}(y_i)}$

S step : generate $\xi_{k,i}$ in $\{1, 2\}$ with probabilities $\widehat{\gamma}_{k,1,i}$ and $\widehat{\gamma}_{k,2,i}$

M step : compute ML estimate $\widehat{\boldsymbol{\theta}}_{k,j}$ on sample $\{y_i : \xi_{k,i} = j\}$

Missing Values : Single Imputation

Classical idea : Principal Component Analysis (PCA)

Approximate $n \times p$ matrix $\boldsymbol{X}$ with a lower rank matrix,

$$\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\leq s}{\text{argmin}} \left\{ \|\boldsymbol{X}-\boldsymbol{Y}\|_2^2 \right\} = \boldsymbol{U}_s \boldsymbol{\Lambda}_s^{1/2} \boldsymbol{V}_s^\top$$

(using Singular Value Decomposition)

One can consider PCA with missing values, based on weighted least squares

$$\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\leq s}{\text{argmin}} \left\{ \|\boldsymbol{W}(\boldsymbol{X}-\boldsymbol{Y})\|_2^2 \right\}$$

where $\boldsymbol{W}$ is the $n \times p$ matrix with 1's, and $W_{i,j} = 0$ if $x_{i,j}$ is missing, see Gabriel & Zamir (1979, Lower rank approximation of matrices by least squares with any choice of weights) or Kiers (1997, Weighted least squares fitting using ordinary least squares algorithms)

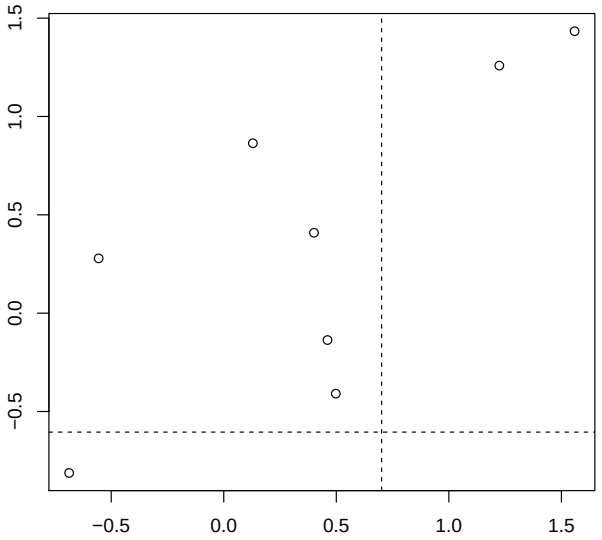
Missing Values : Single Imputation

Iterative PCA

(i) if $x_{i,j}$ is missing, $W_{i,j}=0$,
$x^1_{i,j} = W_{i,j} \cdot x^0_{i,j} + (1-W_{i,j}) \cdot 0$

(ii) for $k = 1, 2, \cdots$

- $\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\leq s}{\text{argmin}} \left\{ \|\boldsymbol{W}(\boldsymbol{X}-\boldsymbol{Y})\|_2^2 \right\}$
- $x^{k+1}_{i,j} = W_{i,j} \cdot x^k_{i,j} + (1-W_{i,j}) \cdot \widetilde{x}_{i,j}$



Connections with fixed effects model, $x_{i,j} = \sum_{k=1}^{s} f_{i,k} u_{j,k} + \varepsilon_{i,j}$ with $\varepsilon_{i,j} \sim \mathcal{N}(0, \sigma^2)$

and random effects model, $\boldsymbol{x}_i = \boldsymbol{\Gamma} \boldsymbol{z}_i + \boldsymbol{\varepsilon}_i$ with $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$ and $\boldsymbol{z}_i \sim \mathcal{N}(0, \mathbb{I})$

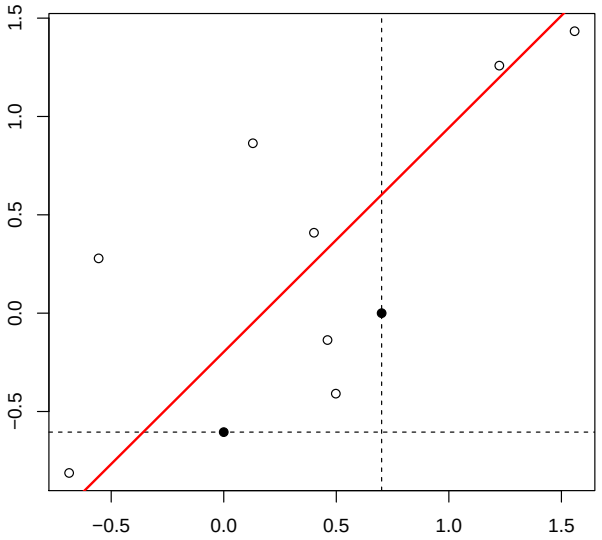
Missing Values : Single Imputation

Iterative PCA

(i) if $x_{i,j}$ is missing, $W_{i,j}=0$,
$x^1_{i,j} = W_{i,j}\cdot x^0_{i,j} + (1-W_{i,j})\cdot 0$

(ii) for $k=1,2,\cdots$

- $\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\le s}{\text{argmin}} \left\{\|\boldsymbol{W}(\boldsymbol{X}-\boldsymbol{Y})\|_2^2\right\}$
- $x^{k+1}_{i,j} = W_{i,j}\cdot x^k_{i,j} + (1-W_{i,j})\cdot \widetilde{x}_{i,j}$



Connections with fixed effects model, $x_{i,j} = \sum_{k=1}^{s} f_{i,k}u_{j,k} + \varepsilon_{i,j}$ with $\varepsilon_{i,j}\sim\mathcal{N}(0,\sigma^2)$

and random effects model, $\boldsymbol{x}_i = \boldsymbol{\Gamma}\boldsymbol{z}_i + \boldsymbol{\varepsilon}_i$ with $\boldsymbol{\varepsilon}_i\sim\mathcal{N}(0,\sigma^2\mathbb{I})$ and $\boldsymbol{z}_i\sim\mathcal{N}(0,\mathbb{I})$

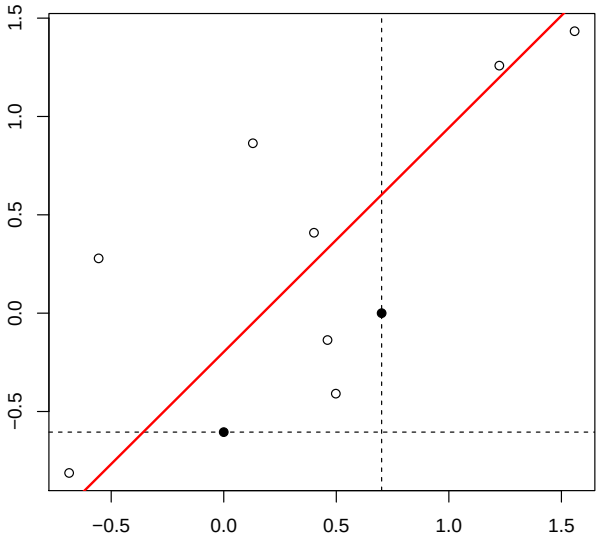
Missing Values : Single Imputation

Iterative PCA

(i) if $x_{i,j}$ is missing, $W_{i,j}=0$,
$x^1_{i,j} = W_{i,j}\cdot x^0_{i,j} + (1-W_{i,j})\cdot 0$

(ii) for $k=1,2,\cdots$

- $\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\le s}{\text{argmin}} \left\{\|\boldsymbol{W}(\boldsymbol{X}-\boldsymbol{Y})\|_2^2\right\}$
- $x^{k+1}_{i,j} = W_{i,j}\cdot x^k_{i,j} + (1-W_{i,j})\cdot \widetilde{x}_{i,j}$



Connections with fixed effects model, $x_{i,j} = \sum_{k=1}^{s} f_{i,k}u_{j,k} + \varepsilon_{i,j}$ with $\varepsilon_{i,j} \sim \mathcal{N}(0,\sigma^2)$

and random effects model, $\boldsymbol{x}_i = \boldsymbol{\Gamma}\boldsymbol{z}_i + \boldsymbol{\varepsilon}_i$ with $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0,\sigma^2\mathbb{I})$ and $\boldsymbol{z}_i \sim \mathcal{N}(0,\mathbb{I})$

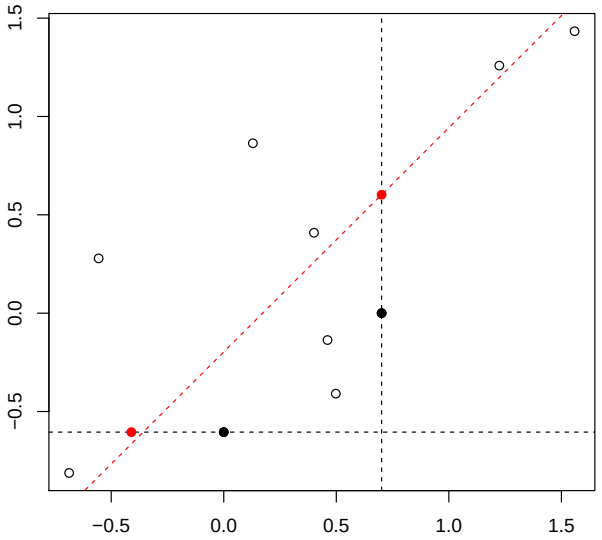
Missing Values : Single Imputation

Iterative PCA

(i) if $x_{i,j}$ is missing, $W_{i,j}=0$,
$x^1_{i,j} = W_{i,j}\cdot x^0_{i,j} + (1-W_{i,j})\cdot 0$

(ii) for $k=1,2,\cdots$

- $\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\le s}{\text{argmin}} \left\{\|\boldsymbol{W}(\boldsymbol{X}-\boldsymbol{Y})\|_2^2\right\}$
- $x^{k+1}_{i,j} = W_{i,j}\cdot x^k_{i,j} + (1-W_{i,j})\cdot \widetilde{x}_{i,j}$



Connections with fixed effects model, $x_{i,j} = \sum_{k=1}^{s} f_{i,k}u_{j,k} + \varepsilon_{i,j}$ with $\varepsilon_{i,j} \sim \mathcal{N}(0,\sigma^2)$

and random effects model, $\boldsymbol{x}_i = \boldsymbol{\Gamma}\boldsymbol{z}_i + \boldsymbol{\varepsilon}_i$ with $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0,\sigma^2\mathbb{I})$ and $\boldsymbol{z}_i \sim \mathcal{N}(0,\mathbb{I})$

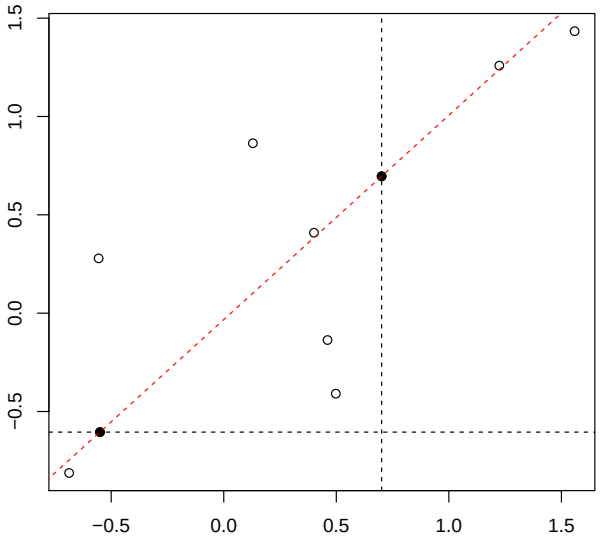
Missing Values : Single Imputation

Iterative PCA

(i) if $x_{i,j}$ is missing, $W_{i,j}=0$,
$x^1_{i,j} = W_{i,j}\cdot x^0_{i,j} + (1-W_{i,j})\cdot 0$

(ii) for $k=1,2,\cdots$

- $\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y})\le s}{\text{argmin}} \left\{\|\boldsymbol{W}(\boldsymbol{X}-\boldsymbol{Y})\|_2^2\right\}$
- $x^{k+1}_{i,j} = W_{i,j}\cdot x^k_{i,j} + (1-W_{i,j})\cdot \widetilde{x}_{i,j}$



Connections with fixed effects model, $x_{i,j} = \sum_{k=1}^{s} f_{i,k}u_{j,k} + \varepsilon_{i,j}$ with $\varepsilon_{i,j} \sim \mathcal{N}(0,\sigma^2)$

and random effects model, $\boldsymbol{x}_i = \boldsymbol{\Gamma}\boldsymbol{z}_i + \boldsymbol{\varepsilon}_i$ with $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(0,\sigma^2\mathbb{I})$ and $\boldsymbol{z}_i \sim \mathcal{N}(0,\mathbb{I})$

Missing Values : Single Imputation

The iterative PCA is simply using EM on fixed effects model,

$$x_{i,j} = \sum_{i=1}^{s} f_{i,j} u_{i,j} + \varepsilon_{i,j} \text{ with } \varepsilon_{i,j} \sim \mathcal{N}(0, \sigma^2)$$

$$\underbrace{\boldsymbol{X}}_{n \times p} = \underbrace{\boldsymbol{F}}_{n \times s} \underbrace{\boldsymbol{U}}_{p \times s}^{\top}$$

Log-likelihood is here

$$\log \mathcal{L}(\boldsymbol{F}, \boldsymbol{u}, \sigma^2) = -\frac{np}{2} \log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2} \|\boldsymbol{X} - \boldsymbol{F}\boldsymbol{u}^{\top}\|^2$$

E step : compute $\mathbb{E}\left[X_{i,j} \middle| \boldsymbol{X}, \boldsymbol{F}_k, \boldsymbol{U}_k, \sigma_k^2\right]$ (imputation)

M step : maximize the log-likelihood

$$\boldsymbol{U}_{k+1} = \widehat{\boldsymbol{X}}_k^{\top} \boldsymbol{F}_k \left(\boldsymbol{F}_k^{\top} \boldsymbol{F}_k\right)^{-1} \text{ and } \boldsymbol{F}_{k+1} = \widehat{\boldsymbol{X}}_k \boldsymbol{U}_k \left(\boldsymbol{U}_k^{\top} \boldsymbol{U}_k\right)^{-1}$$

Missing Values : Single Imputation

One can use regularized iterative PCA. So far, we used (SVD) $\widetilde{\boldsymbol{X}}_s\boldsymbol{U}_s\boldsymbol{\Lambda}_s^{1/2}\boldsymbol{V}_s^\top$

$$\widehat{X}_{i,j} = \sum_{k=1}^{s} \sqrt{\lambda_k} U_{i,k} V_{j,k}$$

Following Efron & Morris (1972, Limiting the Risk of Bayes and Empirical Bayes Estimators) consider a shrinkage version

$$\widetilde{X}_{i,j} = \sum_{k=1}^{s} \left(\frac{\lambda_k - \sigma^2}{\lambda_k} \right) \sqrt{\lambda_k} U_{i,k} V_{j,k} = \sum_{k=1}^{s} \left(\sqrt{\lambda_k} - \frac{\sigma^2}{\lambda_k} \right) U_{i,k} V_{j,k}$$

where $\sigma^2 = \dfrac{n[\lambda s + 1 + \cdots + \lambda_p]}{np - p - ns - ps + s^2 + s}$

See package `missMDA`

Missing Values : Single Imputation

One can use soft-thresholding PCA. Following Hastie & Mazumber (2015, Matrix Completion and Low-Rank SVD)

$$\widetilde{X}_{i,j} = \sum_{k=1}^{s} \left(\sqrt{\lambda_k} - \lambda \right)_+ U_{i,k} V_{j,k}$$

solution of

$$\widetilde{\boldsymbol{X}}_s = \underset{\boldsymbol{Y},\ \text{rank}(\boldsymbol{Y}) \leq s}{\text{argmin}} \left\{ \|\boldsymbol{W}(\boldsymbol{X} - \boldsymbol{Y})\|_2^2 + \lambda \|\boldsymbol{Y}\|_\star \right\}$$

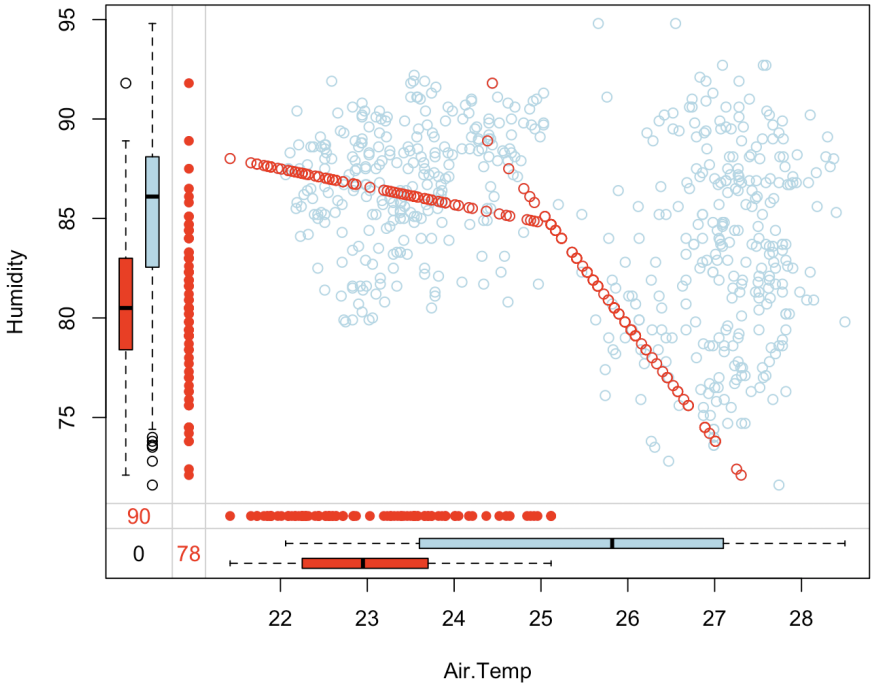
where the penalty is based on the nuclear norm (sum of the singular values).

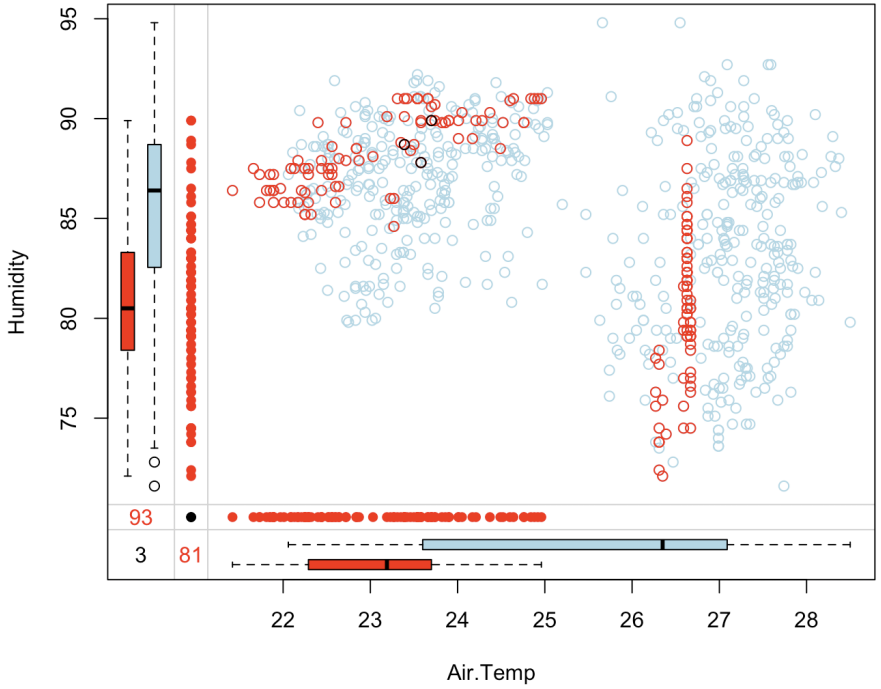
Complicated to select λ...

See package `softImpute`

Missing Values : Single Imputation

One can also use k-nearest neigbors





with `missMDA::imputePCA(y,ncp=1)` and `VIM::kNN(y,k=5)`

Missing Values : Multiple Imputation

It aims to allow for the uncertainty about the missing data by creating several different plausible imputed data sets (via Sterne *et al.* (2009, Multiple imputation for missing data)

Reference, Rubin (2007, Multiple imputation for nonresponse in surveys)

The idea is to generate N possible values for each missing value, see Honaker, King & Blackwell (2010, Amelia) and library `Amelia` using boostrap samples or van Buuren (2018, Multivariate Imputation by Chained Equations) with `mice` using bootstrap and regression

The idea of imputation is both seductive and dangerous. It is seductive because it can lull the user into the pleasurable state of believing that the data are complete after all, and it is dangerous because it lumps together situations where the problem is sufficiently minor that it can be legitimately handled in this way and situations where standard estimators applied to the real and imputed data have substantial biases Dempster & Rubin (1983, Incomplete Data in Sample Surveys)

Missing Values : Gaussian process regression (and krigging)

Extrapolation or interpolation ?

x	y
1	y_1
2	y_2
3	?

x	y
1	y_1
2	?
3	y_3

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \sigma_{1,3} \\ \sigma_{2,1} & \sigma_{2,2} & \sigma_{2,3} \\ \sigma_{3,1} & \sigma_{3,2} & \sigma_{3,3} \end{bmatrix}\right)$$

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_\star \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}_\star \\ \boldsymbol{\Sigma}_\star & \boldsymbol{\Sigma}_{\star\star} \end{bmatrix}\right)$$

$$(\boldsymbol{y}_\star|\boldsymbol{y}) \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{y}}, \boldsymbol{\Sigma}_{\boldsymbol{y}}) \text{ where } \begin{cases} \boldsymbol{\mu}_{\boldsymbol{y}} = \boldsymbol{\Sigma}_\star^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{y} \\ \boldsymbol{\Sigma}_{\boldsymbol{y}} = \boldsymbol{\Sigma}_{\star\star} - \boldsymbol{\Sigma}_\star^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}_\star \end{cases}$$

see Roberts *et al.* (2012, Gaussian Processes for Time Series) or Rasmussen & Williams (2006, Gaussian Processes for Machine Learning)