Missing Value Imputation via Gaussian Copula

Yuxuan Zhao

ORIE 4741, Nov 11 2021

Why I am here today?

- In your course project, it is very likely to run into missing data.

Why I am here today?

- In your course project, it is very likely to run into missing data.
- The missing data imputation method I introduce today can be simply used without selecting hyperparameters.
- The software can be easily installed and used.

Why I am here today?

- In your course project, it is very likely to run into missing data.
- The missing data imputation method I introduce today can be simply used without selecting hyperparameters.
- The software can be easily installed and used.
- We want to know if our method works well for your problem!

Table of Contents

Let's first see a general social survey dataset

	AGE	DEGREE	RINCOME	CLASS_	SATJOB	WEEKSWRK	HAPPY	HEALTH	SOCBAR
1	53.0	3	12.0	3.0	2.0	52.0	1.0	1.0	NaN
2	26.0	3	12.0	3.0	2.0	52.0	1.0	1.0	NaN
3	59.0	1	NaN	2.0	1.0	13.0	3.0	2.0	2.0
4	56.0	3	9.0	3.0	1.0	52.0	1.0	NaN	5.0
5	74.0	3	NaN	3.0	NaN	0.0	1.0	1.0	NaN

Figure 1: 2538 participants and 9 questions. 18.2% entries are missing in total.

Example variables

- **Subjective class identification**: If you were asked to use one of four names for your social class, which would you say you belong in: the lower class, the working class, the middle class, or the upper class?
- **General happiness**: Taken all together, how would you say things are these days–would you say that you are very happy, pretty happy, or not too happy?
- **Respondents income**: In which of these groups did your earnings from (OCCUPATION IN OCC) for last year–[the previous year]–fall? That is, before taxes or other deductions. Just tell me the letter.
- **Weeks r. worked last year**: In [the previous year] how many weeks did you work either full-time or part-time not counting work around the house–include paid vacations and sick leave?

Recap: GLRM imputes mixed data better than PCA

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{k \times p}$ such that XW approximates $Y \in \mathbb{R}^{n \times p}$ well:

$$\text{minimize} \sum_{(i,j) \in \Omega} \ell_j \left(Y_{ij}, {x_i}^T w_j \right) + \sum_{i=1}^{n} r_i \left(x_i \right) + \sum_{j=1}^{d} \tilde{r}_j \left(w_j \right)$$

- ℓ_j can vary for different j.
- The regularizer for row r_i and column $\tilde{r}_j$ can vary.

Recap: GLRM imputes mixed data better than PCA

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{k \times p}$ such that XW approximates $Y \in \mathbb{R}^{n \times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j\left(Y_{ij}, {x_i}^T w_j\right) + \sum_{i=1}^{n} r_i\left(x_i\right) + \sum_{j=1}^{d} \tilde{r}_j\left(w_j\right)$$

- ℓ_j can vary for different j.
- The regularizer for row r_i and column $\tilde{r}_j$ can vary.

Great flexibility usually means many choices to make...

GLRM: practical consideration

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n\times k}$ and $W \in \mathbb{R}^{k\times p}$ such that XW approximates $Y \in \mathbb{R}^{n\times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j\left(Y_{ij}, x_i^T w_j\right) + \sum_{i=1}^{n} r_i\left(x_i\right) + \sum_{j=1}^{d} \tilde{r}_j\left(w_j\right)$$

- What ℓ_j to choose?
- How to assign weights to ℓ_j when columns have different scales?
- What regularizer $r_i, \tilde{r}_j$ to use?

GLRM: practical consideration

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{k \times p}$ such that XW approximates $Y \in \mathbb{R}^{n \times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j\left(Y_{ij}, x_i^T w_j\right) + \sum_{i=1}^{n} r_i\left(x_i\right) + \sum_{j=1}^{d} \tilde{r}_j\left(w_j\right)$$

- What ℓ_j to choose?
- How to assign weights to ℓ_j when columns have different scales?
- What regularizer $r_i, \tilde{r}_j$ to use?

And there are tuning parameters...

GLRM: practical consideration

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n\times k}$ and $W \in \mathbb{R}^{k\times p}$ such that XW approximates $Y \in \mathbb{R}^{n\times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j \left(Y_{ij}, x_i^T w_j\right) + \sum_{i=1}^{n} r_i\left(x_i\right) + \sum_{j=1}^{d} \tilde{r}_j\left(w_j\right)$$

- How to choose the rank k?
- If setting r_i and $\tilde{r}_j$ as quadratic regularization with parameter λ, how to choose λ?

GLRM: practical consideration

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n\times k}$ and $W \in \mathbb{R}^{k\times p}$ such that XW approximates $Y \in \mathbb{R}^{n\times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j\left(Y_{ij}, x_i^T w_j\right) + \sum_{i=1}^{n} r_i\left(x_i\right) + \sum_{j=1}^{d} \tilde{r}_j\left(w_j\right)$$

- How to choose the rank k?
- If setting r_i and $\tilde{r}_j$ as quadratic regularization with parameter λ, how to choose λ?
- Need to search over two-dimensional grid.

GLRM: practical consideration

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{k \times p}$ such that XW approximates $Y \in \mathbb{R}^{n \times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j\left(Y_{ij}, x_i^T w_j\right) + \sum_{i=1}^{n} r_i(x_i) + \sum_{j=1}^{d} \tilde{r}_j(w_j)$$

- How to choose the rank k?
- If setting r_i and $\tilde{r}_j$ as quadratic regularization with parameter λ, how to choose λ?
- Need to search over two-dimensional grid.

Is the problem just about computation?

GLRM: low rank assumption

Generalized low rank model: find low rank matrix $X \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{k \times p}$ such that XW approximates $Y \in \mathbb{R}^{n \times p}$ well:

$$\text{minimize} \sum_{(i,j)\in\Omega} \ell_j\left(Y_{ij}, x_i^T w_j\right) + \sum_{i=1}^{n} r_i\left(x_i\right) + \sum_{j=1}^{d} \tilde{r}_j\left(w_j\right)$$

- Only works well when Y can be approximated by low rank matrix.
- Big data (large n and large p) usually have low rank structure.
  - Movie rating datasets: many movies and many users
- Long skinny data (large n and small p) usually does not have low rank structure.
  - Social survey data: many participants, few questions.

Get over the low rank assumption

- Large n allows learning more complex variable dependence than the low rank structure.
- Statistical dependence structure: model the joint distribution
  - Gaussian distribution for quantitative vector

Get over the low rank assumption

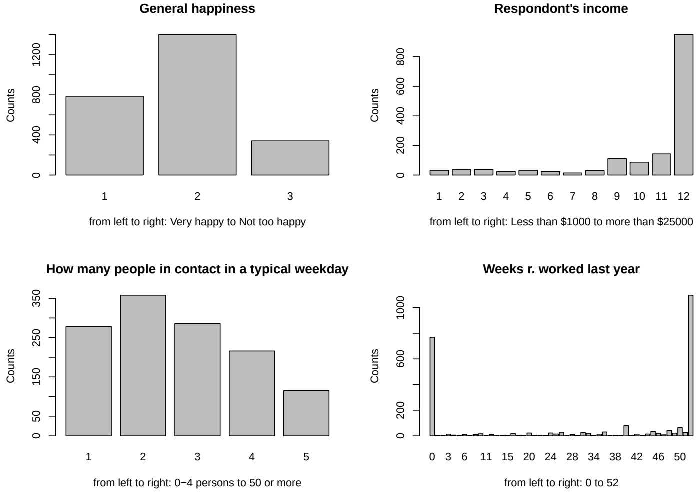
- Large n allows learning more complex variable dependence than the low rank structure.
- Statistical dependence structure: model the joint distribution
  - Gaussian distribution for quantitative vector
    1. All 1-dimensional marginals are Gaussian
    2. The joint p-dimensional distribution is multivariate Gaussian

Get over the low rank assumption

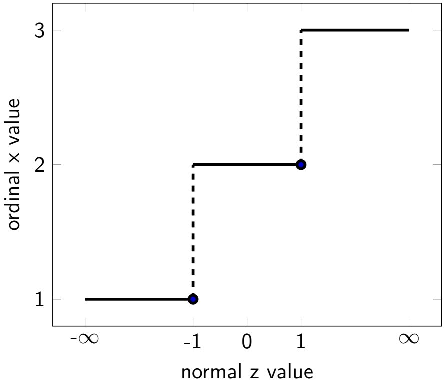
- Large n allows learning more complex variable dependence than the low rank structure.
- Statistical dependence structure: model the joint distribution
  - Gaussian distribution for quantitative vector
    1. All 1-dimensional marginals are Gaussian
    2. The joint p-dimensional distribution is multivariate Gaussian

First, can we use 1-dimensional Gaussian to model ordinal/binary variable?

Histograms for some GSS variables



Generate ordinal data by thresholding Gaussian variable



- Select thresholds to ensure desired class proportion.
- A mapping between ordinal levels and intervals.
- $f(z) = x$ for $z \in [a_x, a_{x+1})$ or $f^{-1}(x) = [a_x, a_{x+1})$.

Estimated thresholds for some GSS variables

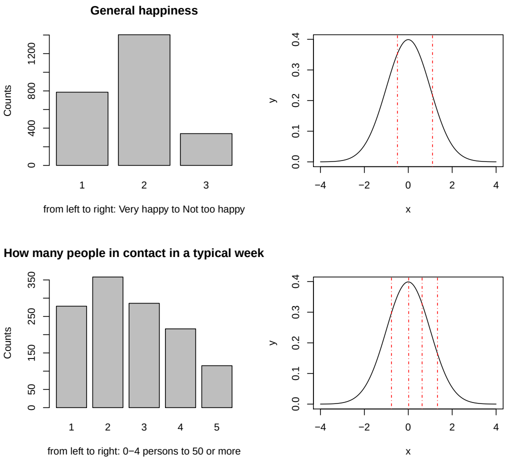


Figure 2: Red vertical lines indicate estimated thresholds.

Table of Contents

Gaussian copula model for mixed data

We say $x = (x_1, \ldots, x_p)$ follows the **Gaussian copula model** if

- **marginals:** $x = \mathbf{f}(z)$ for $\mathbf{f} = (f_1, \ldots, f_p)$ entrywise monotonic,

$$x_j = f_j(z_j), \qquad j = 1, \ldots, p$$

- **copula:** $z \sim \mathcal{N}(0, \Sigma)$ with correlation matrix Σ

Gaussian copula model for mixed data

We say $x = (x_1, \ldots, x_p)$ follows the **Gaussian copula model** if

- **marginals:** $x = \mathbf{f}(z)$ for $\mathbf{f} = (f_1, \ldots, f_n)$ entrywise monotonic,

$$x_j = f_j(z_j), \qquad j = 1, \ldots, p$$

- **copula:** $z \sim \mathcal{N}(0, \Sigma)$ with correlation matrix Σ
- Estimate f_j to match the observed empirical distribution
- Estimate Σ through an EM algorithm

Given parameter estimate, imputation is easy

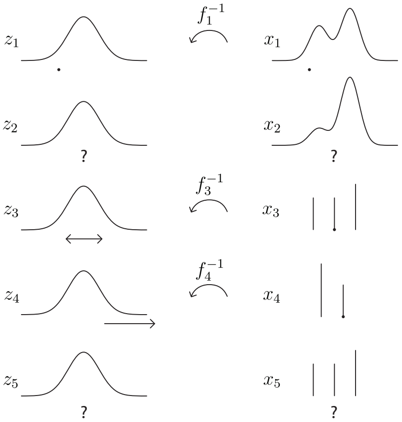


Figure 3: Curves indicate density and dots mark the observation.

Given parameter estimate, imputation is easy

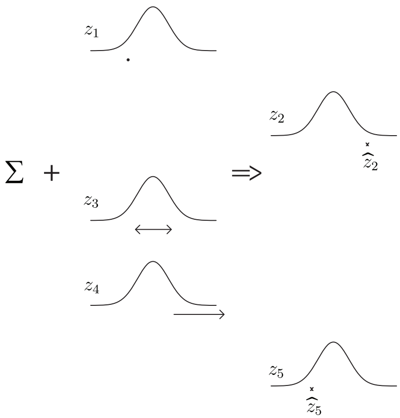


Figure 4: Curves indicate density and crosses mark the prediction.

Given parameter estimate, imputation is easy

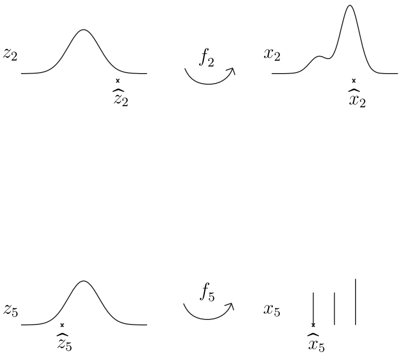


Figure 5: Curves indicate density and crosses mark the prediction.

Given parameters, imputation is easy

- observed entries $\mathbf{x}_\mathcal{O}$ of new row $\mathbf{x} \in \mathbb{R}^p$, $\mathcal{O} \subset \{1, \dots, p\}$
- missing entries $\mathcal{M} = \{1, \dots, p\} \setminus \mathcal{O}$
- marginals $\mathbf{f} = (\mathbf{f}_\mathcal{O}, \mathbf{f}_\mathcal{M})$ and copula correlation matrix Σ
- the truncated region: $\mathbf{z}_\mathcal{O} \in \mathbf{f}_\mathcal{O}^{-1}(\mathbf{x}_\mathcal{O}) := \prod_{j\in\mathcal{O}} f_j^{-1}(x_j)$

impute missing entries using normality of $\mathbf{z}_\mathcal{M}$:

- latent missing $\mathbf{z}_\mathcal{M}$ are normal given $\mathbf{z}_\mathcal{O}$:

$$\mathbf{z}_\mathcal{M} | \mathbf{z}_\mathcal{O} \sim \mathcal{N}(\Sigma_{\mathcal{M},\mathcal{O}}\Sigma_{\mathcal{O},\mathcal{O}}^{-1}\mathbf{z}_\mathcal{O}, \Sigma_{\mathcal{M},\mathcal{M}} - \Sigma_{\mathcal{M},\mathcal{O}}\Sigma_{\mathcal{O},\mathcal{O}}^{-1}\Sigma_{\mathcal{O},\mathcal{M}})$$

Given parameters, imputation is easy

- observed entries $\mathbf{x}_\mathcal{O}$ of new row $\mathbf{x} \in \mathbb{R}^p$, $\mathcal{O} \subset \{1, \ldots, p\}$
- missing entries $\mathcal{M} = \{1, \ldots, p\} \setminus \mathcal{O}$
- marginals $\mathbf{f} = (\mathbf{f}_\mathcal{O}, \mathbf{f}_\mathcal{M})$ and copula correlation matrix Σ
- the truncated region: $\mathbf{z}_\mathcal{O} \in \mathbf{f}_\mathcal{O}^{-1}(\mathbf{x}_\mathcal{O}) := \prod_{j\in\mathcal{O}} f_j^{-1}(x_j)$

impute missing entries using normality of $\mathbf{z}_\mathcal{M}$:

- latent missing $\mathbf{z}_\mathcal{M}$ are normal given $\mathbf{z}_\mathcal{O}$:

$$\mathbf{z}_\mathcal{M} | \mathbf{z}_\mathcal{O} \sim \mathcal{N}(\Sigma_{\mathcal{M},\mathcal{O}}\Sigma_{\mathcal{O},\mathcal{O}}^{-1}\mathbf{z}_\mathcal{O}, \Sigma_{\mathcal{M},\mathcal{M}} - \Sigma_{\mathcal{M},\mathcal{O}}\Sigma_{\mathcal{O},\mathcal{O}}^{-1}\Sigma_{\mathcal{O},\mathcal{M}})$$

- predict with mean

$$\hat{\mathbf{z}}_\mathcal{M} = \Sigma_{\mathcal{M},\mathcal{O}}\Sigma_{\mathcal{O},\mathcal{O}}^{-1}\mathbb{E}[\mathbf{z}_\mathcal{O} | \mathbf{z}_\mathcal{O} \in \mathbf{f}_\mathcal{O}^{-1}(\mathbf{x}_\mathcal{O})]$$

- map back to observed space $\hat{\mathbf{x}}_\mathcal{M} = \mathbf{f}_\mathcal{M}(\hat{\mathbf{z}}_\mathcal{M})$

Multiple imputation

When imputation is the intermediate step to learn some parameter θ, e.g. linear coefficients, on imputed complete dataset:

1. Generate m different imputed datasets $X^{(1)}, \ldots, X^{(m)}$.
2. For each imputed dataset $X^{(j)}$, learn the desired model parameter $\hat{\theta}^{(j)}$ for $j = 1, \ldots, m$.
3. Combine all estiamtes into one: $\hat{\theta} = \frac{\sum_{j=1}^{m} \hat{\theta}^{(j)}}{m}$.

Given parameters, imputation is easy

- observed entries $\mathbf{x}_{\mathcal{O}}$ of new row $\mathbf{x} \in \mathbb{R}^p$, $\mathcal{O} \subset \{1, \ldots, p\}$
- missing entries $\mathcal{M} = \{1, \ldots, p\} \setminus \mathcal{O}$
- marginals $\mathbf{f} = (\mathbf{f}_{\mathcal{O}}, \mathbf{f}_{\mathcal{M}})$ and copula correlation matrix Σ
- the truncated region: $\mathbf{z}_{\mathcal{O}} \in \mathbf{f}_{\mathcal{O}}^{-1}(\mathbf{x}_{\mathcal{O}}) := \prod_{j \in \mathcal{O}} f_j^{-1}(x_j)$

impute missing entries using normality of $\mathbf{z}_{\mathcal{M}}$:

- latent missing $\mathbf{z}_{\mathcal{M}}$ are normal given $\mathbf{z}_{\mathcal{O}}$:

$$\mathbf{z}_{\mathcal{M}} | \mathbf{z}_{\mathcal{O}} \sim \mathcal{N}(\Sigma_{\mathcal{M},\mathcal{O}} \Sigma_{\mathcal{O},\mathcal{O}}^{-1} \mathbf{z}_{\mathcal{O}}, \Sigma_{\mathcal{M},\mathcal{M}} - \Sigma_{\mathcal{M},\mathcal{O}} \Sigma_{\mathcal{O},\mathcal{O}}^{-1} \Sigma_{\mathcal{O},\mathcal{M}})$$

- Sample $\mathbf{z}_{\mathcal{M}}^{(i)}$ from the above distribution for $i = 1, .., m$.
- map back to observed space $\hat{\mathbf{x}}_{\mathcal{M}}^{(i)} = \mathbf{f}_{\mathcal{M}}(\hat{\mathbf{z}}_{\mathcal{M}}^{(i)})$ for $i = 1, .., m$.

Table of Contents

Check out our Github page

- Python package
`https://github.com/udellgroup/GaussianCopulaImp`
- Single line installment: `pip install GaussianCopulaImp`
- More tutorials on multiple imputation, accelerating the algorithm for large datasets, etc.