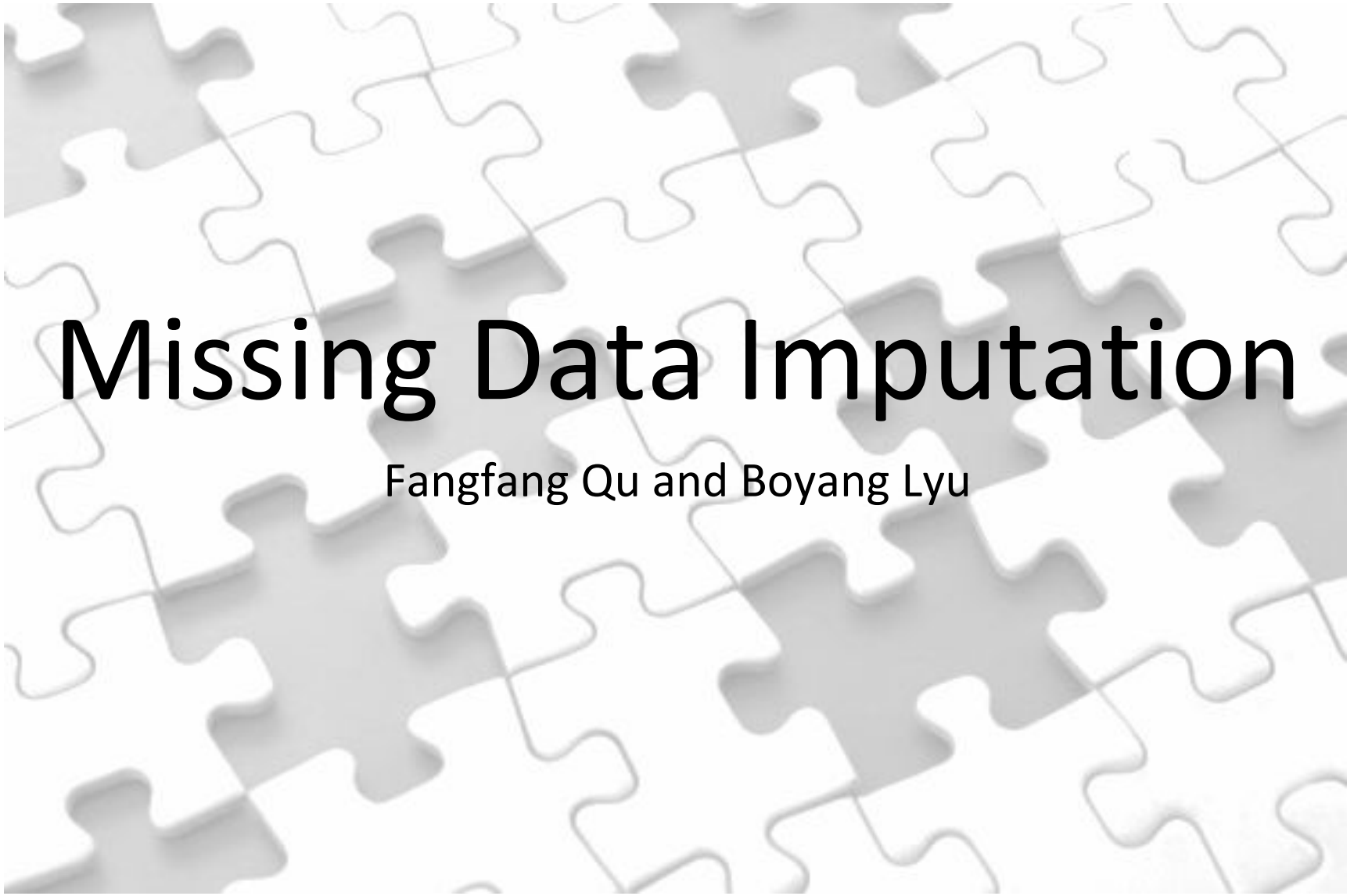


Missing Data Imputation

Fangfang Qu and Boyang Lyu

Outline

- Background
- Method
- Performance comparison
- Future work & reflection

Missing data

- Why data goes missing?
  - Non-responses in questionnaire, e.g. private items income, ages
  - High finical and time cost to generate/collect all possible data
  - Data collection is done improperly by researchers
- Why missing data matters?
  - Inaccurate prediction or classification
- Solutions?
  - Deletion: discard samples with incomplete values
  - Imputation: replace the missing values with an estimate

Data imputation

- Why data imputation rather than deletion?
  - Won’t work for a large missing rate
  - Introduce biases when missing data is related to certain variables(like ages)

- Methods
  - Mean replacement
  - Training an ML model then do prediction
  - Popular models: KNN, Autoencoder, Variational-Autoencoder, PCA and so on

Main Models

- Mean replacement
- KNN
- Autoencoder
- Variational Autoencoder

Baseline method - Mean replacement

- Algorithm:
  - Ignore all missing data
  - Calculate the mean of the remaining data with respect to each feature
  - Fill in blanks with the calculated mean

- Pros and Cons:
  - Fast and simple
  - Reduce the variance of dataset

Comparison - KNN

Implementation:

- Define a Masked Euclidean distance function:

$$Dist(x1, x2) = \frac{\#all\ features}{\#remaining\_features} * L2(remaining\_features)$$

	col_1	col_2	$col3$	
x1	1	nan	1	← incomplete
x2	2	3	4	← complete

- Each missing feature is imputed as the average of their K neighbors

Hyperparameter:

Number of K nearest number

Pros and Cons

- Retain data's structure
- Time consuming
- Meaningless when too many features are missing

Comparison - AE

- Implementation
  - Train a MLP autoencoder model with fully observed training data
  - Objective: *min*(*MSE + Weight-decay)*
  - Replace missing values with mean in test data
  - Input test data to the trained AE then make imputation
- Hyperparameters
  - Architecture, Batch size, learning rate
- Pros and Cons:
  - Can learn non-linear distribution
  - Can’t make good prediction when data doesn’t follow *Dist(training_data)*

Comparison - VAE

- Implementation:
  - Train a MLP variational autoencoder model with fully observed training data
  - Minimize Loss

$$L(\theta,\varphi;x)=-D_{KL}(q_\varphi(z|x) \,||\, p_\theta(z)) + E_{q\varphi(z|x)}\, logp(x|z)$$

  - Sample from latent distribution to generate z
  - Sample from the reconstructed data distribution to generate prediction

- Hyperparameters: Architecture and Batch size

Hypothesis

- VAE will win all competitions!
- KNN may work well for data with small size of features

Hypothesis

- VAE will win all competitions!
- KNN may work well for data with small size of features
- Reasons:
  - VAE fully specifies a generative distribution over the objects it is representing

Hypothesis

- VAE will win all competitions!
- KNN may work well for data with small size of features
- Reasons:
  - VAE fully specifies a generative distribution over the objects it is representing
  - KNN uses all information of the whole datasets

Hypothesis

- VAE will win all competitions!
- KNN may work well for data with small size of features
- Reasons:
  - VAE fully specifies a generative distribution over the objects it is representing
  - KNN uses all information of the whole datasets
  - But for high dimension data, distance is more likely to be equally large or small for KNN

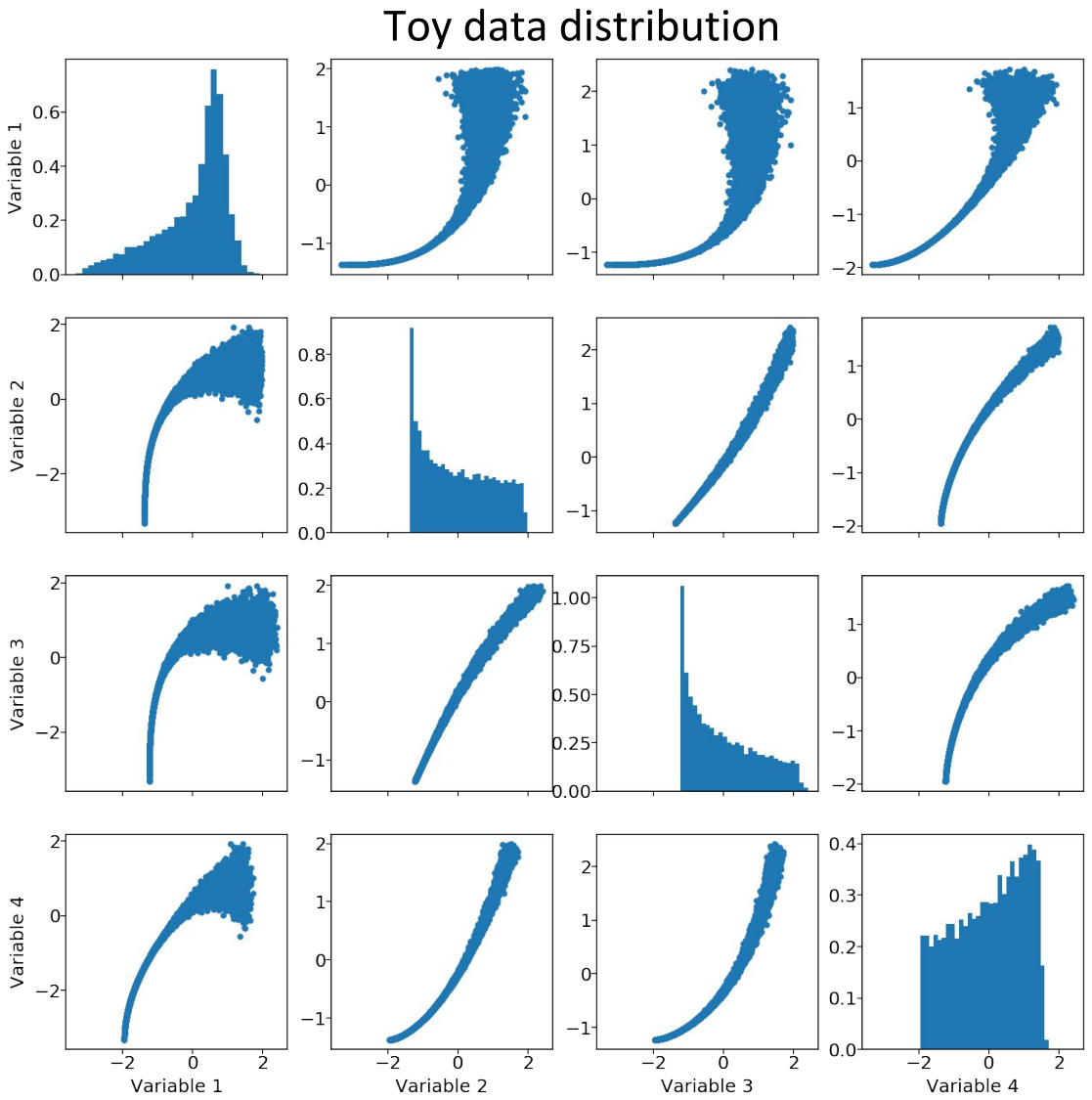
Hypothesis

- VAE will win all competitions!
- KNN may work well for data with small size of features
- Reasons:
  - VAE fully specifies a generative distribution over the objects it is representing
  - KNN uses all information of the whole datasets
  - But for high dimension data, distance is more likely to be equally large or small for KNN
- Measurable metric:
  - RMSE

Experiment – Toy dataset generation

- A dataset with 4 variables
- A base variable z ~ *U(0,1)*
- *r = sqrt(z)*
- η = 0.25 * π * z
- Four variables are then sampled from Gaussian distributions

Mean	Distribution
$x_{1,\mu}=r\times\cos\eta$	$x_1=\mathcal{N}(\mu=x_{1,\mu},\ \sigma^2=0.07\times z)$
$x_{2,\mu}=5r\times\sin\eta$	$x_2=\mathcal{N}(\mu=x_{2,\mu},\ \sigma^2=0.06\times z)$
$x_{3,\mu}=2r\times\tan\eta$	$x_3=\mathcal{N}(\mu=x_{2,\mu},\ \sigma^2=0.05\times z)$
$x_{4,\mu}=3\times exp(-\eta)$	$x_4=\mathcal{N}(\mu=x_{2,\mu},\ \sigma^2=0.04\times z)$

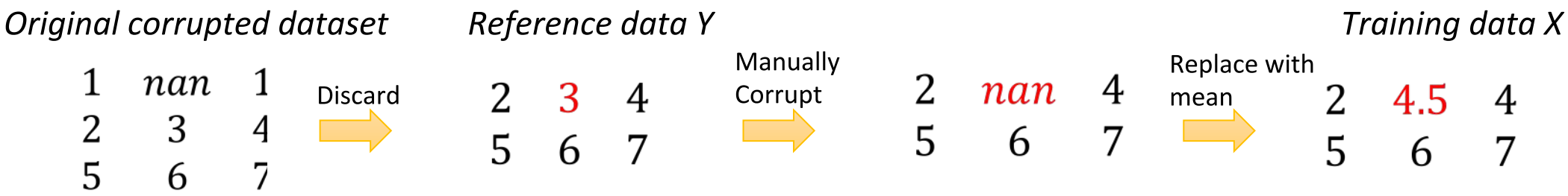


Experiment – Toy dataset corruption

- Two kinds of corrupted datasets
  - Lightly corrupted - 20% of the rows with missing value
  - Heavily corrupted - 80% of the rows with missing value
- The uncorrupted datasets were used as ground truth to calculate the accuracy of different imputation methods

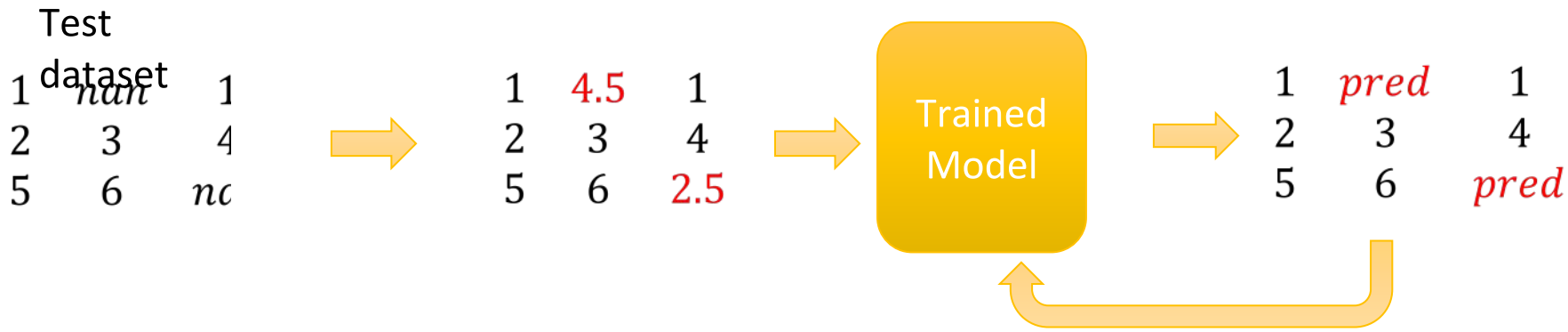
Experiment – Training

- Get a fully observed training dataset by removing all missing instances – call it Y
- Manually corrupt Y
  - replace some values with mean – X
- Use the data X as input, Y as truth(reference) to train AE and VAE model



Experiment – Test

1. Replace missing values with mean
2. Feed data to trained model, get the reconstructed values
3. Replace the missing values with the reconstructed values
4. Multiple imputation by iterate step 2 and 3 for 25 times

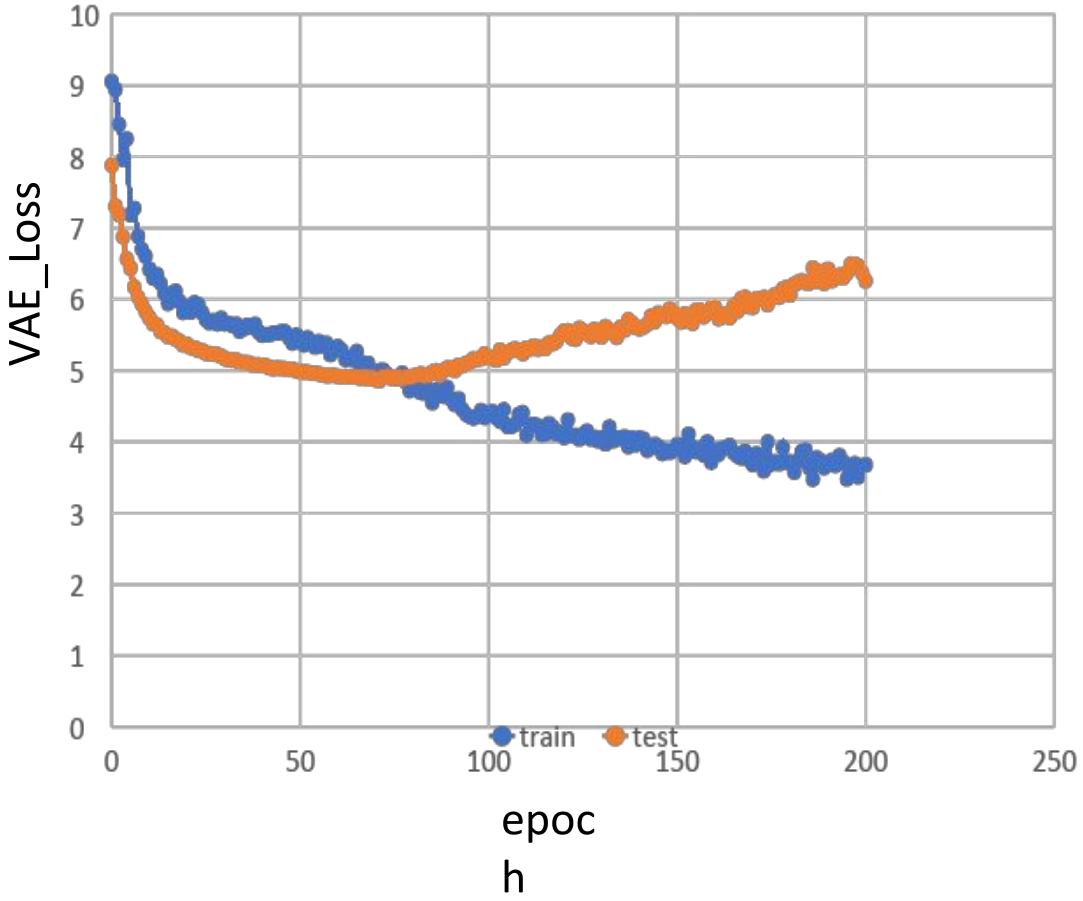


Experiment – Evaluation

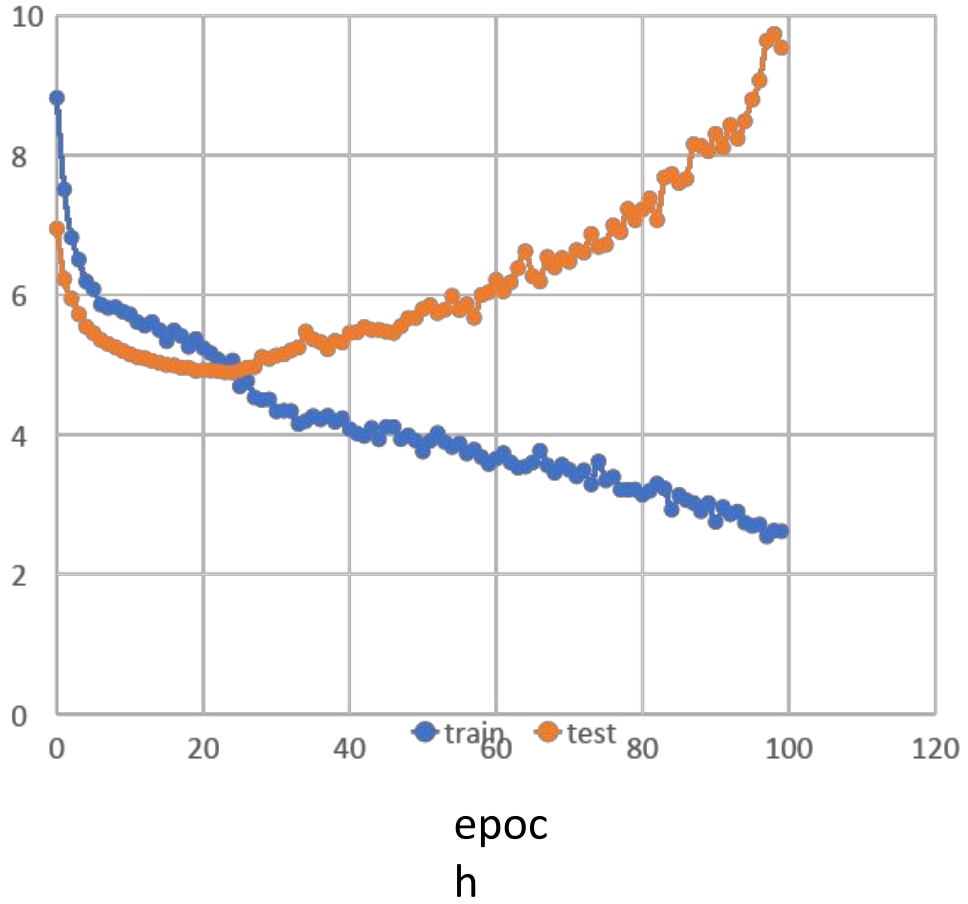
- L2 error between ground truth and imputed values
- ELBO for VAE to check overfit and underfit
- For MNIST, though we haven’t done all, we plan to examine the quality of reconstructed images by visualization

Experiment – VAE hyperparameter tuning

Learning rate
1e-4



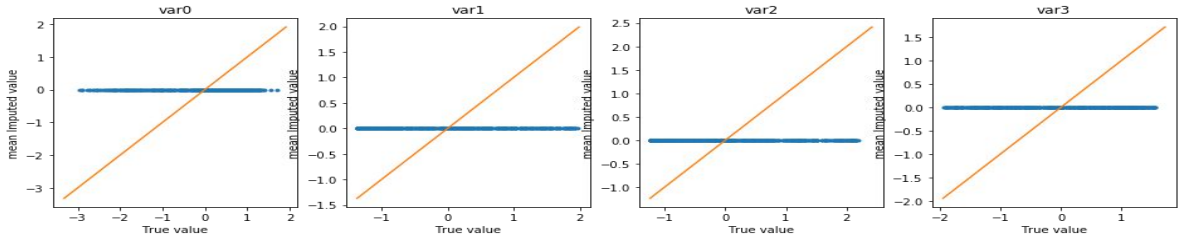
Learning rate 1e-3



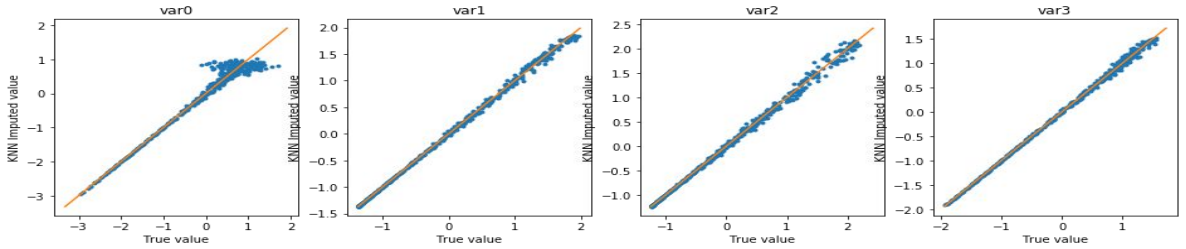
Result – imputation of lightly corrupted dataset

Method	RMSE (x1000) **Light**	heavy
Mean replacement	**22.39**	8.254
KNN imputation	**2.310**	2.117
AE imputation	**4.573**	1.747
VAE imputation	**2.411**	0.9601

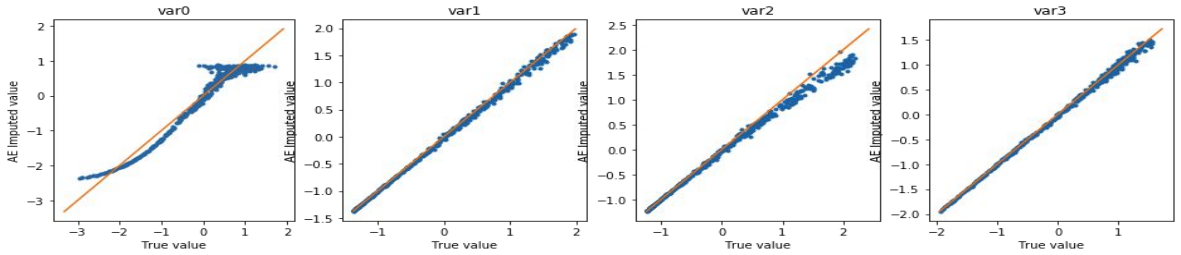
Mean



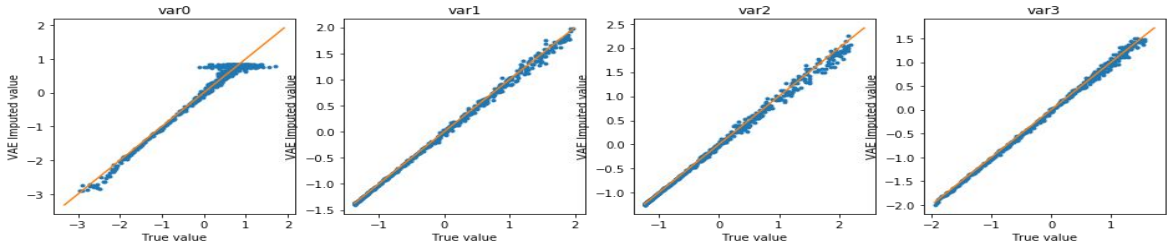
KNN



AE



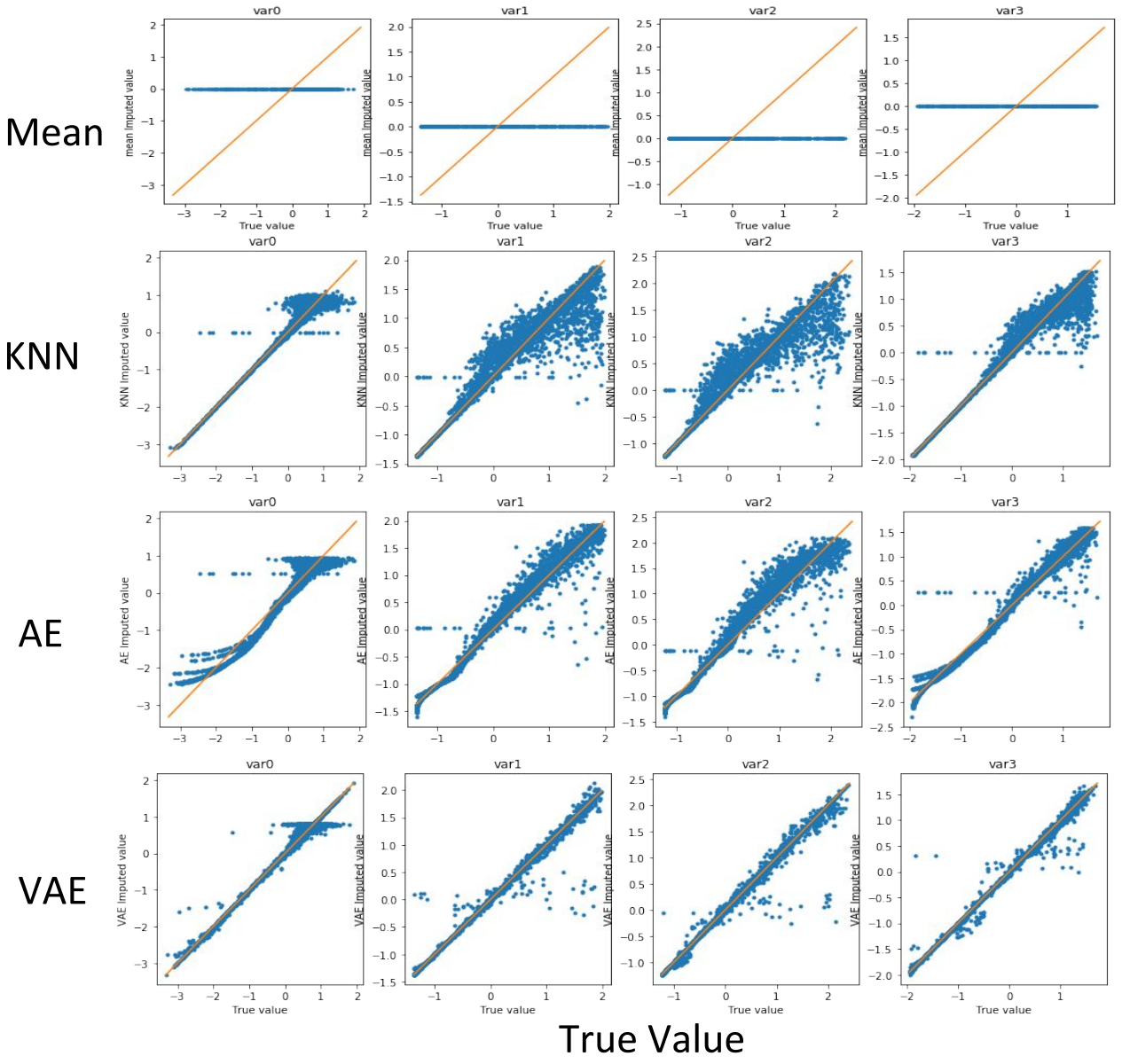
VAE



True Value

Result – imputation of heavily corrupted dataset

Method	RMSE (x1000) Light	**Heavy**
Mean replacement	22.39	**8.254**
KNN imputation	2.310	**2.117**
AE imputation	4.573	**1.747**
VAE imputation	2.411	**0.9601**

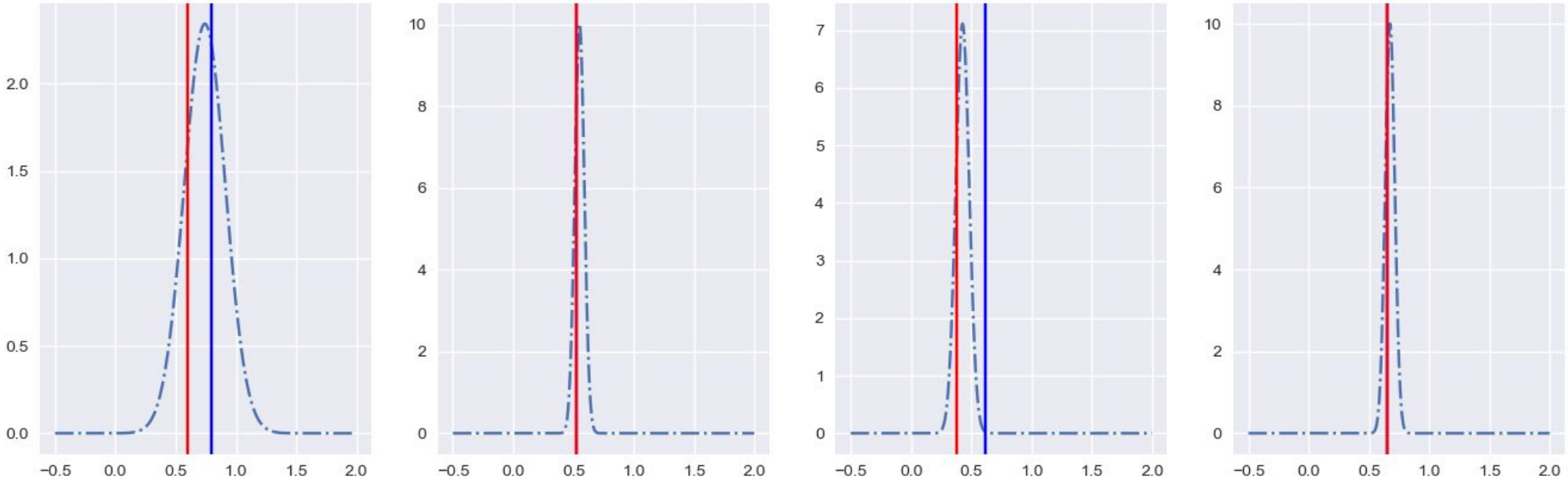


Result – imputed data distribution of VAE

distribution of imputed value by VAE

AE imputed value

True value



Imputed value

Future work

- Finish VAE, KNN part for MNIST dataset

- Tuning hyperparameters of all models for MNIST dataset

Reflection

- Don’t ignore simple method

- Adjust input according to objective

- Planning ahead and follow timeline

- Version control


THANK
YOU!