A Additional Derivations

A.1 Information reward approximation

In our paper, given the VAE model $p(\mathbf{x}|z)$ and a partial inference network $q(\mathbf{z}|\mathbf{x}_o)$, the experimental design problem is formulated as maximization of the information reward:

$$R(i,\mathbf{x}_o) = \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[D_{KL}(p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o))]$$

Where $p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o) = \int_{\mathbf{z}} p(\mathbf{x}_\phi|\mathbf{z})q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)$, $p(\mathbf{x}_\phi|\mathbf{x}_o) = \int_{\mathbf{z}} p(\mathbf{x}_\phi|\mathbf{z})q(\mathbf{z}|\mathbf{x}_o)$ and $q(\mathbf{z}|\mathbf{x}_o)$ are approximate condition distributions given by partial VAE models. Now we consider the problem of directly approximating $R(i,\mathbf{x}_o)$.

Applying the chain rule of KL-divergence, we have:

$$\begin{aligned}
&D_{KL}(p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o)) \\
&= D_{KL}(p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_o)) \\
&- \mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[D_{KL}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right],
\end{aligned}$$

Using again the KL-divergence chain rule on $D_{KL}(p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_o))$, we have:

$$\begin{aligned}
&D_{KL}(p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi,\mathbf{z}|\mathbf{x}_o)) \\
&= D_{KL}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)) + D_{KL}(p(\mathbf{x}_\phi|\mathbf{z},\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{z},\mathbf{x}_o)) \\
&= D_{KL}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)) + D_{KL}(p(\mathbf{x}_\phi|\mathbf{z})||p(\mathbf{x}_\phi|\mathbf{z})) \\
&= D_{KL}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)).
\end{aligned}$$

The KL-divergence term in the reward formula is now rewritten as follows,

$$\begin{aligned}
&D_{KL}(p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o)) \\
&= D_{KL}(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o)) \\
&- \mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[D_{KL}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right].
\end{aligned}$$

One can then plug in the partial VAE inference approximation:

$$\begin{aligned}
&p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o), \\
&p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o), \quad p(\mathbf{z}|\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_o)
\end{aligned}$$

Finally, the information reward is now approximated as:

$$\begin{aligned}
&R(i,\mathbf{x}_o) \\
&\approx \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[D_{KL}(q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_o))] \\
&- \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[D_{KL}(q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right] \\
&= \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[D_{KL}(q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_o))] \\
&- \mathbb{E}_{\mathbf{x}_\phi,\mathbf{x}_i \sim p(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)}\left[D_{KL}(q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right] = \hat{R}(i,\mathbf{x}_o).
\end{aligned}$$

This new objective tries to maximize the shift of belief on latent variables $\mathbf{z}$ by introducing $\mathbf{x}_i$, while penalizing the information that cannot be absorbed by $\mathbf{x}_\phi$ (by the penalty term $D_{KL}(q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))$). Moreover, it is more computationally efficient since one set of samples $\mathbf{x}_\phi,\mathbf{x}_i \sim p(\mathbf{x}_\phi,\mathbf{x}_i|\mathbf{x}_o)$ can be shared across different terms, and the KL-divergence between common parameterizations of encoder (such as Gaussians and normalizing flows) can be computed exactly without the need for approximate integrals. Note also that under approximation

$$p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o), \quad p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o), \quad p(\mathbf{z}|\mathbf{x}_o) \approx q(\mathbf{z}|\mathbf{x}_o)$$

, sampling $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)$ is approximated by $\mathbf{x}_i \sim \hat{p}(\mathbf{x}_i|\mathbf{x}_o)$, where $\hat{p}(\mathbf{x}_i|\mathbf{x}_o)$ is defined by the following process in Partial VAE. It is implemented by first sampling $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}_o)$, and then $\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{z})$. The same applies for $p(\mathbf{x}_i,\mathbf{x}_\phi|\mathbf{z})$.

B Additional Experimental Results

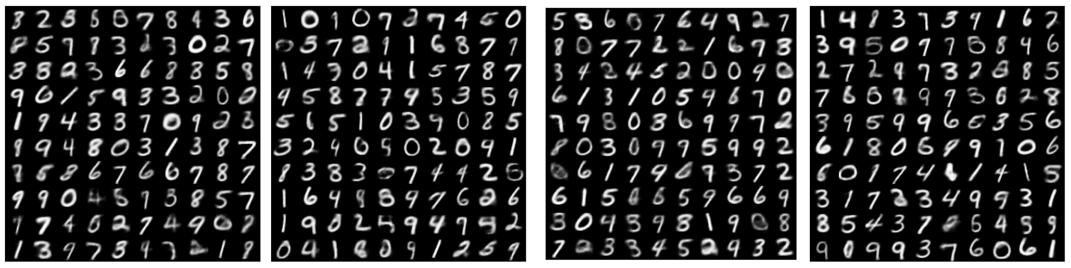
B.1 Image inpainting

B.1.1 Preprocessing and model details

For our MNIST experiment, we randomly draw 10% of the whole data to be our test set. Partial VAE models (ZI, ZI-m, PNP and PNs) share the same size of architecture with 20 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 20-200-500-500 fully connected neural network with ReLU activations (where D is the data dimension, $D = 784$). The inference nets (encoder) share the same structure of D-500-500-200-40 that maps the observed data into distributional parameters of the latent space. For the PN-based parameterizations, we use a 500 dimensional feature mapping h parameterized by a single layer neural network, and 20 dimensional ID vectors $\mathbf{e}_i$ (see Section 3.2) for each variable. We choose the symmetric operator g to be the basic summation operator.

During training, we apply Adam optimization (Kingma & Ba, 2015) with default hyperparameter setting, learning rate of 0.001 and a batch size of 100. We generate partially observed MNIST dataset by adding artificially missingness at random in the training dataset during training. We first draw a missing rate parameter from a uniform distribution $\mathscr{U}(0, 0.7)$ and randomly choose variables as unobserved. This step is repeated at each iteration. We train our models for 3K iterations.

B.1.2 Image generation of partial VAEs



(a) (b) (c) (d)

Figure 8: Random images generated using **(a)** naive zero imputing, **(b)** zero imputing with mask, **(c)** PN and **(d)** PNP, respectively.

B.2 UCI datasets

We applied EDDI on 6 UCI datasets; Boston Housing, Concrete compressive strength, energy efficiency, wine quality, Kin8nm, and Yacht Hydrodynamics. The variables of interest $\mathbf{x}_\phi$ are chosen to be the target variables of each UCI dataset in the experiment.

B.2.1 Preprocessing and model details

All data are normalized and then scaled between 0 and 1. For each of the 10 - in total- repetitions, we randomly draw 10% of the data to be our test set. Partial VAE models (ZI, ZI-m, PNP and PNs) share the same size of architecture with 10 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 10-50-100-D neural network with ReLU activations (where D is the data dimensions). The inference nets (encoder) share the same structure D-100-50-20 that maps the observed data into distributional parameters of the latent space. For the PN-based parameterizations, we further use a 20 dimensional feature mapping h parameterized by a single layer neural network and 10 dimensional ID vectors $\mathbf{e}_i$ (please refer to section 3.2) for each variable. We choose the symmetric operator g to be the basic summation operator.

As in the image inpainting experiment, we apply Adam optimization during training with default hyperparameter setting, and a batch size of 100 and ingest random missingness as before. We trained our models for 3K iterations.

During active learning, we draw 50 samples in order to estimate the expectation under $\mathbf{x}_\phi, \mathbf{x}_i \sim p(\mathbf{x}_\phi, \mathbf{x}_i | \mathbf{x}_o)$ in Equation (8). Other than information curves based on test RMSEs, we will also provide information curves based on test negative log likelihoods. This will be provided in Appendix B.2.4. Note that this test nllh of the target variable is also estimated using 50 samples of $\mathbf{x}_\phi \sim p(\mathbf{x}_\phi | \mathbf{x}_o)$. Then, we approximately compute the (expected) log predictive likelihood through $\log p(\mathbf{x}_\phi | \mathbf{x}_o) \approx \log \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{x}_\phi | \mathbf{z}_m)$, where $\mathbf{z}_m \sim q(\mathbf{z} | \mathbf{x}_o)$.

B.2.2 Statistical signifcant test results

In this section, we perform Wilcoxcon signed-rank significance test on the AUIC (RMSE-based) performance of different methods, to support our result in Table 3. Since Table 3 suggests that EDDI-PNP-Partial VAE is the best algorithm overall, we set EDDI-PNP-Partial VAE as default and perform Wilcoxcon test between EDDI-PNP-Partial VAE and all other 15 different settings, to see whether the improvement is significant. Table 6 displays the corresponding p-value for each test. It is obvious that in all 15 tests, the EDDI-PNP-Partial VAE results are significant (compared with the standard $\alpha = 0.05$ cutoff). This provides strong evidence that confirms our results in Table 3.

Table 6: p- values of Wilcoxon signed-rank test of EDDI-PNP vs. 11 other settings, on 6 UCI datasets, using AUIC (RMSE-based) as evaluation metric.

Method	ZI	ZI-m	PNP	PN
EDDI	$< 10^{-48}$	$<10^{-23}$	N/A	$<10^{-2}$
Random	0	0	0	0
Single best	0	$<10^{-13}$	$<10^{-2}$	$<10^{-4}$

B.2.3 Additional plots of PN, ZI and ZI-m on UCI datasets

Here we present additional plots of the RMSE information curves during active learning. Figure 9 presents the results for the Boston Housing, the Energy and the Wine datasets and for the three approaches, i.e. PN, ZI and masked ZI.

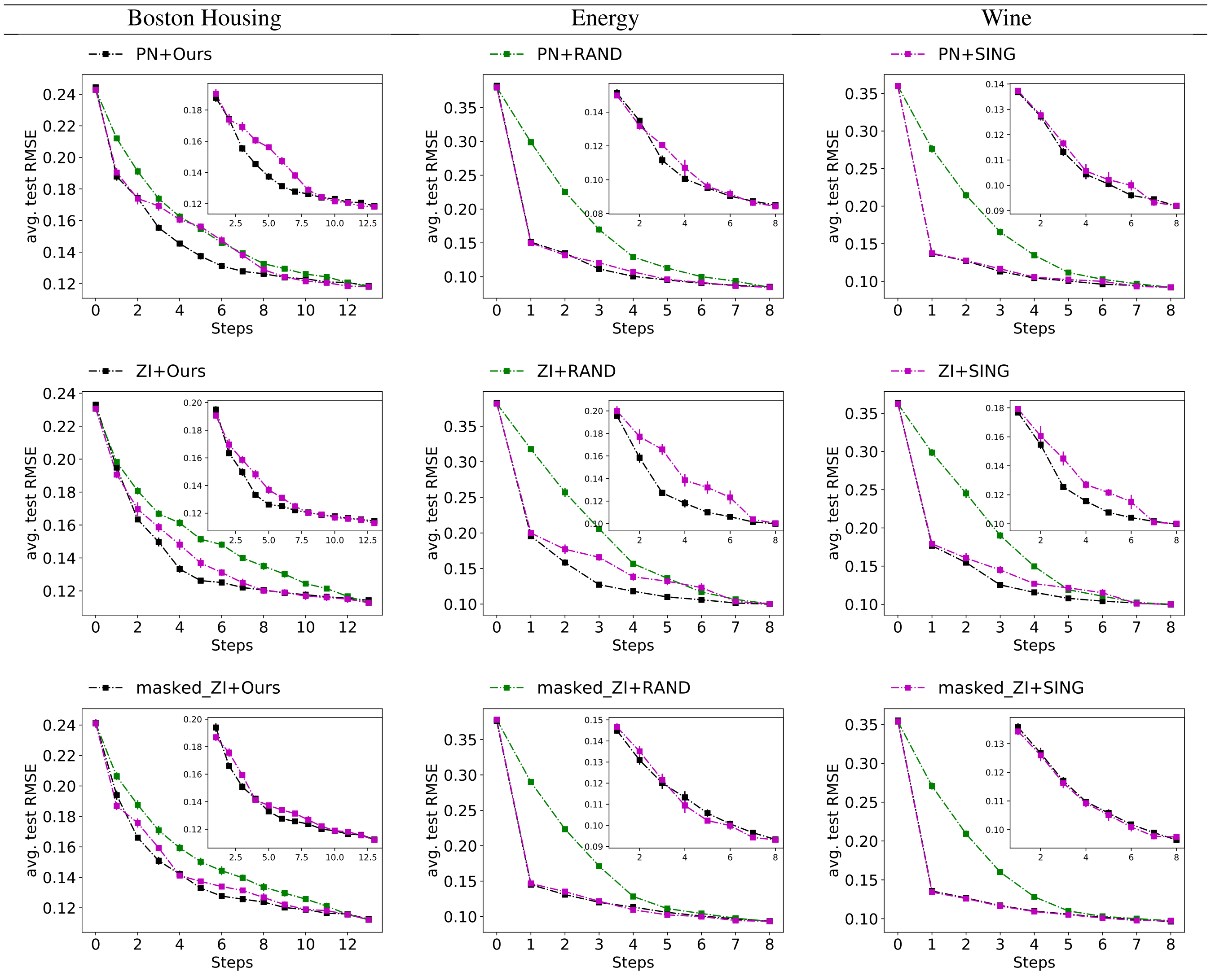


Figure 9: information curves (based on RMSE) of active variable selection for the three UCI datasets and the three approaches, i.e. **(First row)** PointNet (PN), **(Second row)** Zero Imputing (ZI), and **(Third row)** Zero Imputing with mask (ZI-m). **Green**: random strategy; **Black**: EDDI; **Pink**: Single best ordering. This displays RMSE (y axis, the lower the better) during the course of active selection (x-axis).

B.2.4 Negative test log Likelihood plots of PN, ZI and ZI-m on UCI datasets

Here we present additional plots of the negative test log likelihood curves during active variable selection. Figure 10 presents the results for the Boston Housing, the Energy and the Wine datasets and for the three approaches, i.e. PN, ZI and masked ZI.

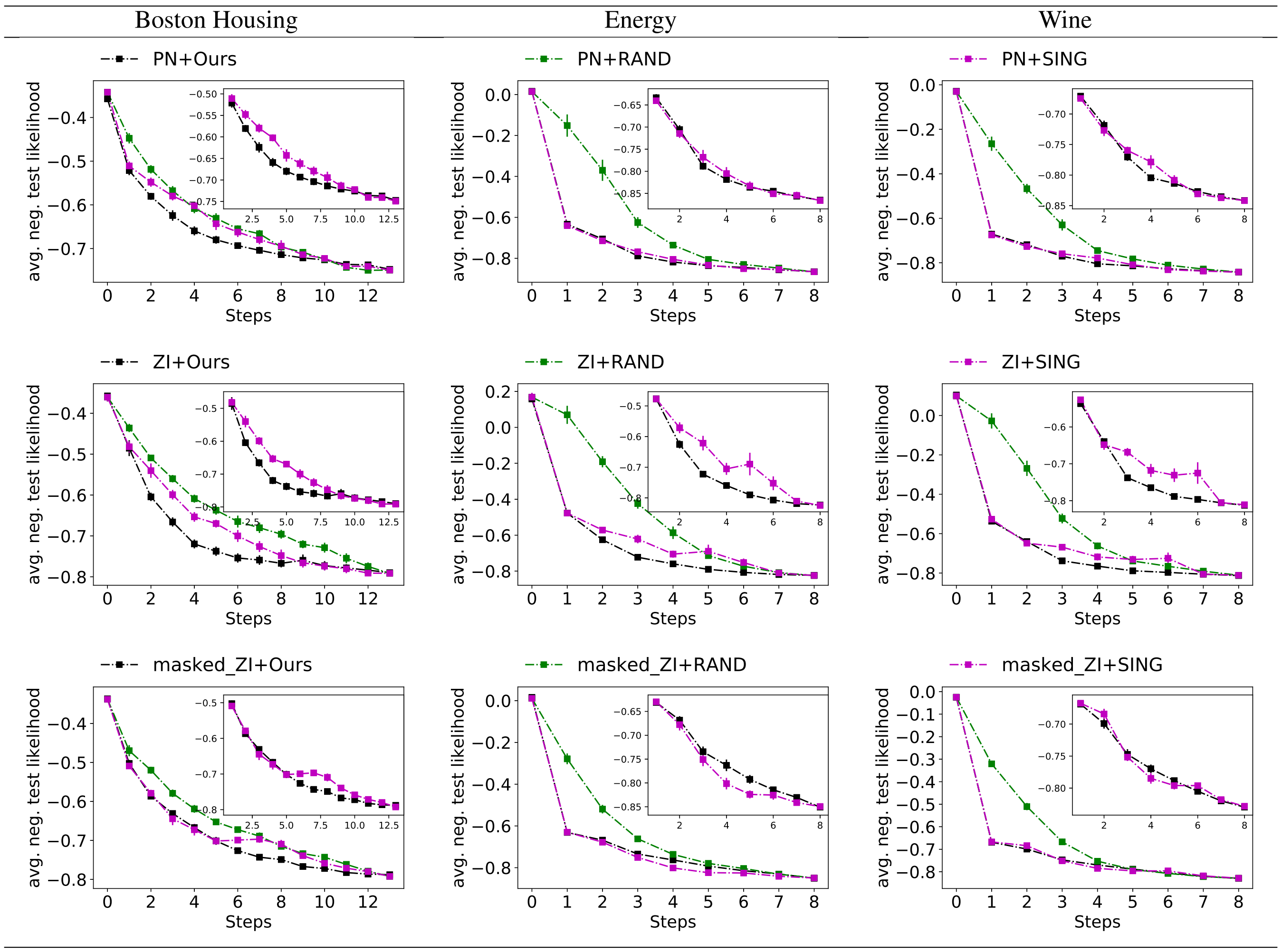


Figure 10: Information curves (based on test negative log-likelihood) of active variable selection for the three UCI datasets and the three approaches, i.e. **(First row)** PointNet (PN), **(Second row)** Zero Imputing (ZI), and **(Third row)** Zero Imputing with mask (ZI-m). **Green**: random strategy; **Black**: EDDI; **Pink**: Single best ordering. This displays negative test log likelihood (y axis, the lower the better) during the course of active selection (x-axis).

B.2.5 Comparisons between EDDI and LASSO-based method

Here we present additional results of a new baseline, the LASSO-based feature selection. This is not presented in the main text since LASSO is designed for a different problem setting. It requires fully observed data, and only works in regression problems with one dimensional outputs. Both MIMIC III and NHANES tasks do not fulfill these requirements. Additionally, LASSO aims to select a global set of features to obtain the best performance instead of select the most informative feature given partially observed information, thus cannot be used in a sequential setting. We thus construct the LASSO feature selection baseline as follows for comparison: we first apply LASSO regression on training dataset which is fully observed in these UCI datasets, and select the features (denoted by $\mathscr{A}$) that correspond to non-zero coefficients. Then, during test time, LASSO strategy will observe the features one by one from $\mathscr{A}$ randomly. When all variables selected by LASSO are already picked, we stop the feature selection progress. Once LASSO has completed feature selection, we use we use the corresponding partial-VAE (ZI,ZI-m,PNP,PN) to make predictions for fairness.

Figure 11 presents the results for the Boston Housing, the Energy and the Wine datasets as examples. Full results of all UCI datasets are presented in Table 7. Note that in Table 7, Wilcoxon signed-rank test is performed between EDDI and LASSO strategies for each Partial VAE models, respectively. The results indicates that EDDI significantly outperforms LASSO in all circumstances. This is despite the fact that EDDI is a greedy sequential variable selection method that built upon partially observed data, while LASSO-baseline makes use of the information from *fully observed data*, and selects the set of variables in a *non-greedy, global manner*, which is often unrealistic in many pratical application settings.

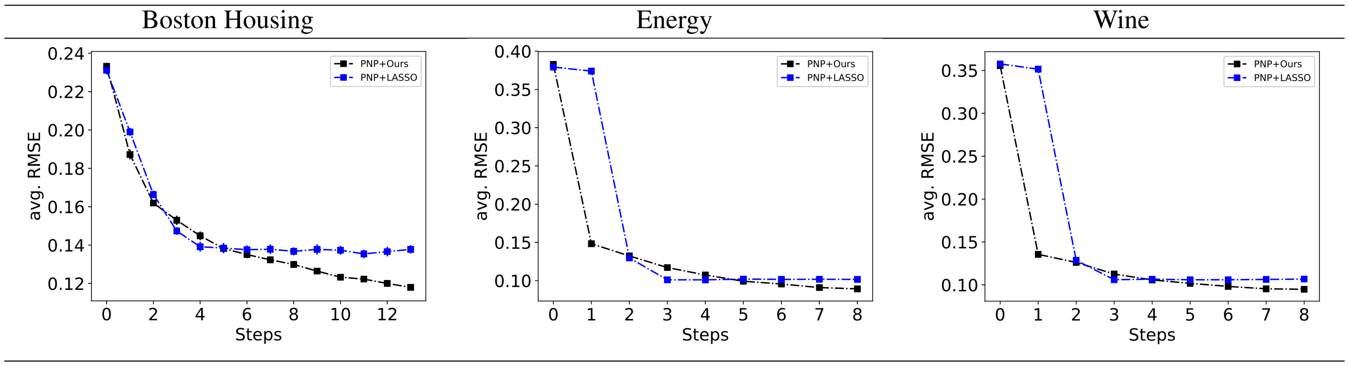


Figure 11: Information curves of active variable selection for the three UCI datasets and PNP-Partial VAE. **Black**: EDDI; **Blue**: Single best ordering. This displays test RMSE (y axis, the lower the better) during the course of active selection (x-axis).

Table 7: Avg. rankings of AUIC (RMSE-based), and p- values of Wilcoxon signed-rank test that EDDI outperforms LASSO (on 6 UCI datasets).

Method	ZI	ZI-m	PNP	PN
EDDI	4.66 (0.02)	4.53(0.02)	**4.14**(0.02)	4.24(0.02)
LASSO	4.86(0.02)	4.63(0.02)	4.41(0.02)	4.48(0.02)
p-value	$<10^{-4}$	$<10^{-6}$	$<10^{-24}$	$<10^{-19}$

B.2.6 Illustration of decision process of EDDI (Boston Housing as example)

The decision process facilitated by the active selection of the variables (for the EDDI framework) is efficiently illustrated in Figure 12 and Figure 13 for the Boston Housing dataset and for the PNP and PNP with single best ordering approaches, respectively.

For completeness, we provide details regarding the abbreviations of the variables used in the Boston dataset and appear both figures.

CR - per capita crime rate by town

PRD - proportion of residential land zoned for lots over 25,000 sq.ft.

PNB - proportion of non-retail business acres per town.

CHR - Charles River dummy variable (1 if tract bounds river; 0 otherwise)

NOC - nitric oxides concentration (parts per 10 million)

ANR - average number of rooms per dwelling

AOUB - proportion of owner-occupied units built prior to 1940

DTB - weighted distances to five Boston employment centres

ARH - index of accessibility to radial highways

TAX - full-value property-tax rate per $10,000

OTR - pupil-teacher ratio by town

PB - proportion of blacks by town

LSP - % lower status of the population

B.3 MIMIC-III

Here we provide additional results of our approach on the MIMIC-III dataset.

B.3.1 Preprocessing and model details

For our active learning experiments on MIMIC III datasets, we chose the variable of interest $\mathbf{x}_\phi$ to be the binary mortality indicator of the dataset. All data (except the binary mortality indicator) are normalized and then scaled between 0 and 1. We transformed the categorical variables into real-valued using the dictionary deduced from (Johnson et al., 2016) that makes use of the actual medical implications of each possible values. The binary mortality indicator are treated as Bernoulli variables and Bernoulli likelihood function is applied. For each repetition (of the 5 in total), we randomly draw 10% of the whole data to be our test set. Partial VAE models (ZI, ZI-m, PNP and PNs) share the same size of architecture with 10 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 10-50-100-D neural network with ReLU activations (where D is the data dimensions). The inference nets (encoder) share the same structure of D-100-50-20 that maps the observed data into distributional parameters of the latent space. Additionally, for PN-based parameterizations, we further use a 20 dimensional feature mapping h parameterized by a single layer neural network, and 10 dimensional ID vectors $\mathbf{e}_i$ (please refer to section 3.2) for each variable. We choose the symmetric operator g to be the basic summation operator.

Adam optimization and random missingness is applied as in the previous experiments. We trained our models for 3K iterations. During active learning, we draw 50 samples in order to estimate the expectation under $\mathbf{x}_\phi, \mathbf{x}_i \sim p(\mathbf{x}_\phi, \mathbf{x}_i|\mathbf{x}_o)$ in Equation (8). Loss functions (RMSEs and negative log likelihoods) of the target variable is also estimated using samples of $\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)$ through $p(\mathbf{x}_\phi|\mathbf{x}_o) \approx \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{x}_\phi|\mathbf{z}_m)$, where $\mathbf{z}_m \sim q(\mathbf{z}|\mathbf{x}_o)$.

B.3.2 Additional plots of ZI, PN and ZI-m on MIMIC III

Figure 14 shows the information curves (Bernoulli negative test likelihood-based) of active variable selection on the risk assessment task for MIMIC-III as produced by the three approaches, i.e. ZI, PN and masked ZI.

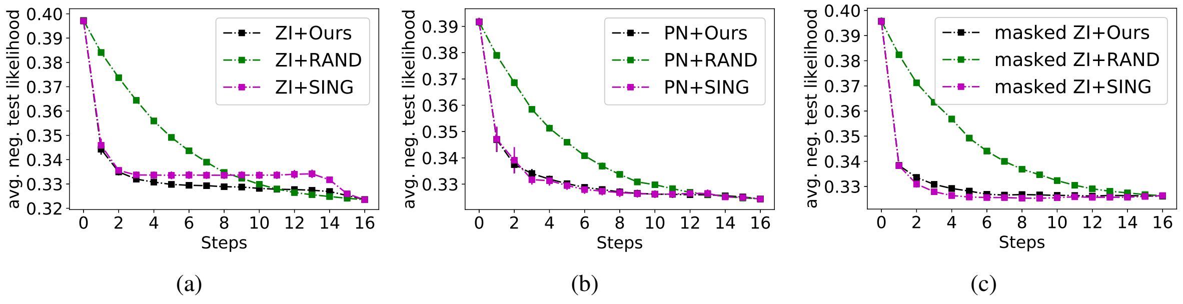


Figure 14: Information curves of active variable selection on risk assessment task on MIMIC III, produced from: **(a)** Zero Imputing (ZI), **(b)** PointNet (PN) and **(c)** Zero Imputing with mask (ZI-m). **Green**: random strategy; **Black**: EDDI; **Pink**: Single best ordering. This displays negative test Bernoulli likelihood (y axis, the lower the better) during the course of active selection (x-axis)

.

B.4 NHANES

B.4.1 Preprocessing and model details

For our active learning experiments on NHANES datasets, we chose the variable of interest $\mathbf{x}_\phi$ to be the lab test result section of the dataset. All data are normalized and scaled between 0 and 1. For categorical variables, these are transformed into real-valued variables using the code that comes with the dataset, which makes use of the actual ordering of variables in questionnaire. Then, for each repetition (of the 5 repetitions in total), we randomly draw 8000 data as training set and 100 data to be test set. All partial VAE models (ZI, ZI-m, PNP and PNs) uses gaussian likelihoods, with an diagonal Gaussian inference model (encoder). Partial VAE models share the same size of architecture with 20 dimensional diagonal Gaussian latent variables: the generator (decoder) is a 20-50-100-D neural network. The inference nets (encoder) share the same structure of D-100-50-20 that maps the observed data into distributional parameters of the latent space. Additionally, for PN-based parameterizations, we further use a 20 dimensional feature mapping h parameterized by a single layer neural network, and 100 dimensional ID vectors $\mathbf{e}_i$ (please refer to section 3.2) for each variable. We choose the symmetric operator g to be the basic summation operator.

Adam optimization and random missingness is applied as in the previous experiments. We trained all models 1K iterations. During active learning, 10 samples were drawn to estimate the expectation in Equation (9). Losses (RMSEs) of the target variable is also estimated using 10 samples.

C Additional Theoretical Contributions

C.1 Zero imputing as a Point Net

Here we present how the zero imputing (ZI) and PointNet (PN) approaches relate.

Zero imputation with inference net In ZI, the natural parameter of λ (e.g., Gaussian parameters in variational autoencoders) is approximated using the following neural network:

$$f(\mathbf{x}) := \sum_{l=1}^{L} w_l^{(1)} \sigma(\mathbf{w}_l^{(0)} \mathbf{x}^T)$$

,

where L is the number of hidden units, $\mathbf{x}$ is the input image with x_i be the value of the i^{th} pixel. To deal with partially observed data $\mathbf{x} = \mathbf{x}_o \cup \mathbf{x}_u$, ZI simply sets all $\mathbf{x}_u$ to zero, and use the full inference model $f(\mathbf{x})$ to perform approximate inference.

PointNet parameterization The PN approach approximates the natural parameter λ by a permutation invariant set function

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)),$$

where $\mathbf{s}_i = [x_i, \mathbf{e}_i]$, $\mathbf{e}_i$ is the I dimensional embedding/ID/location vector of the i^{th} pixel, $g(\cdot)$ is a symmetric operation such as max-pooling and summation, and $h(\cdot)$ is a nonlinear feature mapping from $\mathbb{R}^{I+1}$ to $\mathbb{R}^K$ (we will always refer h as *feature maps*). In the current version of the partial-VAE implementation, where Gaussian approximation is used, we set $K = 2H$ with H being the dimension of latent variables. We set g to be the element-wise summation operator, i.e. a mapping from $\mathbb{R}^{KO}$ to $\mathbb{R}^K$ defined by:

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) = \sum_{i \in O} h(\mathbf{s}_i).$$

This parameterization corresponds to products of multiple Exp-Fam factors $\prod_{i \in O} \exp\{-\langle h(\mathbf{s}_i), \Phi \rangle\}$.

From PN to ZI To derive the PN correspondence of the above ZI network we define the following PN functions:

$$h(\mathbf{s}_i) := \mathbf{e}_i * x_i$$

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) := \sum_{k=1}^{I} \theta_k \sigma(\sum_{i \in O} h_k(\mathbf{s}_i)),$$

where $h_k(\cdot)$ is the k^{th} output feature of $h(\cdot)$. The above PN parameterization is also permutation invariant; setting $L = I$, $\theta_l = w_l^{(1)}$,$(\mathbf{w}_l^{(0)})_i = (\mathbf{e}_i)_l$ the resulting PN model is equivalent to the ZI neural network.

Generalizing ZI from PN perspective In the ZI approach, the missing values are replaced with zeros. However, this ad-hoc approach does not distinguish missing values from actual observed zero values. In practice, being able to distinguish between these two is crucial for improving uncertainty estimation during partial inference. One the other hand, we have found that PN-based partial VAE experiences difficulties in training. To alleviate both issues, we proposed a generalization of the ZI approach that follows a PN perspective. One of the advantages of PN is setting the *feature maps* of the unobserved variables to zero instead of the related weights. As discussed before, these two approaches are equivalent to each other only if the factors are linear. More generally, we can parameterize the PN by:

$$h^{(1)}(\mathbf{s}_i) := \mathbf{e}_i * x_i$$

$$h^{(2)}(h_i^{(1)}) := NN_1(h_i^{(1)})$$

$$g(h(\mathbf{s}_1), h(\mathbf{s}_2), ..., h(\mathbf{s}_O)) := NN_2(\sigma(\sum_{i \in O} h_k^{(2)}(h_i^{(1)}))),$$

where NN_1 is a mapping from $\mathbb{R}^I$ to $\mathbb{R}^K$ defined by a neural network, and NN_2 is a mapping from $\mathbb{R}^K$ to $\mathbb{R}^{2H}$ defined by another neural network.

C.2 Approximation Difficulty of the Acquisition Function

Traditional variational approximation approaches provide wrong approximation direction when applied in this case (resulting in an upper bound of the objective $R_\phi(i, \mathbf{x}_O)$ which we maximize). Justification issues aside, (black box) variational approximation requires sampling from approximate posterior $q(\mathbf{z}|\mathbf{x}_O)$, which leads to extra uncertainties and computations. For common proposals of approximation:

- Directly estimate entropy via sampling $\Rightarrow$ problematic for high dimensional target variables
- Using reversed information reward $\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[D_{KL}(p(\mathbf{x}_\phi|\mathbf{x}_o)||p(\mathbf{x}_\phi|\mathbf{x}_o, \mathbf{x}_i))]$, and then apply ELBO (KL-divergence) $\Rightarrow$ This does not make sense mathematically, since this will result in upper bound approximation of the (reversed) information objective, this is in the wrong direction.
- Ranganath's bound (Ranganath et al., 2016) on estimating entropy$\Rightarrow$ gives upper bound of the objective, wrong direction.
- All the above methods also needs samples from latent space (therefore second level approximation needed).

C.3 Connection of EDDI information reward with BALD

We briefly discuss connection of EDDI information reward with BALD (Houlsby et al., 2011) and. MacKay's work (MacKay, 1992). Assuming the model is correct, i.e. $q = p$, we have

$$\begin{aligned} R(i, \mathbf{x}_o) = &\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[D_{KL}(p(\mathbf{z}|\mathbf{x}_i, \mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o))] \\ &- \mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i, \mathbf{x}_o)}\left[D_{KL}(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_i, \mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_o))\right]. \end{aligned}$$

Note that based on McKay's relationship between entropy and KL-divergence reduction, we have:

$$\begin{aligned} &\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[D_{KL}(p(\mathbf{z}|\mathbf{x}_i, \mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_o))] \\ =&\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}[H(p(\mathbf{z}|\mathbf{x}_i, \mathbf{x}_o)) - H(p(\mathbf{z}|\mathbf{x}_o))]. \end{aligned}$$

Similarly, we have

$$\begin{aligned} &\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i, \mathbf{x}_o)}\left[D_{KL}(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_i, \mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_o))\right] \\ =&\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_\phi, \mathbf{x}_o)}\left[D_{KL}(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_i, \mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_o))\right] \\ =&\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_\phi, \mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_i, \mathbf{x}_o)) - H(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_o))\right] \\ =&\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i, \mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_i, \mathbf{x}_o))\right] - \mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_\phi, \mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_o))\right] \\ =&\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_i, \mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_i, \mathbf{x}_o))\right] - \mathbb{E}_{\mathbf{x}_\phi \sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi, \mathbf{x}_o))\right], \end{aligned}$$

where MacKay's result is applied to $\mathbb{E}_{\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{x}_\phi,\mathbf{x}_o)}\left[D_{KL}(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o)||p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]$. Putting everything together, we have

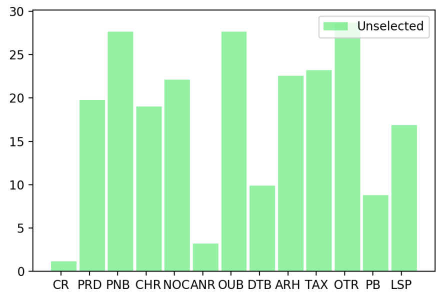
$$\begin{aligned}
R(i,\mathbf{x}_o) = & \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)) - H(p(\mathbf{z}|\mathbf{x}_o))\right]] \\
& - \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right] + \mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right] \\
& = \left\{\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o))\right] - \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right]\right\} \\
& - \left\{\mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_o))\right] - \mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_o))\right]\right\}.
\end{aligned}$$

We can show that

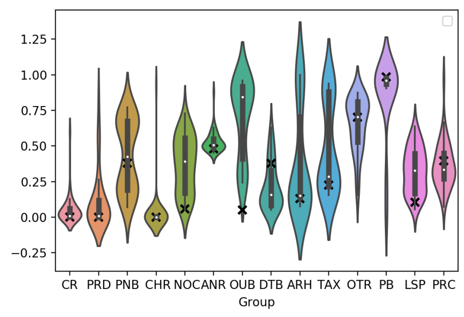
$$\begin{aligned}
& H(p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)) - \mathbb{E}_{\mathbf{x}_\phi\sim p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}\left[H(p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o))\right] \\
= & -\int_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)\log p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)d\mathbf{z} + \int_{\mathbf{z},\mathbf{x}_\phi} p(\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)\log p(\mathbf{z}|\mathbf{x}_\phi,\mathbf{x}_i,\mathbf{x}_o) \\
= & \int_{\mathbf{z},\mathbf{x}_\phi} p(\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)\log\frac{p(\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)}{p(\mathbf{z}|\mathbf{x}_i,\mathbf{x}_o)p(\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o)} \\
= & \mathscr{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o\right],
\end{aligned}$$

which is exactly the conditional mutual information $\mathscr{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o\right]$ used in BALD. Therefore, our chain rule representation of reward function leads us to

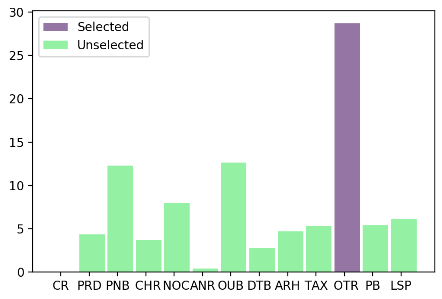
$$R(i,\mathbf{x}_o) = \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathscr{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_i,\mathbf{x}_o\right] - \mathbb{E}_{\mathbf{x}_i\sim p(\mathbf{x}_i|\mathbf{x}_o)}\mathscr{I}\left[\mathbf{z},\mathbf{x}_\phi|\mathbf{x}_o\right].$$



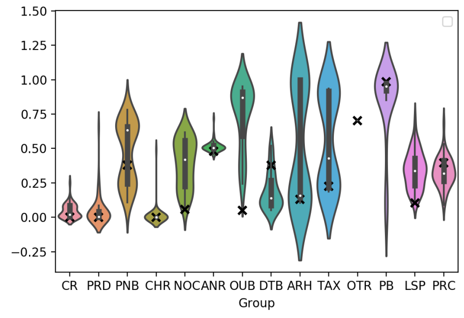
(a)



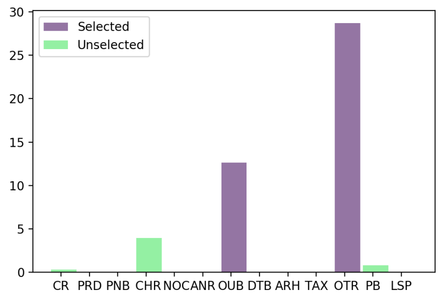
(b)



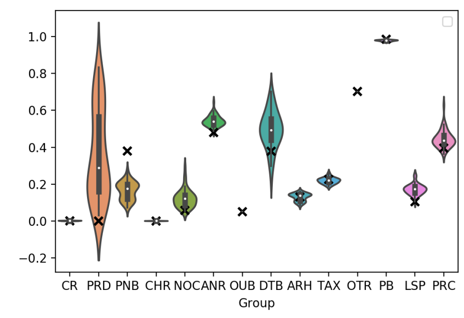
(c)



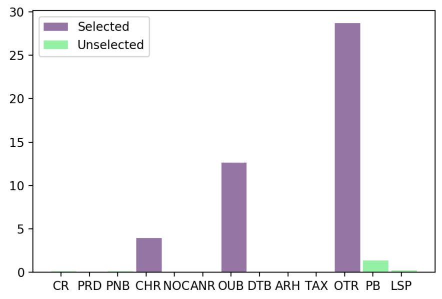
(d)



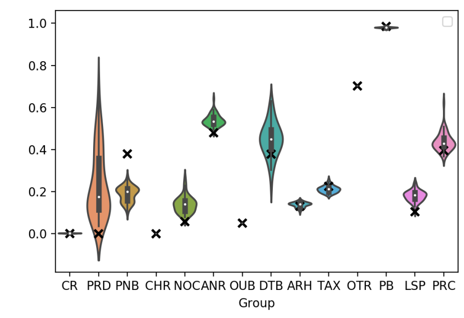
(e)



(f)



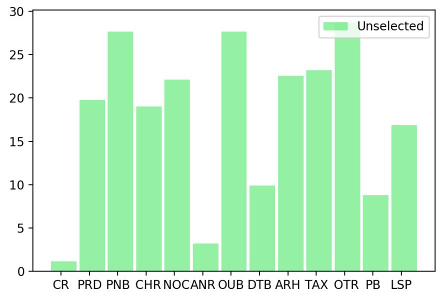
(g)



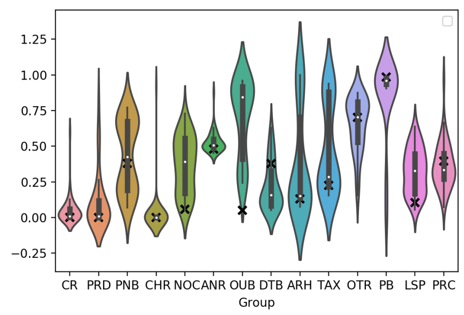
(h)

Figure 12: Information reward estimated during the first 4 active variable selection steps on a randomly chosen Boston Housing test data point. **Model**: PNP, strategy: EDDI. Each row contains two plots regarding the same time step. **Bar plots on the left** show the information reward estimation of each variable on the y-axis. All unobserved variables start with green bars, and turns purple once selected by the algorithm. **Right**: violin plot of the posterior density estimations of remaining unobserved variables.

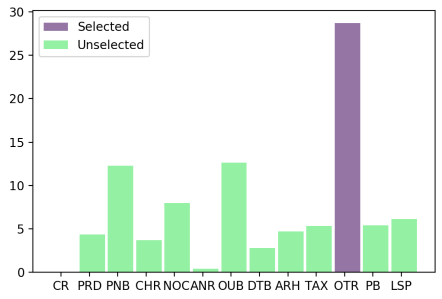
.



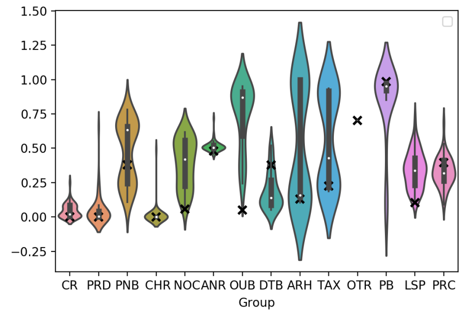
(a)



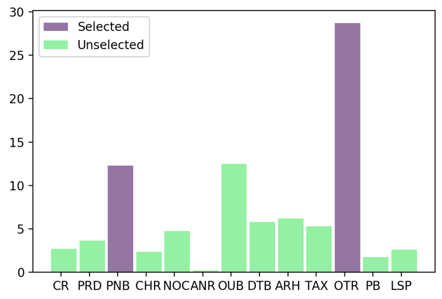
(b)



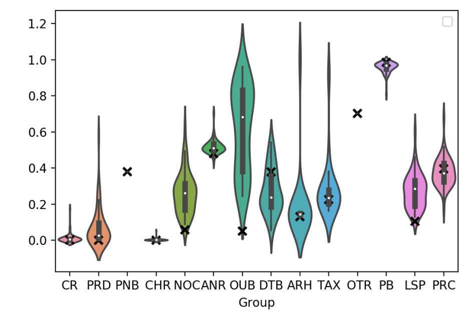
(c)



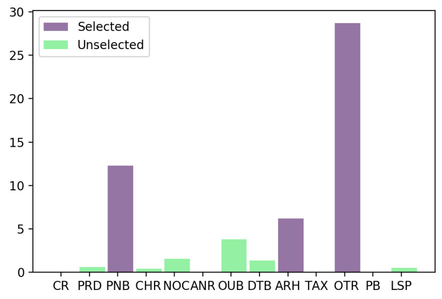
(d)



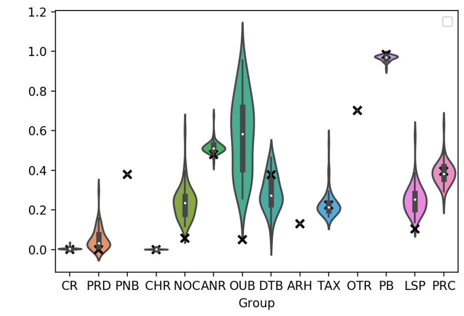
(e)



(f)



(g)



(h)

Figure 13: Information reward estimated during the first 4 active variable selection steps on a randomly chosen Boston Housing test data point. **Models**: PNP, strategy: single ordering. Each row contains two plots regarding the same time step. **Bar plots on the left** show the information reward estimation of each variable on the y-axis. All unobserved variables start with green bars, and turns purple once selected by the algorithm. **Right**: violin plot of the posterior density estimations of remaining unobserved variables.

.