Imputation for prediction: beware of diminishing returns.

Marine Le Morvan
Soda, Inria Saclay
marine.le-morvan@inria.fr

Gaël Varoquaux
Soda, Inria Saclay
gael.varoquaux@inria.fr

Abstract

Missing values are prevalent across various fields, posing challenges for training and deploying predictive models. In this context, imputation is a common practice, driven by the hope that accurate imputations will enhance predictions. However, recent theoretical and empirical studies indicate that simple constant imputation can be consistent and competitive. This empirical study aims at clarifying *if* and *when* investing in advanced imputation methods yields significantly better predictions. Relating imputation and predictive accuracies across combinations of imputation and predictive models on 20 datasets, we show that imputation accuracy matters less i) when using expressive models, ii) when incorporating missingness indicators as complementary inputs, iii) matters much more for generated linear outcomes than for real-data outcomes. Interestingly, we also show that the use of the missingness indicator is beneficial to the prediction performance, *even in MCAR scenarios*. Overall, on real-data with powerful models, improving imputation only has a minor effect on prediction performance. Thus, investing in better imputations for improved predictions often offers limited benefits.

1 Introduction

Databases are often riddled with missing values due to faulty measurements, unanswered questionnaire items or unreported data. This is typical of large health databases such as the UK Biobank [Sudlow et al., 2015], the National Health Interview Survey [Blewett et al., 2019] and others [Perez-Lebel et al., 2022]. Statistical analysis with missing values has been widely studied, particularly to estimate parameters such as means and variances [Little and Rubin, 2019]. However, how to best deal with missing values for prediction has been less studied. Since most machine learning models do not natively handle missing values, common practice is to impute missing values before training a model on the completed data, often with the expectation that "good" imputation improves predictions. Considerable efforts have been dedicated to improving imputation techniques, utilizing Generative Adversarial Networks [Yoon et al., 2018], Variational AutoEncoders [Mattei and Frellsen, 2019], optimal transport [Muzellec et al., 2020] or AutoML-enhanced iterative conditional imputation [Jarrett et al., 2022] among others. Most of these studies concentrate on imputation accuracy without assessing performance on subsequent tasks. However, theoretical arguments suggest that good imputation is not needed for good prediction [Le Morvan et al., 2021, Josse et al., 2019]. These arguments are asymptotic and whether they hold in typical cases is debatable. To address the discrepancy between this theory and the emphasis on imputation efforts, there is a critical need for empirical studies to determine whether better imputations actually lead to better predictions.

Theory does establish that in some scenarios, better imputations imply better predictions. For instance with a linearly-generated outcome, the optimal prediction writes as a linear model on the optimally-imputed data Le Morvan et al. [2021]. Thus, using a linear model for prediction, better imputations typically yield better predictions. Interestingly though, theoretical results indicate that for linear

Preprint. Under review.

models, simple constant imputations suffice in very high dimensions [Ayme et al., 2023], or in small dimensions with uncorrelated features [Ayme et al., 2024]. Beyond linear models, empirical studies on real data have shown the competitiveness of simple imputations –such as mean– [Paterakis et al., 2024, Perez-Lebel et al., 2022, Shadbahr et al., 2023] aligning with theoretical arguments. However, their findings may be driven by "predictive" missingness [Missing Not At Random data, for which most imputation methods are invalid Little and Rubin, 2019, Josse et al., 2019]. In addition, these studies have not quantified the effect of imputation accuracy on prediction performance nor how these effects are modulated by the choice of downstream model, the use of a missingness indicator, the missingness mechanism or dataset-related features. Our work fills this gap and provides actionable conclusions.

We study settings where good imputations are most likely to benefit downstream predictive performance, in order to obtain an upper bound on the potential benefits of improved imputation in general cases. If only minor benefits are observed in this context, it would suggest even smaller benefits in general. Specifically, there are 3 types of missingness mechanisms [Rubin, 1976]: (i) Missing Completely At Random (MCAR), where each entry is missing with a fixed probability; (ii) Missing At Random (MAR), where the probability of a value being missing depends only on observed variables, (iii) Missing Not At Random settings (MNAR), where the missingness is informative and encompasses all other scenarios. Most imputation algorithms are valid under MCAR but not under MNAR. Therefore, we focus on the MCAR settings where imputation algorithms are most likely to produce accurate imputations. Additionally, we focus on datasets with only numerical values. For categorical features, handling missing values by treating them as a new category is a popular and effective approach that does not involve a notion of imputation accuracy. In this setting, we characterize the relationship between imputation accuracy and predictive performance for several downstream prediction models, with and without the missingness indicator. Properly benchmarking methodologies with missing values is particularly resource-intensive, as already emphasized in previous works [Jäger et al., 2021, Perez-Lebel et al., 2022]. Indeed, computing costs are driven by the combinatorics of imputation and prediction models, hyperparameter optimization for both, inclusion and exclusion of the indicator, and varying missing rates.

Section 2 introduces related work, covering both benchmarks for prediction with missing values and available theory. Section 3 details our experimental procedures, specifically the methods examined. Section 4 presents our findings, relating gains in imputation to prediction performance. Finally, section 5 summarizes the lessons learned.

2 Related work

Benchmarks. Several benchmark studies have investigated imputation in a prediction context [Paterakis et al., 2024, Jäger et al., 2021, Ramosaj et al., 2022, Woźnica and Biecek, 2020, Perez-Lebel et al., 2022, Poulos and Valle, 2018, Shadbahr et al., 2023, Li et al., 2024]. However, drawing definitive conclusions from most studies is challenging due to various limitations in scope and experimental choices. For example, Bertsimas et al. [2018], Li et al. [2024] trained imputation methods using both the training and test sets, rather than applying the imputation learned on the training set to the test set, which is not possible with many imputation packages. This approach creates data leakage. Woźnica and Biecek [2020] trained imputers separately on the train and test sets, which creates an "imputation shift" between the training and test data. Jäger et al. [2021] discards and imputes values in a single column of the test set, chosen at random and fixed throughout the experiments. Yet as they note, conclusions can change drastically depending on the importance of the to-be-imputed column for the prediction task or its correlation with other features. Some studies [Poulos and Valle, 2018, Ramosaj et al., 2022] use a small number of datasets (resp. 2 and 5 datasets from the UCI machine learning repository respectively), thus limiting the significance of their conclusions. Woźnica and Biecek [2020] do not perform hyperparameter tuning for the prediction models, while Ramosaj et al. [2022] tunes hyperparameters on the complete data, though it is unclear whether the best hyperparameters on complete data are also the best on incomplete data. Furthermore, some benchmarks focus on specific types of downstream prediction models, such as linear models [Jäger et al., 2021], AutoML models [Paterakis et al., 2024] or Support Vector Machines [Li et al., 2024], meaning their conclusions should not be generalized to all types of downstream models. The data used in benchmarks also affects the scope of the conclusions. For instance, Paterakis et al. [2024] considers 35 datasets, most with only a few hundred samples and a 50% train-test split, making

their conclusions more relevant to small data scenarios. Finally, only Perez-Lebel et al. [2022] and Paterakis et al. [2024] evaluate the use of the missingness indicator as complementary input features.

Among the benchmarks with largest scope, Paterakis et al. [2024] recommend mean/mode imputation with the indicator as the default option in AutoML settings, both for native and simulated missingness. It is among the top-performing approaches, never statistically significantly outperformed, and is also the most cost-effective. They also show that using the missingness indicator as input improves performances slightly but significantly for most imputation methods. Perez-Lebel et al. [2022] focus on predictive modeling for large health databases, which contain many naturally occurring missing values. They compare various imputation strategies (mean, median, k-nearest neighbors imputation, MICE) combined with gradient-boosted trees (GBTs) for prediction, as well as GBTs with native handling of missing values. Similarly to Paterakis et al. [2024], they find that appending the indicator to the imputed data significantly improves performances, which may reflect MNAR data. While they recommend resorting to the native handling of missing values as it is relatively cheap, their results further indicate that no method is significantly better than using the mean as imputation method together with the indicator. Shadbahr et al. [2023] also find that the best imputations do not necessarily result in the best downstream performances. Using an analysis of variance, they show that the choice of imputation method has a significant but small effect on the classification performance.

Whether better imputation leads to better prediction may vary depending on factors like the choice of downstream model, the missingness rate, or the specific datasets. Yet, many studies seek a definitive conclusion across diverse settings. Only Paterakis et al. [2024] conducted a meta-analysis, yet it did not determine when more advanced imputation strategies are beneficial compared to mean or mode imputations. Identifying scenarios in which better imputations are more likely to improve predictions is however of strong practical interest.

Theoretical insights. Previous works have also addressed this question from a theoretical point of view. Le Morvan et al. [2021] showed that for all missingness mechanisms and almost all deterministic imputation functions, universally consistent algorithms trained on imputed data asymptotically achieve optimal performances in prediction. This is in particular true for simple imputations such as the mean [Josse et al., 2019], thereby providing rationale to favor simple imputations over more accurate ones. Essentially, optimal prediction models can be built on mean-imputed data by modeling the mean as a special value encoding for missingness. Ayme et al. [2023] also provide theoretical support for the use of simple imputations, as they advocate for the use of zero imputation in high-dimensional settings. They show that, for a linear regression problem and MCAR missingness, learning on zero-imputed data instead of complete data incurs an imputation bias that goes to zero when the dimension increases. This holds given certain assumptions on the covariance matrix, which intuitively impose some redundancy among variables. Finally, Van Ness et al. [2023] prove that in MCAR settings, the best linear predictor assigns zero weights to the missingness indicator, whereas these weights are non-zero in MNAR settings. Their theoretical results imply that the missingness indicator neither degrades nor enhances performances asymptotically in MCAR.

3 Experimental setup.

Imputation methods. We chose four imputation models to cover a wide range of imputation qualities. Importantly, our goal *is not* to extensively benchmark different imputation strategies but to understand the link between imputation quality and prediction performance.

mean - each missing value is imputed with the mean of the observed values in a given variable. It provides a useful baseline for assessing the effectiveness of advanced techniques.

iterativeBR - each feature is imputed based on the other features in a round-robin fashion using a Bayesian ridge regressor. This method is related to `mice` [Van Buuren and Groothuis-Oudshoorn, 2011] in that it is also based on a fully conditional specification [Van Buuren, 2018]. It is implemented in `scikit-learn`'s `IterativeImputer` [Pedregosa et al., 2011].

missforest [Stekhoven and Bühlmann, 2012] - operates in a manner analogous to `iterativeBR`, wherein it imputes one feature using all others and iteratively enhances the imputation by sequentially addressing each feature multiple times. The key distinction lies in its utilization of a random forest for imputation rather than a linear model. We used `scikit-learn`'s `IterativeImputer` with `RandomForestRegressor` as estimators. Default parameters for Missforest were set to `n_estimators=30` and `max_depth=15` for the random forests (the higher the better) to keep a

reasonable computational budget. Note that in `HyperImpute` [Jarrett et al., 2022], random forests are more limited: 10 trees and a maximum depth of 4.

condexp - uses the conditional expectation formula of a multivariate normal distribution to impute the missing entries given the observed ones. The mean and covariance matrix of the multivariate normal distribution are estimated with (pairwise) available-case estimates [Little and Rubin, 2019, section 3.4], i.e., the $(i, j)^{th}$ entry of the covariance matrix is estimated solely from samples where both variables i and j are observed. This approach offers computational advantages over more resource-intensive approaches such as the Expectation-Maximization (EM) algorithm. It is related to Buck's method [Buck, 1960, Little and Rubin, 2019, section 4.2].

Mean imputation can be expected to give the worst imputation, with other methods offering varying improvements. In particular, `missforest` often delivers top-tier performance on tabular data.

Models: As the effect of imputation on prediction quality can be modulated by the predictive model used, we included three predictive models. In particular, we took care to include both deep learning and tree-based representatives, as the prediction functions produced by these models have different properties, for example regarding their smoothness.

- **MLP**: a basic Multilayer Perceptron with ReLU activations, to serve as a simple baseline.
- **SAINT** [Somepalli et al., 2021]: Self-Attention and Intersample Attention Transformer (SAINT) is a deep tabular model that performs both row and column attention. The numerical features are first embedded to a d-dimensional space before being fed to the transformer. We chose SAINT as it has been shown to be state-of-the-art among deep learning approaches for tabular data in several surveys [Borisov et al., 2022, Grinsztajn et al., 2022].
- **XGBoost** [Chen and Guestrin, 2016]: We chose XGBoost as it is a popular state-of-the-art boosting method, and it has been shown to be the best tree-based model on regression tasks with numerical features only in Grinsztajn et al. [2022].

We tuned hyperparameters with Optuna [Akiba et al., 2019] for XGBoost and MLP. For SAINT, we used the default hyperparameters provided by its authors [Somepalli et al., 2021] for computational reasons. Tables 2 to 4 give all details on hyperparameter tuning and default hyperparameters.

Native handling of missing values: Both SAINT and XGBoost can directly be applied on incomplete data, without prior imputation of the missing values, each with its own strategy. XGBoost uses the Missing Incorporated in Attribute (MIA) [Twala et al., 2008, Josse et al., 2019] approach. When splitting, samples with missing values in the split feature can go left, right, or form their own leaf. MIA retains the option that minimizes the prediction error. In SAINT, numerical features are embedded in a d-dimensional space using simple MLPs. In case of missing value, a learnable d-dimensional embedding is used to represent the `NaN`. Each feature has its own missingness embedding.

The datasets We use a benchmark created by Grinsztajn et al. [2022] for tabular learning. It comprises 20 datasets (listed in table 1), each corresponding to a regression task with continuous and ordinal features. Missing data is generated according to a MCAR mechanism with either 20% or 50% misssing rate. Continuous features are gaussianized using scikit-learn's `QuantileTransformer` while ordinal features are standard scaled to have a zero mean and unit variance. This is true for all imputation and model combination except for XGBoost with native handling of missing values, as it is not expected to benefit from a normalization. The outputs y are also standard scaled. In all cases, the parameters of these data normalizations are learned on the train set with missing values. We also provide experiments on semi-synthetic data where the response y is simulated as a linear function of the original data X. The coefficients β of the linear function are all taken equal and scaled so that the variance of $\beta^\top X$ is equal to 1. Noise is added with a signal-to-noise ratio of 10.

Evaluation strategy Each dataset is randomly split into 3 folds (train - 80%, validation - 10% and test - 10%), and each split is furthermore capped at 50,000 samples (table 1). Train, validation and test sets are imputed using the same imputation model trained on the train set. Prediction models are then trained on the imputed train set. For XGBoost and the MLP, hyperparameters are optimized using Optuna [Akiba et al., 2019] with 50 trials, i.e, Optuna draws 50 sets of hyperparameters, trains a model for each of these hyperparameter sets, and retains the best one according to the prediction performance on the validation set. When the indicator is used, it is appended as extra features to the imputed data, it is not leveraged for the imputation stage. We run all combinations of the 3 prediction

models with the 4 imputation techniques, with and without the indicator, resulting in $4 \times 3 \times 2 = 24$ models to which we add XGBoost and SAINT with native handling of missing values. This results in a total of 26 models displayed in Figure 1. Finally, the whole process is repeated with 10 different train/validation/test splits. For reproducibility, the code will be available on GitHub upon publication of the preprint.

Computational resources. Multiplying the number of models (26) with the number of datasets (20 + 20 linear versions), the hyperparameter tuning (50 trials), the number of repetitions of the experiments (10), and the 2 missing rates, we get a very large number of runs (around 700,000). As some methods are computationally expensive –such as missforest for imputing, as well as SAINT notably when the indicator is used–, these experiments required a total of 2.5 CPU years. A quarter of this time was dedicated to imputation.

4 Results: Determinants of predictions performance with MCAR missingness

4.1 Benefits of sophisticated imputation, the indicator, and XGBoost amid high variance.

Figure 1 summarizes the relative performance of the various predictors combined with the different imputation schemes across the 20 datasets. Some trends emerge: more sophisticated imputers tend to improve prediction, with missForest-based predictors often outperforming those using condexp or iterativeBR imputers, which in turn outperform predictors based on mean imputation. However, using the missingness indicator decreases this effect. Additionally, this effect is barely noticeable for the best predictor, XGBoost, which appears to maintain its advantages on tabular data [as described in Grinsztajn et al., 2022] even in the presence of missing values.

That the best predictor barely benefits from fancy imputers brings us back to our original question: should efforts go into imputation? Drawing a conclusion from fig. 1 would be premature: the variance across datasets is typically greater than the difference in performance between methods (critical difference diagram in figs. 6 and 7). For example, missforest + XGBoost + indicator outperforms all other methods on only a third of datasets. Additionally, XGBoost + indicator does not perform significantly better with missforest than with condexp, while mean imputation does not always lead to the worst prediction. In what follows, we focus on quantifying the effects of improved imputation accuracy on predictions in different scenarios.

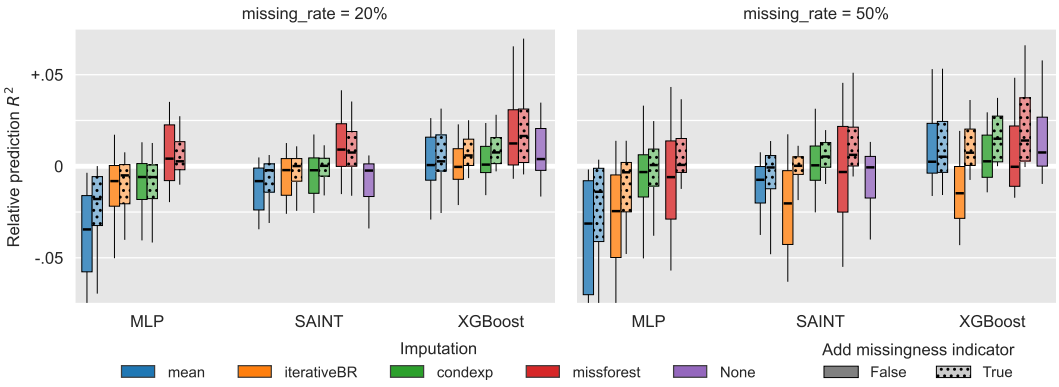

Figure 1: **Relative prediction performances across datasets for different imputations, predictors, and use of the missingness indicator.** Each boxplot represents 200 points (20 datasets with 10 repetitions per dataset). The performances shown are R^2 scores on the test set relative to the mean performance across all models for a given dataset and repetition. A value of 0.01 indicates that a given method outperforms the average performance on a given dataset by 0.01 on the R^2 score. Corresponding critical difference plots in figs. 6 and 7.

4.2 A detour through imputation accuracies: how do imputers compare?

Although comparing imputers is not our main objective, it is enlightening for our prediction purpose to characterize their relative performance range. Figure 2 (left) gives imputation performances measured as the R^2 score between the imputed and ground truth values, relative to the average across methods and missing rates for each dataset. At 20% missing rate, the best imputer is missforest, followed by condexp and iterativeBR, nearly tied, and far behind mean imputation. At 50% missing rate, the imputation accuracy of all but mean imputation drop, but interestingly condexp is much less affected. It is interesting that such a simple method performs best. It is notably two orders of magnitude faster than missforest (figure 2 right), which makes it an imputation technique worth considering. It is possible that the gaussianization of the features helped condexp, although a feature-wise gaussianization does not produce a jointly Gaussian dataset.

This work does not aim to compare or identify the best imputers, but rather to achieve varying imputation qualities to highlight the link between imputation and prediction quality. In this regard, the high range of imputation accuracy between the best and worst methods (an average difference of 0.5 R^2 points at 20% and 0.3 R^2 points at 50%) allows capturing differences in prediction performance.

4.3 Linking imputation accuracy and prediction performances.

Combining the four imputation techniques with 10 repetitions of each experiment yields 40 (imputation R^2 , prediction R^2) pairs for each model and dataset. To quantify how improvements in imputation accuracy translate into downstream prediction performance, we fit a linear regression using these 40 points for each model and dataset[1]. Figure 3 gives two examples of such fit: on the `Bike_Sharing_Demand` dataset, for a missing rate of 50%, the prediction R^2 increases as a function of the imputation R^2 ; the effect is greater for the MLP, for which the fit gives a slope of 0.24, than for the MLP with indicator for which the slope is only 0.03. Figure 4 summarizes the slopes estimated using the aforementioned methodology across all datasets, predictors with and without the indicator, and varying missing rates. Firstly, the fact that most slopes are positive indicates that better imputations correlate with better predictions, aligning with common beliefs. However, this observation should be nuanced by the size of the effects.

Gains in prediction R^2 are 10% or less of the gains in imputation R^2 . Figure 4 shows that the slopes are typically small, rarely exceeding 0.1. This implies that an improvement of 0.1 in imputation R^2 typically leads to an improvement in prediction R^2 that is 10 times smaller, i.e. a gain of 0.01 in prediction R^2 , or even less. For XGBoost, the average slope across datasets in rather close to 0.025 or less (even zero without the mask at 20% missing rate). Thus, an enhancement of 0.3 in

[1]The repetition identifier is also used as a covariate in the linear regression to account for the effects of the various train/test splits on prediction performance.

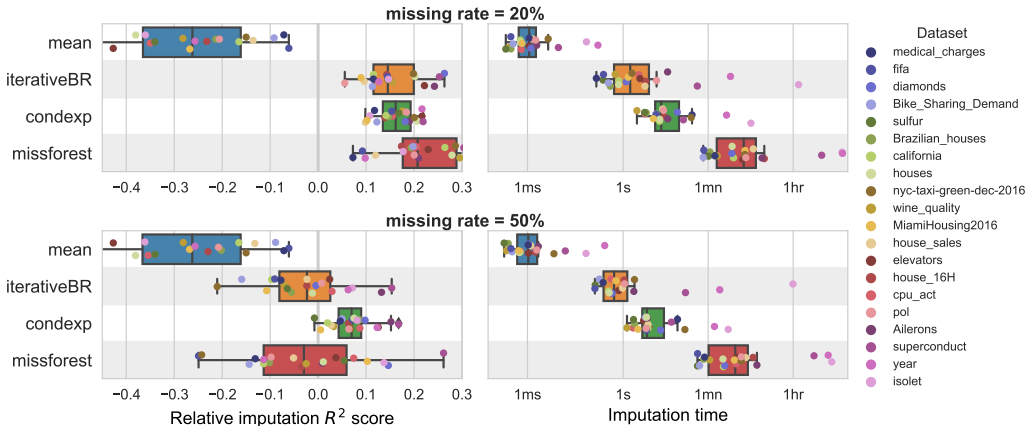

Figure 2: **Left: Imputer performance for recovery.** Performances are given as R^2 scores for each dataset relative to the mean performance across imputation techniques. A negative value indicates that a method perform worse than the average of other methods. **Right: Imputation time.**

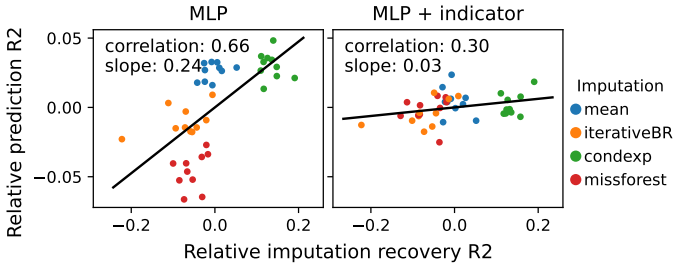

Figure 3: **Example fit of prediction performance as a function of imputation accuracy**, for the `Bike_Sharing_Demand` dataset and a missing rate of 50%: on the left using an MLP as predictor, and on the right an MLP with missingness indicator.

imputation R^2 , which represents the average difference between the best of the worst imputer in this scenario (mean vs condexp in fig. 2), implies a gain in prediction R^2 of only 0.0075.

Good imputations matter less for more expressive predictors. Comparison between models shows a decrease in slope from MLP to SAINT, to XGBooost. These results illustrate the idea that a powerful model can compensate for the simplicity or inaccuracy of an imputation (in our case, the MLP can be considered the least expressive model, and XGBoost the most expressive). Le Morvan et al. [2021] gives a formal proof in an extreme case: given enough samples, a sufficiently expressive model can always build a Bayes optimal predictor even on the simplest imputations (e.g. a constant).

Good imputations matter less when adding the indicator. Figure 4 shows that adding the missingness indicator clearly decreases the effect size: imputing better has less impact on performance when the indicator is used (we discuss this effect further in section 4.4).

Good imputations matter less when the response is non-linear. When the response y is a linear function of the input X, the best predictor can be built using a linear model on the most accurate simple imputation. However, when the responses are non-linear, it may be difficult to learn the best possible predictor even with the most accurate imputation [Le Morvan et al., 2021]. There are thus reasons to believe that response non-linearity, which is common in real data, alters the relationship between imputation accuracy and prediction performance. To investigate this, we compare the real datasets with matching semi-simulated datasets where y is simulated as a linear function of the input X. We also measure correlation[2] (fig. 5) in addition to the slope, to quantify the reliability of the association: correlation captures not only the effect size (slope) but also the amount of noise (appendix C recalls this classic result) in the relationship. While the effects are similar between real and linear outcomes (fig. 11 gives effects in the semi-simulated case), the correlation between imputation accuracy and prediction performance, averaged across all datasets, is systematically smaller for real outcomes than for linear ones (fig. 5). The average decrease in correlation lies between 0.1 and 0.3 across models. Moreover, the variance in correlations for real outcomes is

[2]Specifically we use the partial correlation, partialing out the effect of repeated train/test splits.

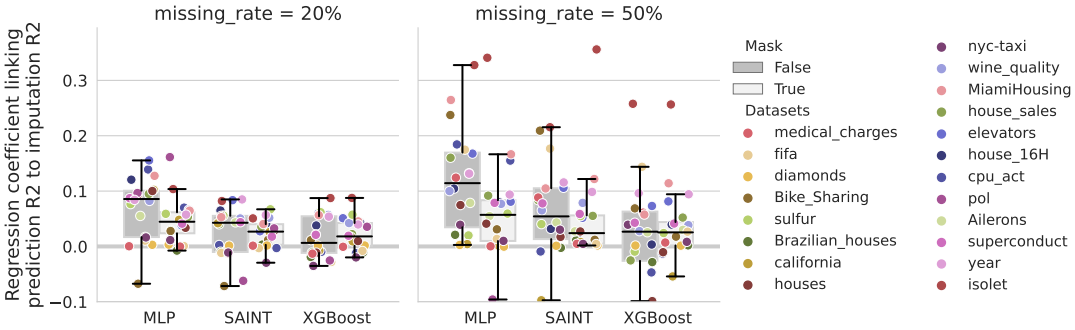

Figure 4: **Effect of the imputation recovery on the prediction performance.** We report the slope of the regression line where imputation quality is used to predict prediction performance.

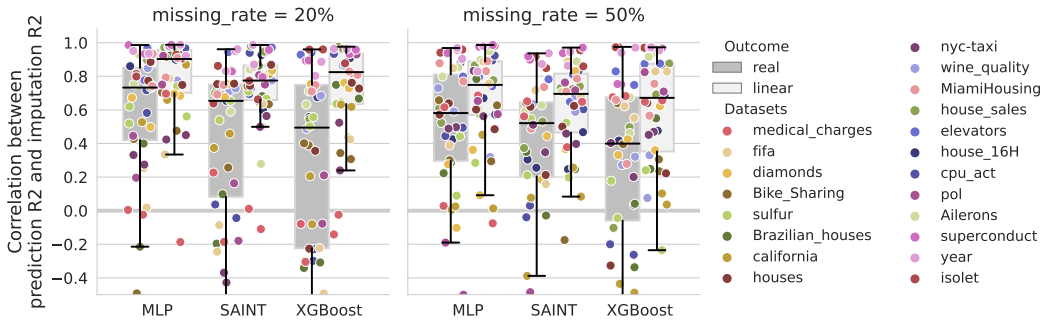

Figure 5: **Correlation between imputation quality and prediction performance.** A correlation close to 1 indicates that the quality of imputations is stronly associated to the quality of predictions, while a correlation close to zero means that the quality of predictions is not linked to the quality of imputations. Each correlation is computed using 40 different imputation/performance pairs, made of 4 imputation methods (mean, iterativeBR, missforest, condexp) repeated 10 times.

much larger, with many datasets with a near-zero correlation. This shows that the gains expected in prediction from better imputation are much more reliably achieved when the response is linear.

4.4 Why is the indicator beneficial, even with MCAR data?

In general, we find that adding the missingness indicator really helps prediction. While it is expected that adding the indicator is beneficial in MNAR scenarios, as the missingness is informative, it is less obvious in the MCAR settings studied here. Indeed, the indicator contains absolutely no relevant information for predicting the outcome. To the best of our knowledge, the benefit of using an indicator in MCAR has not yet been established. Below, we propose a theoretical insight to explain this finding.

The best possible predictor in the presence of missingness can always be expressed as the composition of an imputation and a prediction function [Le Morvan et al., 2021]. But, in general, the best prediction function on the imputed data can be challenging to learn, even for perfect conditional imputation. In fact, it often displays discontinuities on imputed points. We hypothesize that adding the missingness indicator simplifies modeling functions that exhibit discontinuities at these points, as the indicator can act as a switch to encode these discontinuities.

The case of XGBoost in Figure 1 illustrates the importance of keeping the missingness information encoded. For 50% missing rate, in the absence of an indicator, no imputation really benefits prediction with XGBoost, and the best option is to use the native handling of missing values. This suggests that XGBoost benefits from knowing which values are missing. With advanced imputations, distinguishing between imputed and observed values becomes challenging. Appending the indicator to the imputed data reinstates the missingness information unambiguously, which enables XGBoost to benefit from more advanced imputations, in particular missforest.

5 Conclusion

For prediction, imputation matters but marginally. Prior theoretical work showed that in extreme cases (asymptotics), imputation does not matter for predicting with missing values. We quantified empirically the effect of imputation accuracy gains on prediction performance across many datasets and scenarios. We show that in practice, imputation does play a role. But various factors modulate the importance of better imputations for prediction: investing in better imputations will be less beneficial when a flexible model is used, when a missing-value indicator is used, and if the response is thought to be non-linear. These results are actually in line with the theoretical results suggesting that imputation does not matter, as these hold for very flexible models (ie universally consistent). A notable new insight is that adding a missing-value indicator as input is beneficial for prediction performances *even for MCAR settings*, where missingness is uninformative.

We show that large gains in imputation accuracy translate into small gains in prediction performance. These results were drawn from a favorable MCAR setting, and it is likely that with native missingness,

often Missing Non At Random (MNAR), the performance gains are even smaller. As novel imputation methods usually provide small gains in imputation accuracy compared to the state-of-the-art, the corresponding gains in downstream prediction tasks are likely to be even smaller.

There are multiple potential reasons why imputation gains do not always correlate with performance gains. For instance, some features may be well recovered, but not useful in the prediction because they are not predictive. Or even with accurate imputations, it may still be difficult to learn a predictor that performs well for all missing data patterns. Finally, the imputation accuracy is also probably an imperfect measure of the potential gains in prediction: in our experiments on 50% missing rate, missforest and iterativeBR performs comparable on average yet missforest-based predictors tend to outperform those based on iterativeBR.

Limitations and future work. A remaining question is whether better imputations matter more for higher-dimensional datasets. In our study, the 3 datasets with higher dimension (superconduct, isolet and year) show high correlations of imputation with prediction gains, and larger effects, notably for isolet. It would also be useful to investigate whether imputations based on random draws outperform deterministic imputations for downstream prediction tasks, and the usefulness of multiple imputations [Perez-Lebel et al., 2022]. A related question is whether reconstructing well the data distribution is important for better predictions. Shadbahr et al. [2023] shows that it does not seem crucial for classification performances but may compromise more seriously model interpretability. Finally, adding the indicator to the input enables better predictors to be learned, yet it may not be the best way to represent the missingness information, and further investigation is needed in this direction.

Outlook We have seen that often, improving imputation is a difficult way of improving prediction. On top of imputation, future research could focus more on developing advanced modeling techniques that can inherently handle missing values and effectively incorporate missingness indicators to improve predictive performance.

References

Cathie Sudlow, John Gallacher, Naomi Allen, Valerie Beral, Paul Burton, John Danesh, Paul Downey, Paul Elliott, Jane Green, Martin Landray, et al. Uk biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age. *PLoS medicine*, 12(3): e1001779, 2015.

Lynn A Blewett, Julia A Rivera Drew, Miriam L King, Kari CW Williams, Natalie Del Ponte, and Pat Convey. Ipums health surveys: National health interview survey. *Minneapolis, MN: IPUMS*, 2019.

Alexandre Perez-Lebel, Gaël Varoquaux, Marine Le Morvan, Julie Josse, and Jean-Baptiste Poline. Benchmarking missing-values approaches for predictive models on health databases. *GigaScience*, 11:giac013, 2022.

Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning*, pages 5689–5698. PMLR, 2018.

Pierre-Alexandre Mattei and Jes Frellsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International conference on machine learning*, pages 4413–4423. PMLR, 2019.

Boris Muzellec, Julie Josse, Claire Boyer, and Marco Cuturi. Missing data imputation using optimal transport. In *International Conference on Machine Learning*, pages 7130–7140. PMLR, 2020.

Daniel Jarrett, Bogdan C Cebere, Tennison Liu, Alicia Curth, and Mihaela van der Schaar. Hyperimpute: Generalized iterative imputation with automatic model selection. In *International Conference on Machine Learning*, pages 9916–9937. PMLR, 2022.

Marine Le Morvan, Julie Josse, Erwan Scornet, and Gael Varoquaux. What's a good imputation to predict with missing values? In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 11530–11540. Curran Associates, Inc., 2021.

Julie Josse, Nicolas Prost, Erwan Scornet, and Gaël Varoquaux. On the consistency of supervised learning with missing values. *arXiv preprint arXiv:1902.06931*, 2019.

Alexis Ayme, Claire Boyer, Aymeric Dieuleveut, and Erwan Scornet. Naive imputation implicitly regularizes high-dimensional linear models. In *International Conference on Machine Learning*, pages 1320–1340. PMLR, 2023.

Alexis Ayme, Claire Boyer, Aymeric Dieuleveut, and Erwan Scornet. Random features models: a way to study the success of naive imputation. *arXiv preprint arXiv:2402.03839*, 2024.

George Paterakis, Stefanos Fafalios, Paulos Charonyktakis, Vassilis Christophides, and Ioannis Tsamardinos. Do we really need imputation in automl predictive modeling? *ACM Transactions on Knowledge Discovery from Data*, 2024.

Tolou Shadbahr, Michael Roberts, Jan Stanczuk, Julian Gilbey, Philip Teare, Sören Dittmer, Matthew Thorpe, Ramon Viñas Torné, Evis Sala, Pietro Lió, et al. The impact of imputation quality on machine learning classifiers for datasets with missing values. *Communications Medicine*, 3(1):139, 2023.

Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

Sebastian Jäger, Arndt Allhorn, and Felix Bießmann. A benchmark for data imputation methods. *Frontiers in big Data*, 4:693674, 2021.

Burim Ramosaj, Justus Tulowietzki, and Markus Pauly. On the relation between prediction and imputation accuracy under missing covariates. *Entropy*, 24(3):386, 2022.

Katarzyna Woźnica and Przemysław Biecek. Does imputation matter? benchmark for predictive models. *arXiv preprint arXiv:2007.02837*, 2020.

Jason Poulos and Rafael Valle. Missing data imputation for supervised learning. *Applied Artificial Intelligence*, 32(2):186–196, 2018.

JiaHang Li, ShuXia Guo, RuLin Ma, Jia He, XiangHui Zhang, DongSheng Rui, YuSong Ding, Yu Li, LeYao Jian, Jing Cheng, et al. Comparison of the effects of imputation methods for missing data in predictive modelling of cohort study datasets. *BMC Medical Research Methodology*, 24(1):41, 2024.

Dimitris Bertsimas, Colin Pawlowski, and Ying Daisy Zhuo. From predictive methods to missing data imputation: an optimization approach. *Journal of Machine Learning Research*, 18(196):1–39, 2018.

Mike Van Ness, Tomas M Bosschieter, Roberto Halpin-Gregorio, and Madeleine Udell. The missing indicator method: From low to high dimensions. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5004–5015, 2023.

Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45:1–67, 2011.

Stef Van Buuren. *Flexible imputation of missing data*. CRC press, 2018.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Samuel F Buck. A method of estimation of missing values in multivariate data suitable for use with an electronic computer. *Journal of the Royal Statistical Society: Series B (Methodological)*, 22(2): 302–306, 1960.

Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342*, 2021.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? *Advances in Neural Information Processing Systems*, 35: 507–520, 2022.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

Bheki ETH Twala, MC Jones, and David J Hand. Good methods for coping with missing data in decision trees. *Pattern Recognition Letters*, 29(7):950–956, 2008.

Supplementary materials – Imputation for prediction: beware of diminishing returns.

A List of datasets.

This benchmark was created by Grinsztajn et al. [2022], and is available on OpenML at `https://www.openml.org/search?type=benchmark&study_type=task&sort=tasks_included&id=297`.

Table 1: **Datasets dimensions.**

dataset	d	n_train	n_test
house_16H	16	18185	2274
cpu_act	21	6553	820
elevators	16	13279	1661
wine_quality	11	5197	651
Brazilian_houses	8	8553	1070
house_sales	15	17290	2162
sulfur	6	8064	1009
Ailerons	33	11000	1375
Bike_Sharing_Demand	6	13903	1739
california	8	16512	2064
diamonds	6	43152	5394
fifa	5	14450	1807
houses	8	16512	2064
isolet	613	6237	781
medical_charges	3	50000	50000
MiamiHousing2016	13	11145	1394
nyc-taxi-green-dec-2016	9	50000	50000
pol	26	12000	1500
superconduct	79	17010	2127
year	90	50000	50000

B Hyperparameter search spaces

Table 2: **XGBoost hyperparameter space.** We used the `XGBRegressor` from the `xgboost` Python library. The hyperparameters optimized are commonly accepted as the most important ones. The variation ranges are inspired by the ones used in Grinsztajn et al. [2022], while the default hyperparameters are those of the `xgboost` library.

parameter	range	log scale	default
n_estimators	$[100, 2000]$	no	100
max_depth	$[1, 6]$	no	6
learning_rate	$[10^{-5}, 0.7]$	yes	0.3
reg_alpha	$[10^{-8}, 10^{2}]$	yes	10^{-8}
reg_lambda	$[1, 4]$	yes	1
early_stopping_rounds	-	-	20

Table 3: **MLP hyperparameter space.** We implemented the MLP in PyTorch. The parameter d for the width of the MLP represents the number of features. When $d > 1024$, the width is taken equal to the number of features d.

	parameter	range	default
MLP	depth	$[\![0, 6]\!]$	3
	width	$[d, \min(10d, 1024)]$	3d
	dropout rate	$[0, 0.5]$	0.2
Optimizer	name	-	AdamW
	weight decay	-	10^{-6}
	learning rate	-	10^{-3}
Scheduler	name	-	`ReduceLROnPlateau`
	factor	-	0.2
	patience	-	10
	threshold	-	10^{-4}
General	max nb. epochs	-	2000
	early stoppping	-	Yes
	batch size	-	256

Table 4: **SAINT default hyperparameters.** We used the implementation provided by Somepalli et al. [2021]. d refers to the number of features of the dataset. We did not use a scheduler with SAINT. We followed the default configuration provided by the paper introducing SAINT [Somepalli et al., 2021] when there is both intersample and feature attention (i.e. attention_type = `'colrow'`).

	parameter	default
SAINT	dim	32 if $d < 70$, 16 if $d \in [70, 200]$, 4 if $d \geq 200$
	depth	1
	heads	4
	attn_dropout	0.8
	ff_dropout	0.8
	attentiontype	colrow
Optimizer	name	AdamW
	weight decay	10^{-2}
	learning rate	10^{-4}
General	max nb. epochs	100
	early stopping	yes
	batch size	256

C Link between correlation and effect size.

For completeness, we recall below the relationship between correlation and effect size.

Proposition C.1 (Link between correlation and effect size.)**.** *Let $X \in \mathbb{R}$ be a random variable, and $\beta \in \mathbb{R}$ a parameter. Furthermore, define:*

$$Y = \beta X + \epsilon \quad \textit{where} \quad \mathbb{E}[\epsilon|X] = 0,\ \textit{var}\,(\epsilon) = \sigma^2.$$

Then:

$$\textit{cor}\,(X, Y) = \frac{1}{\sqrt{1 + \frac{\sigma^2}{\beta^2 \textit{var}(X)}}}$$

Proof. Let's first derive the expression of the variance of Y:

$$\begin{aligned}
\text{Var}\,(Y) &= \mathbb{E}\left[(Y - \mathbb{E}[Y])^2\right] \\
&= \mathbb{E}\left[(\beta X + \epsilon - \beta\mathbb{E}[X])^2\right] \\
&= \mathbb{E}\left[(\beta\,(X - \mathbb{E}[X]))^2 + \epsilon^2\right] \\
&= \beta^2 \text{var}(X) + \sigma^2
\end{aligned}$$

It follows that:

$$\begin{aligned}
\text{cor}\,(Y, X) &= \frac{\mathbb{E}[(Y - \mathbb{E}[Y])\,(X - \mathbb{E}[X])]}{\sqrt{\text{var}(X)\,\text{var}(Y)}} \\
&= \frac{\mathbb{E}\left[\beta\,(X - \mathbb{E}[X])^2\right]}{\sqrt{\text{var}(X)\,\text{var}(Y)}} \\
&= \beta \frac{\sqrt{\text{var}(X)}}{\sqrt{\text{var}(Y)}} \\
&= \beta \frac{\sqrt{\text{var}(X)}}{\sqrt{\beta^2 \text{var}(X) + \sigma^2}} \\
&= \frac{1}{\sqrt{1 + \frac{\sigma^2}{\beta^2 \text{var}(X)}}}
\end{aligned}$$

□

Hence, in a case where the imputation accuracy X covers a wider range of values, i.e., $\text{var}(X)$ is larger, but the effect β and the noise σ^2 stay the same, then the correlation increases.

D Critical Difference diagrams.

Figures 6 to 9 give the **Critical Difference diagrams across all predictors and imputers** of average score ranks for a significance level of 0.05. The difference in ranks for all methods covered by the same black crossbar are not statistically significant according to a Nemenyi test for multiple pairwise comparisons. The colors encode the imputation type, the markers identify the model, and the line types encode the presence or absence of an indicator.

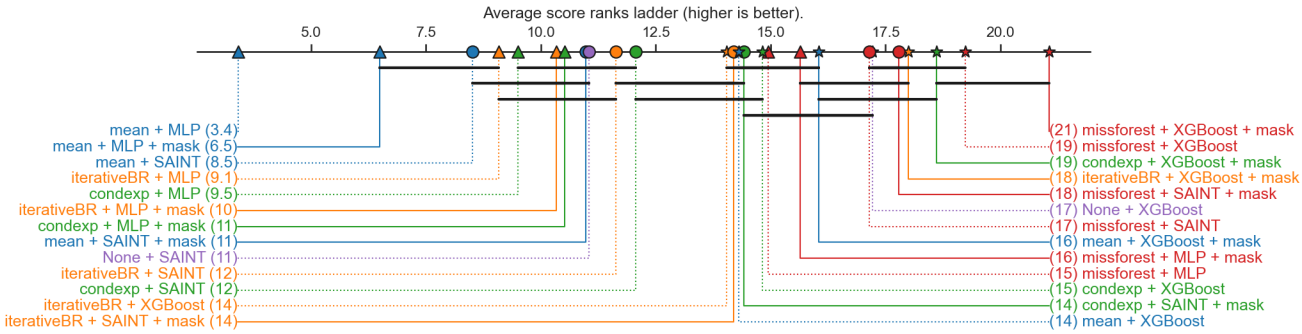

Figure 6: Critical Difference diagram - 20% missingness rate.

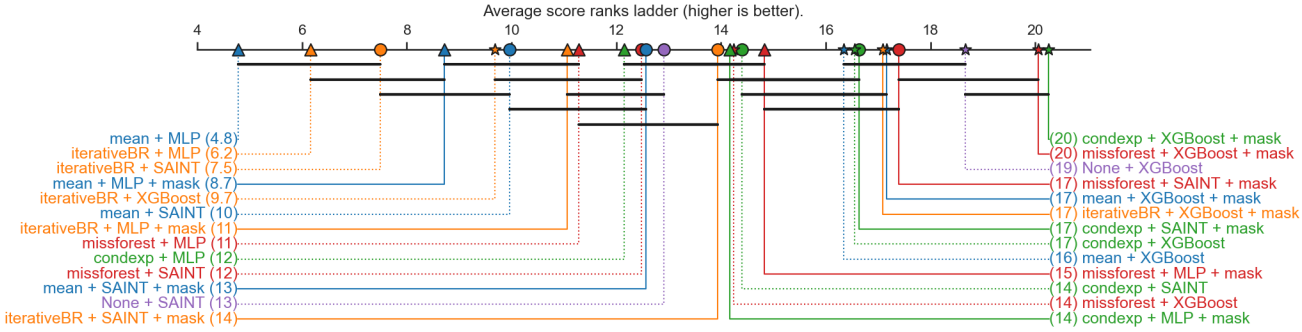

Figure 7: Critical Difference diagram - 50% missingness rate.

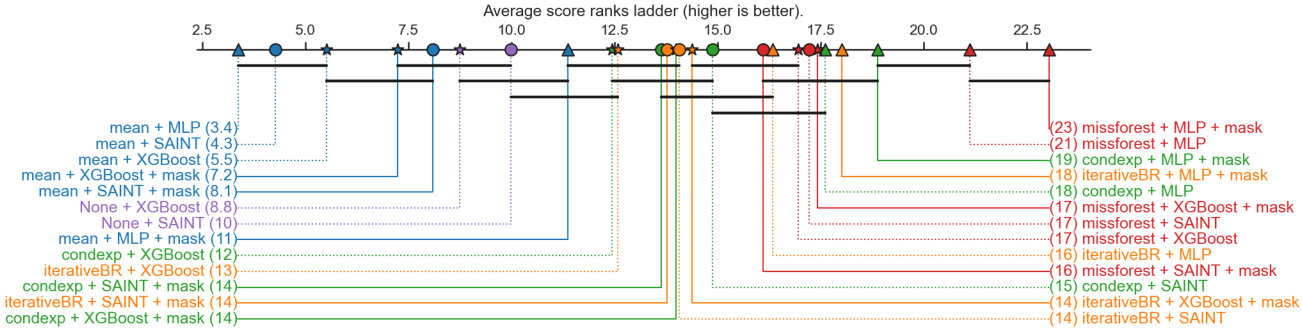

Figure 8: Critical Difference diagram - 20% missingness rate, semi-synthetic data with linear outcomes.

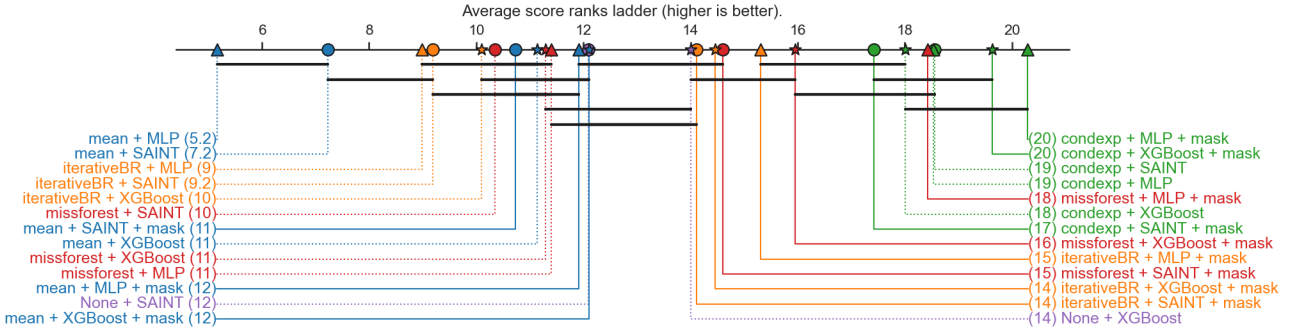

Figure 9: Critical Difference diagram - 50% missingness rate, semi-synthetic data with linear outcomes.

E Prediction performances for the semi-synthetic data.

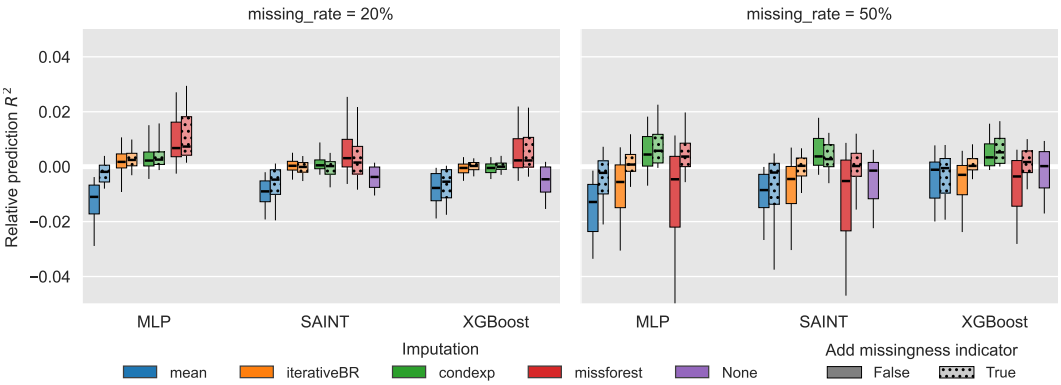

Figure 10: **Relative prediction performances for the semi-synthetic data with linear outcomes across datasets for different imputations, predictors, and use of the missingness indicator.** Each boxplot represents 200 points (20 datasets with 10 repetitions per dataset). The performances shown are R^2 scores on the test set relative to the mean performance across all models for a given dataset and repetition. A value of 0.01 indicates that a given method outperforms the average performance on a given dataset by 0.01 on the R^2 score.

F Regression slopes for the semi-synthetic data.

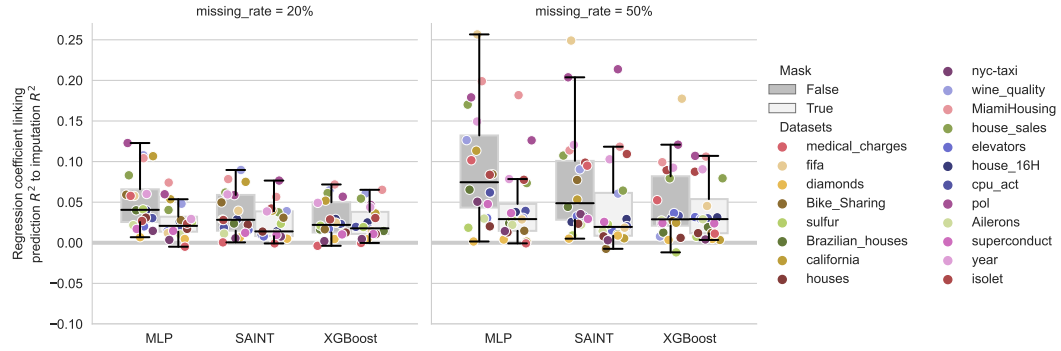

Figure 11: **Effect of the imputation recovery on the prediction performance** for the semi-synthetic data with linear outcomes. We report the slope of the regression line where imputation quality is used to predict prediction performance.

G Computation times per method.

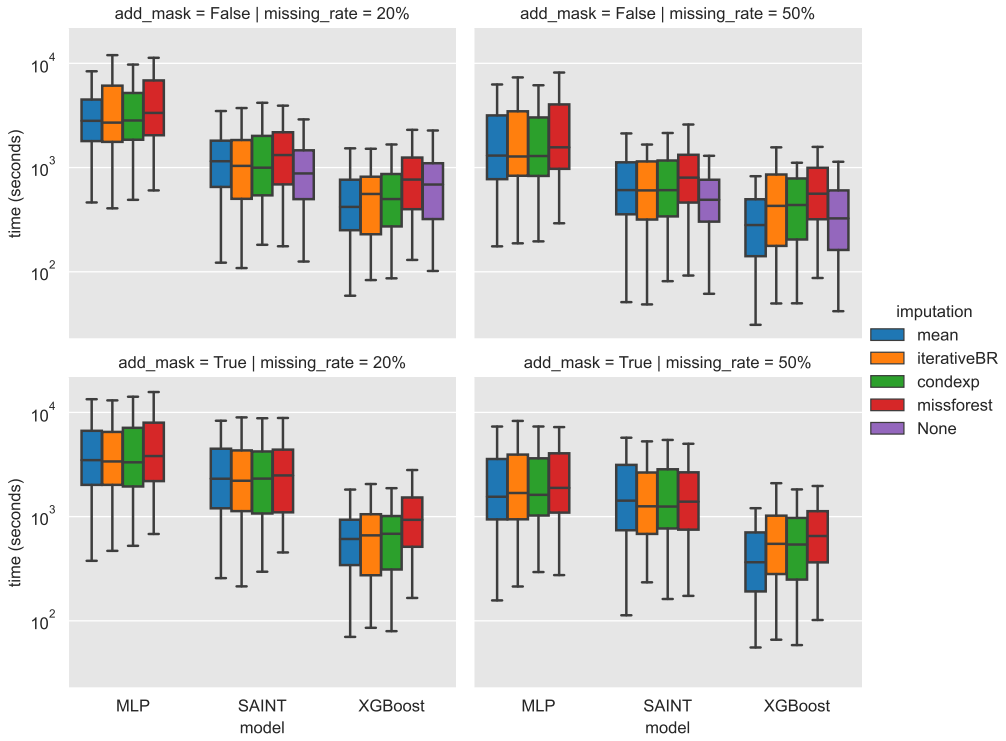

Figure 12: **Running time for each model**, including the 50 iterations of hyperparameter search for XGBoost and MLP.

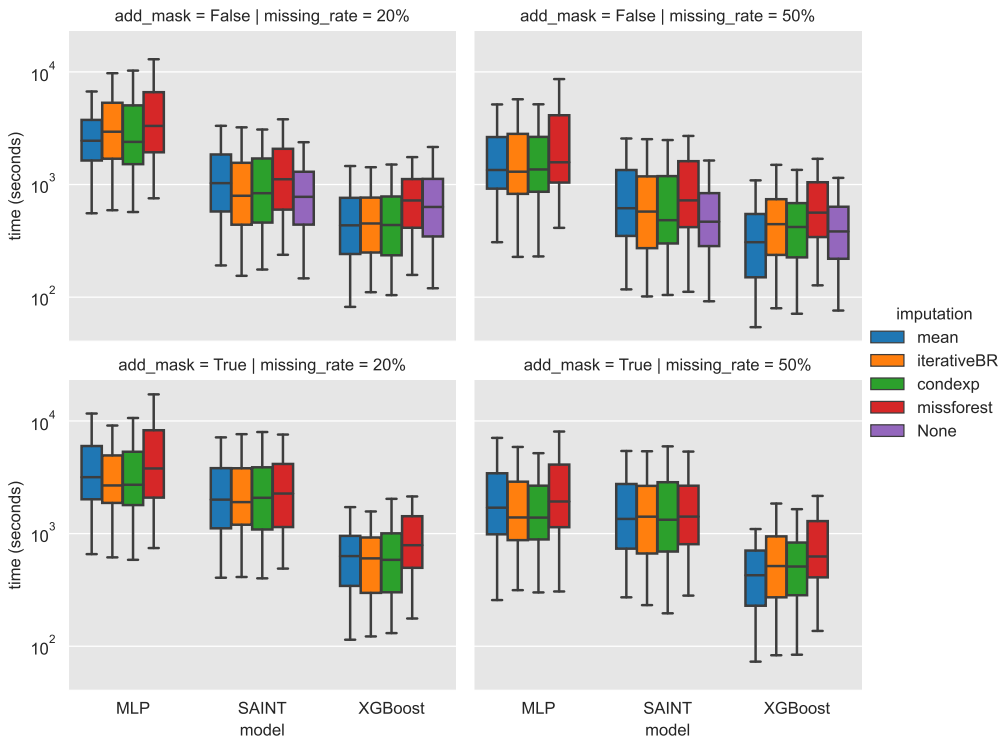

Figure 13: **Running time for each model** for the semi-synthetic data with linear outcomes, including the 50 iterations of hyperparameter search for XGBoost and MLP.

H Scatterplots of prediction R^2 vs imputation R^2 for each model and dataset.

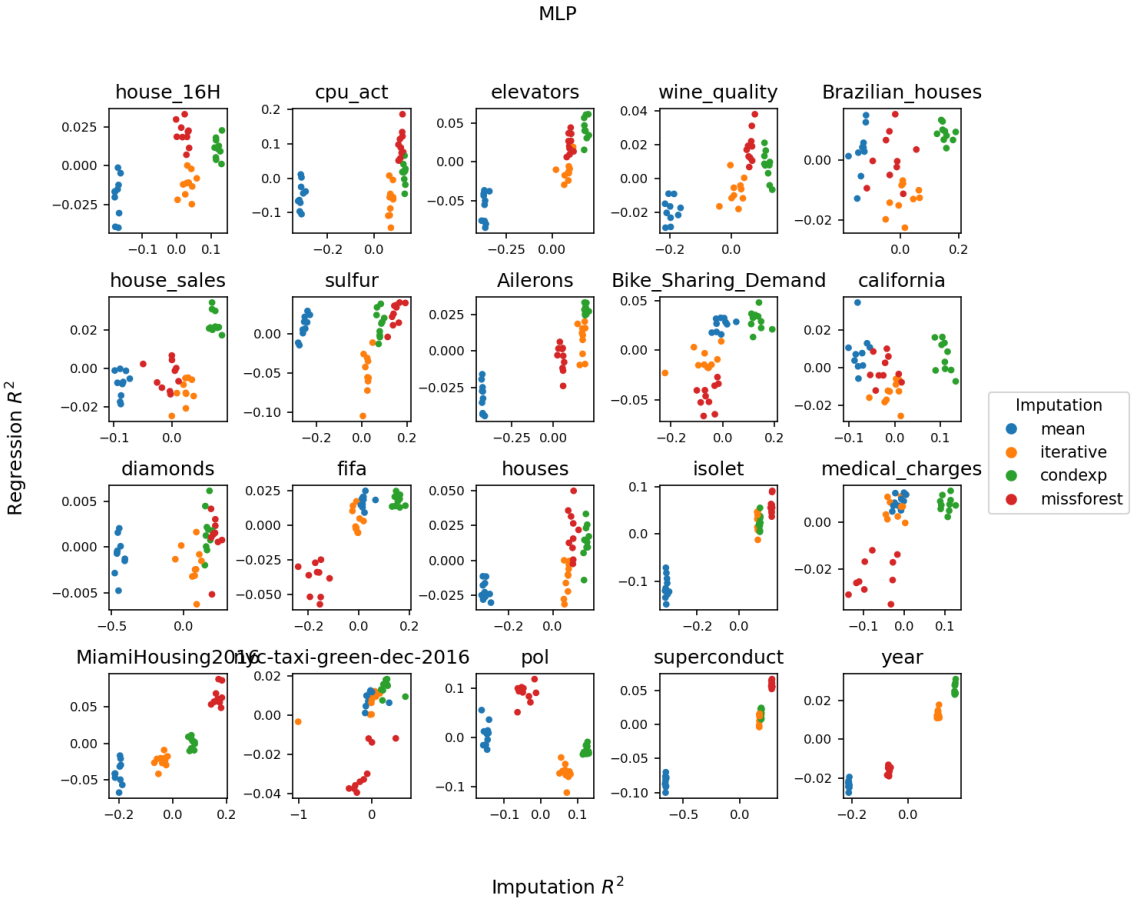

Figure 14: **Prediction R^2 vs imputation R^2 for a MLP** - missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

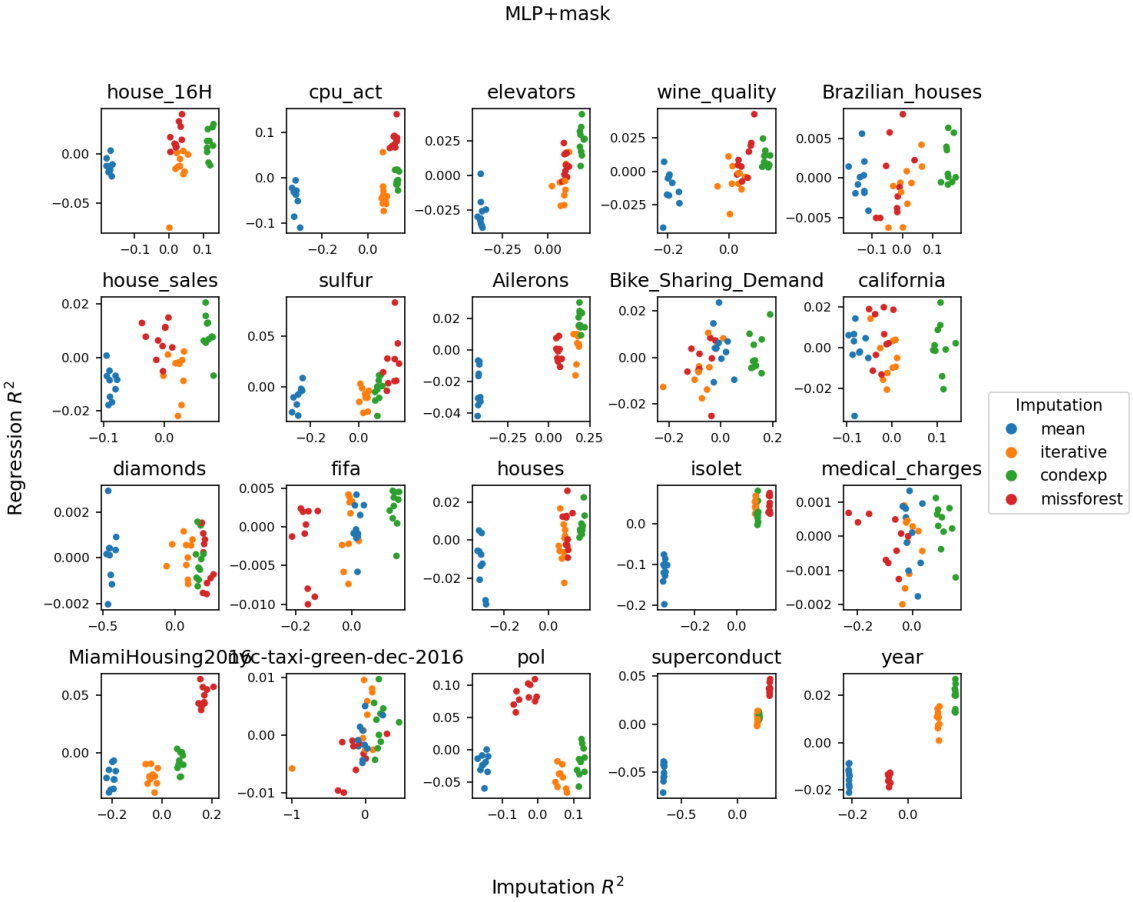

Figure 15: **Prediction R^2 vs imputation R^2 for a MLP + indicator** - missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

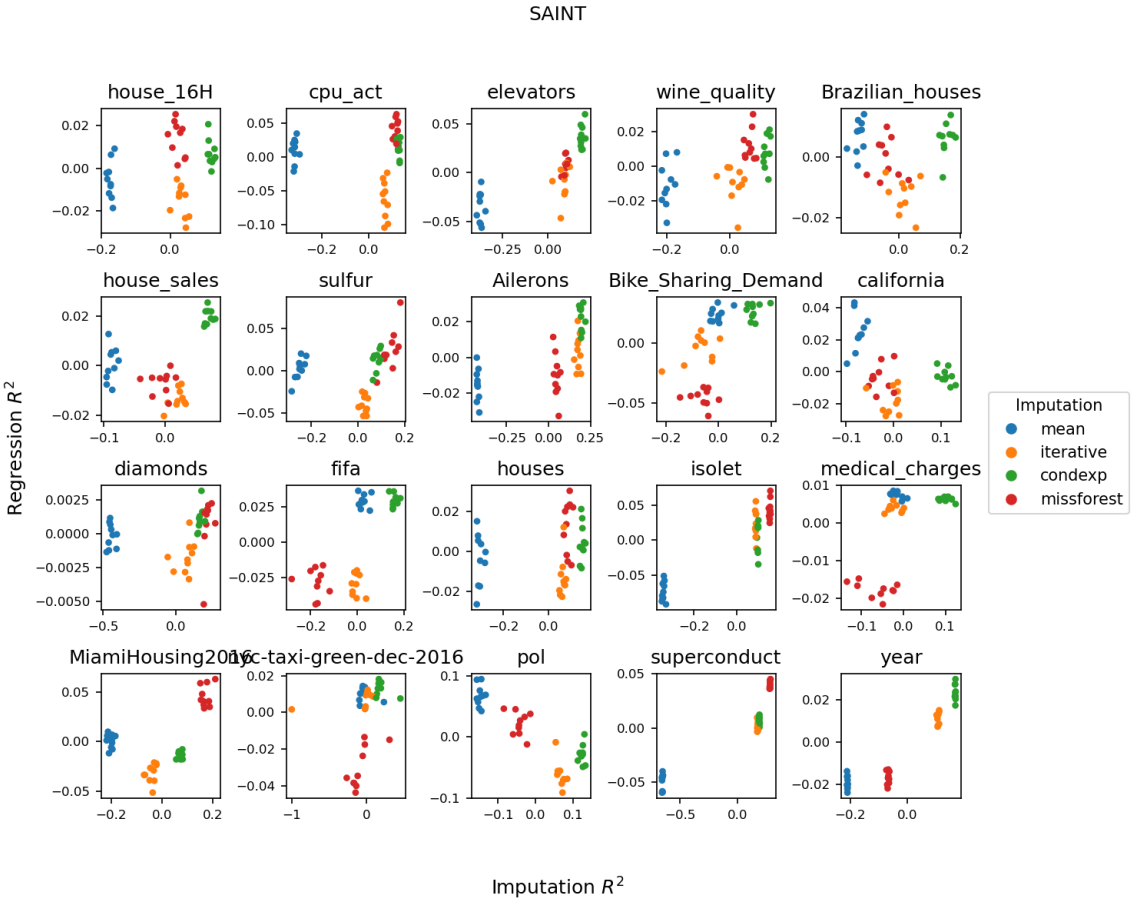

Figure 16: **Prediction R^2 vs imputation R^2 for SAINT** - missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

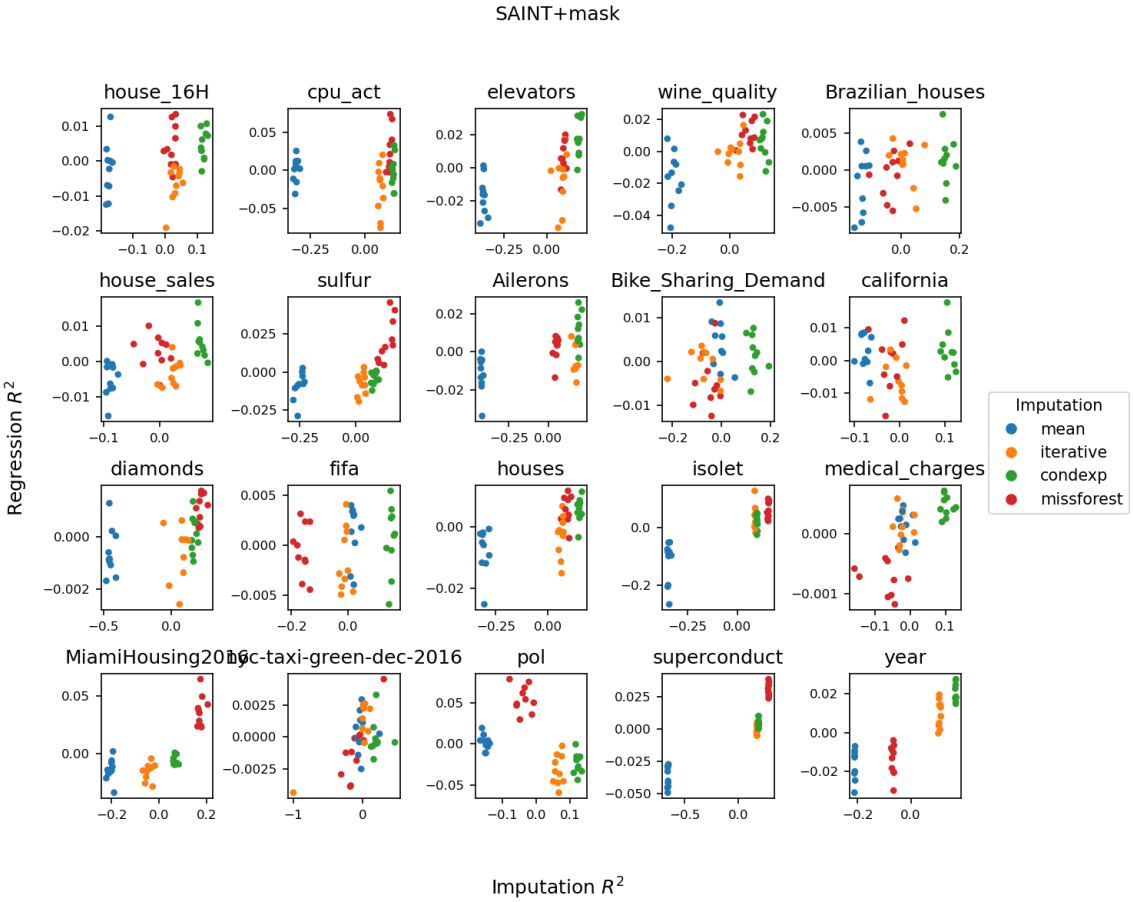

Figure 17: **Prediction R^2 vs imputation R^2 for SAINT + indicator** - missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

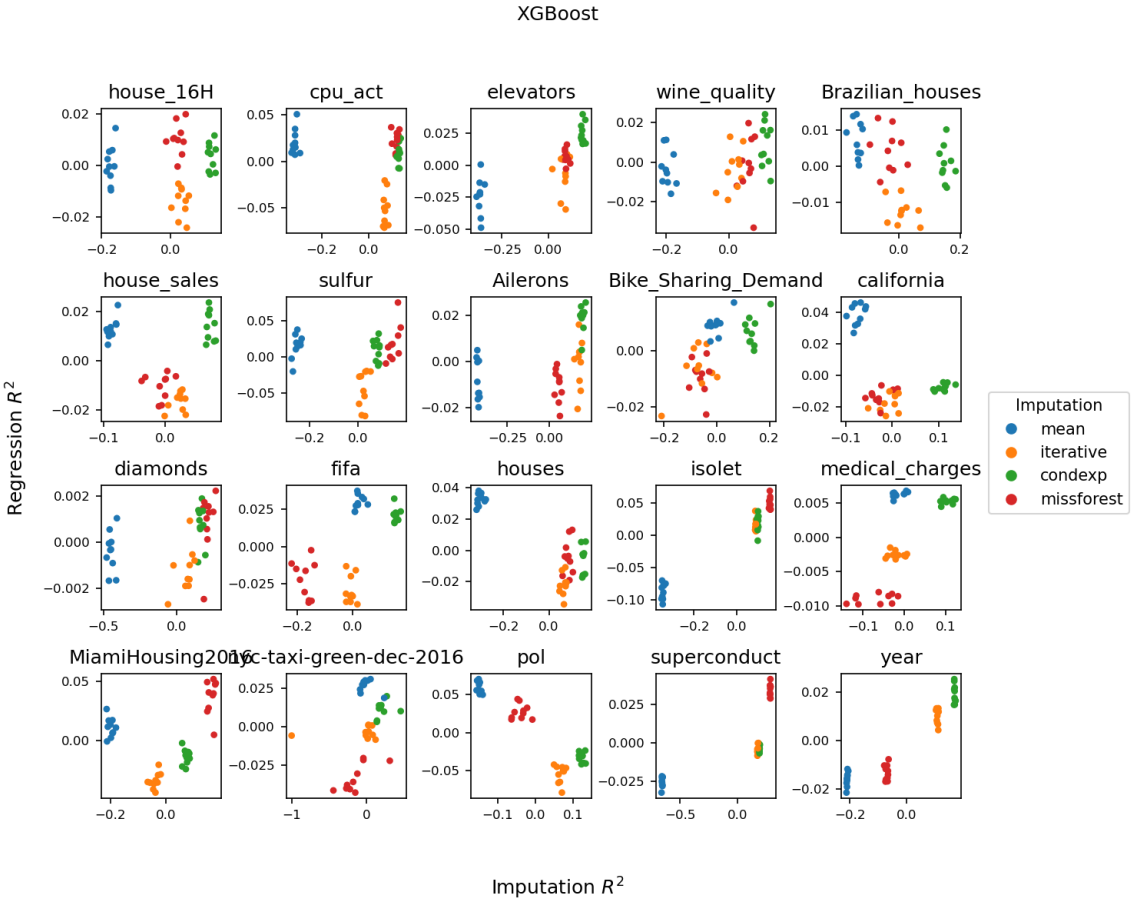

Figure 18: **Prediction R^2 vs imputation R^2 for XGBoost** - missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

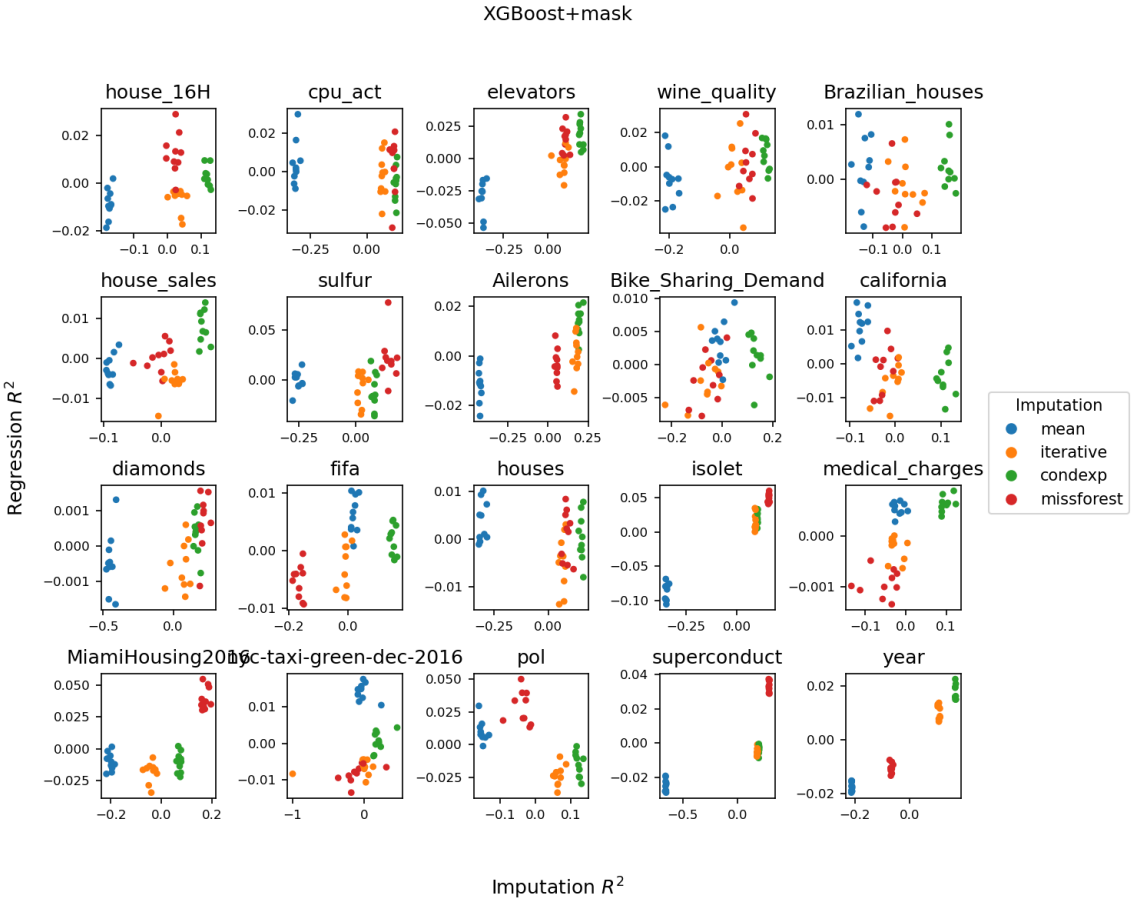

Figure 19: **Prediction R^2 vs imputation R^2 for XGBoost + indicator** - missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

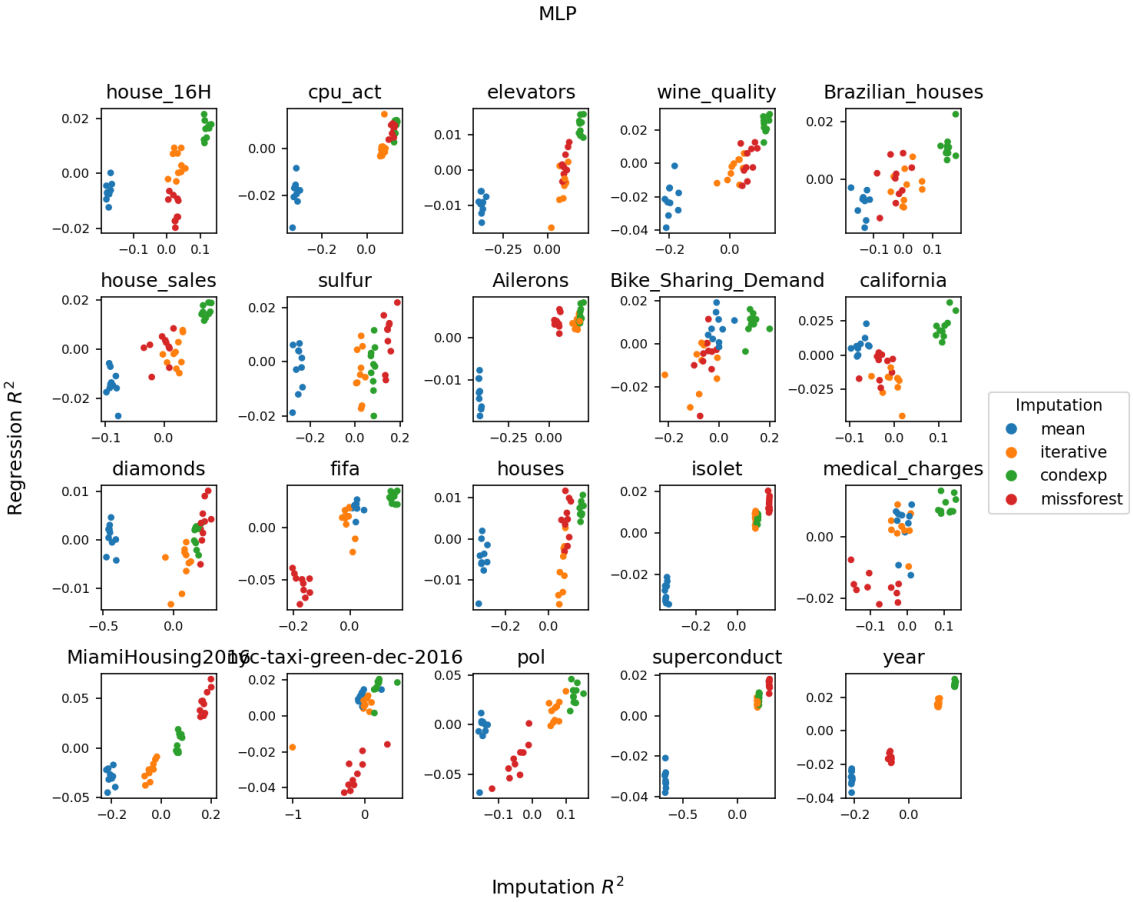

Figure 20: **Prediction R^2 vs imputation R^2 for a MLP** - semi-synthetic data with linear outcomes, missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

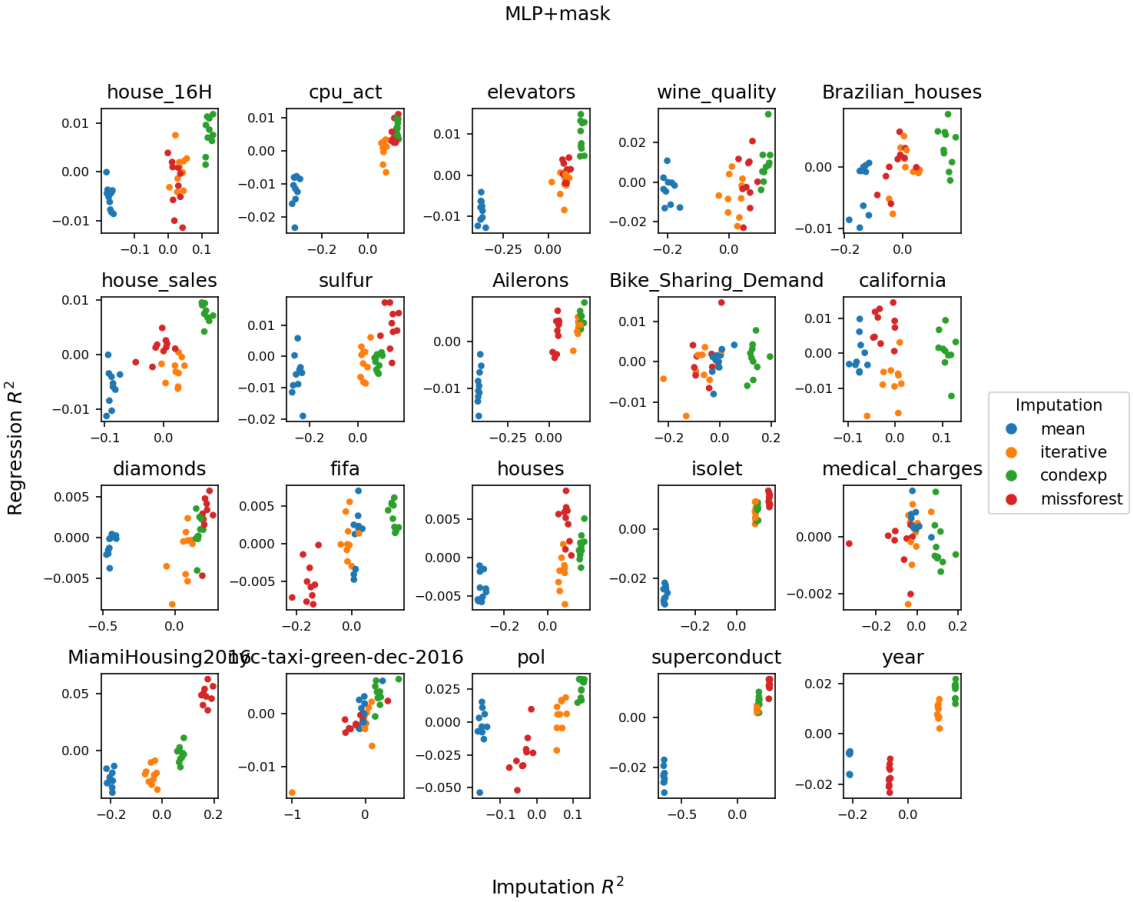

Figure 21: **Prediction R^2 vs imputation R^2 for a MLP + indicator** - semi-synthetic data with linear outcomes, missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

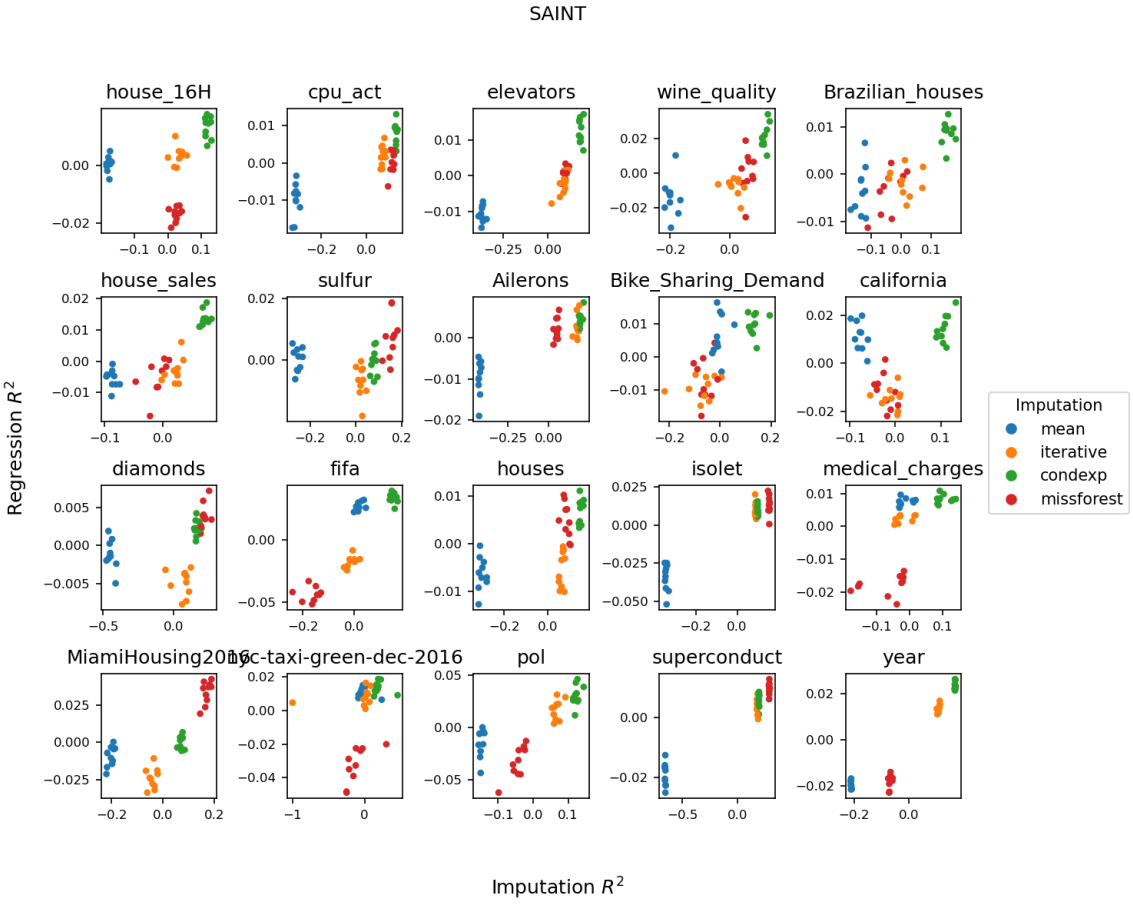

Figure 22: **Prediction R^2 vs imputation R^2 for SAINT** - semi-synthetic data with linear outcomes, missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

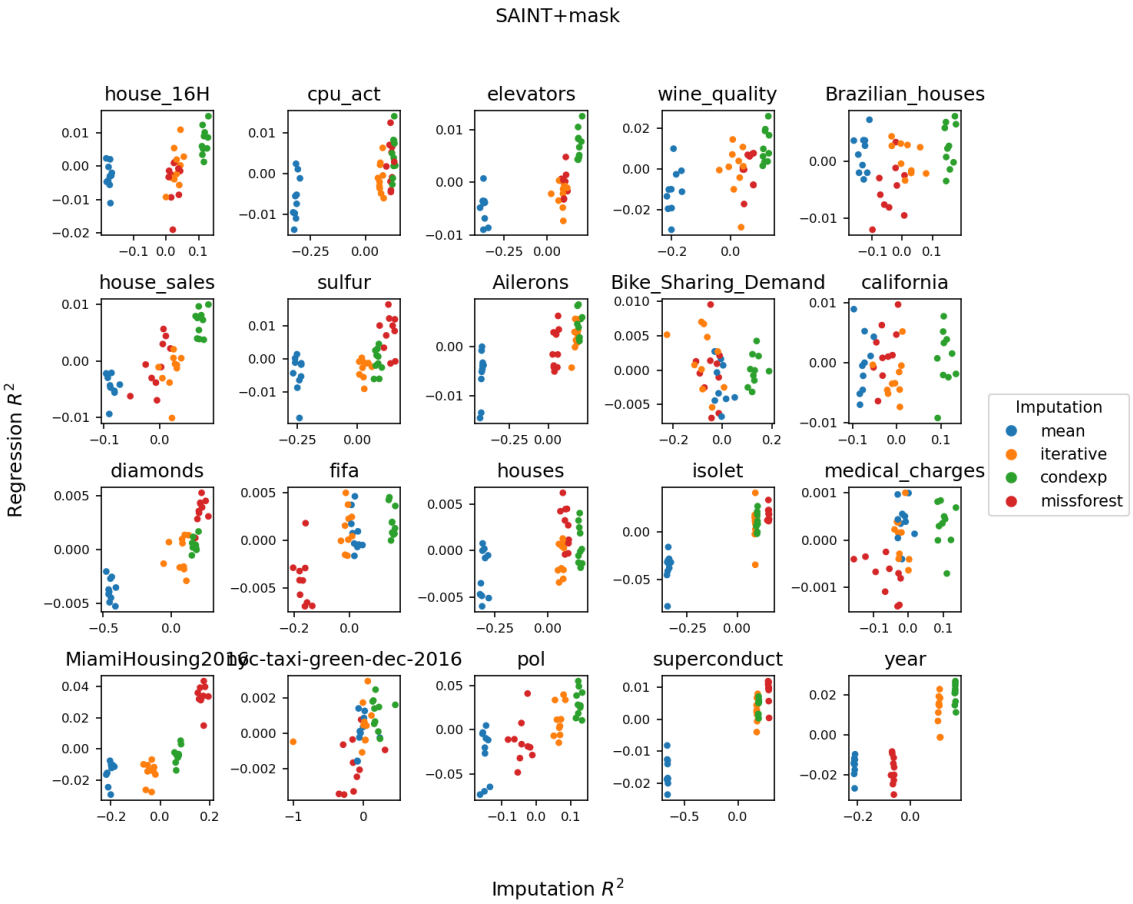

Figure 23: **Prediction R^2 vs imputation R^2 for SAINT + indicator** - semi-synthetic data with linear outcomes, missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

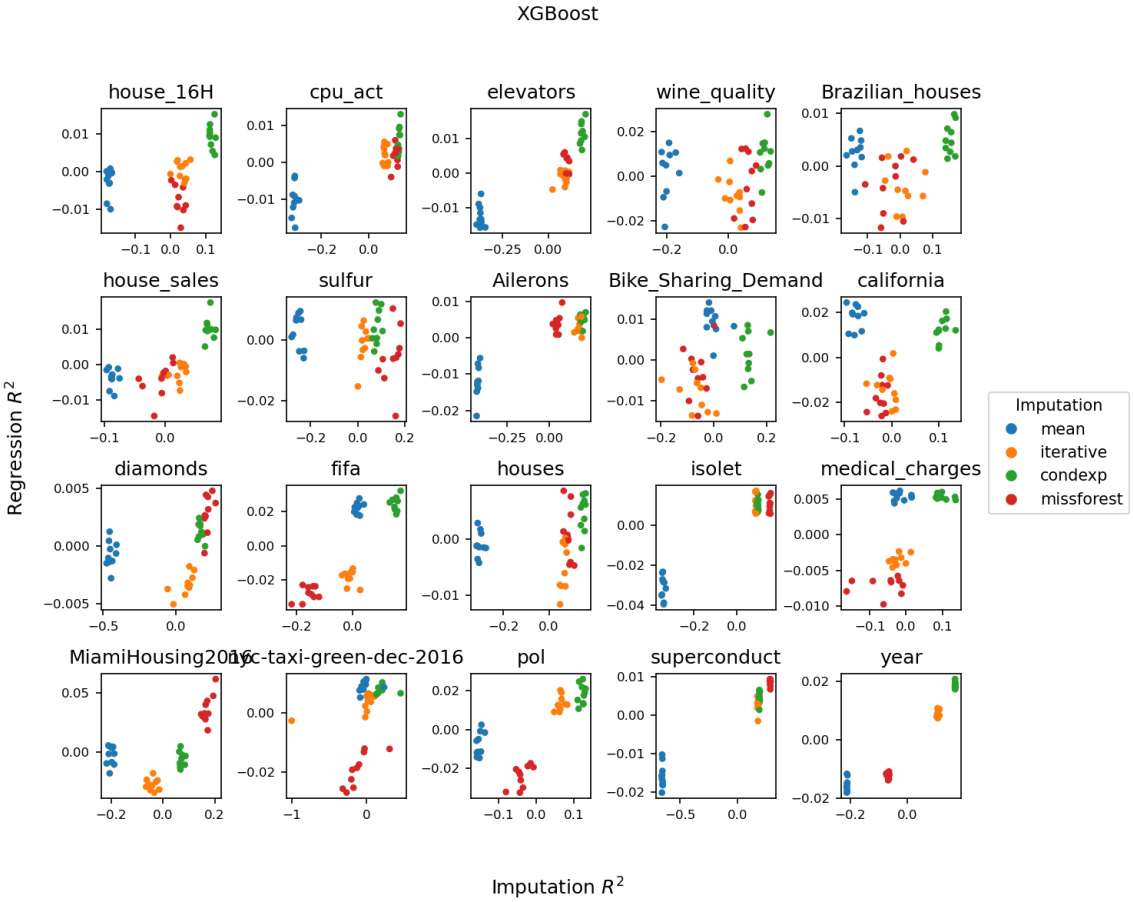

Figure 24: **Prediction R^2 vs imputation R^2 for XGBoost** - semi-synthetic data with linear outcomes, missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)

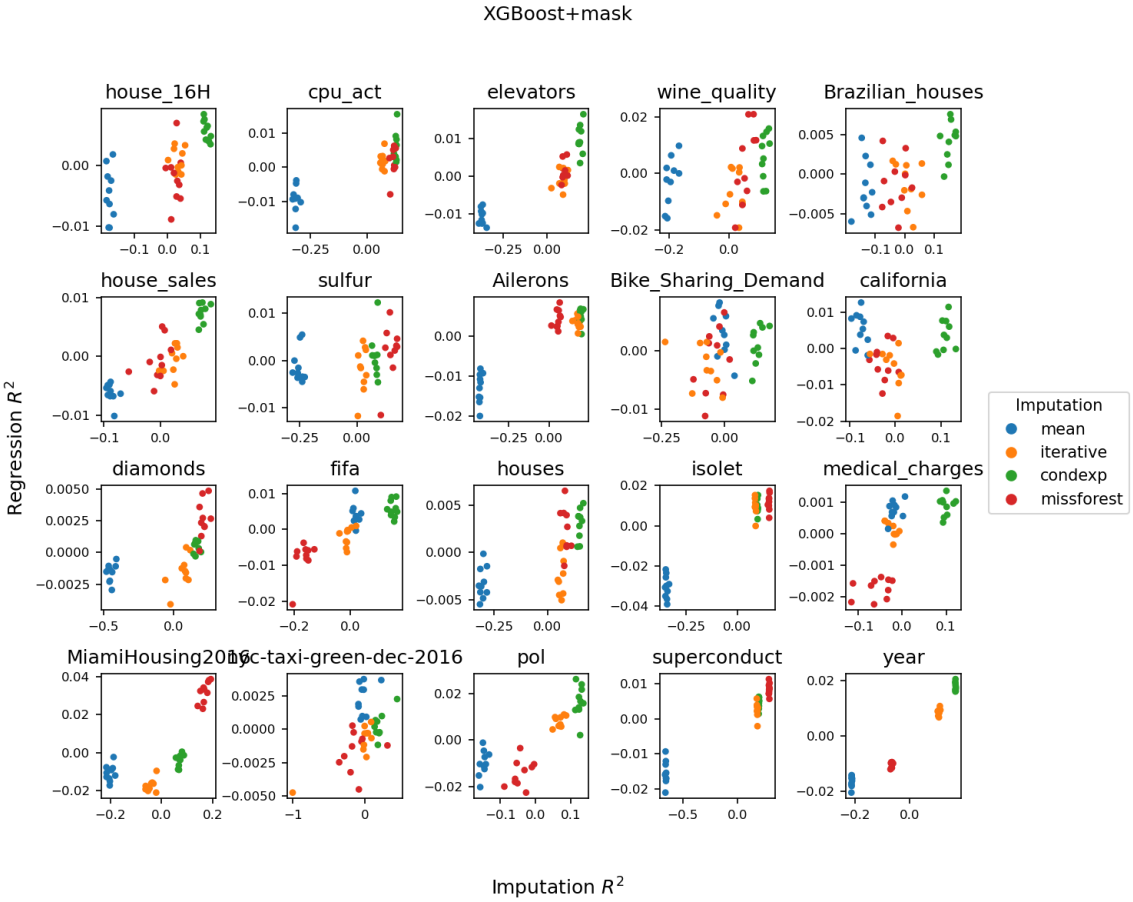

Figure 25: **Prediction R^2 vs imputation R^2 for XGBoost + indicator** - semi-synthetic data with linear outcomes, missing rate 50%. The R^2 scores are given relative to the mean R^2 score, with the effects of experiment repetitions eliminated (i.e. the effect of the train/test splits on the performance)