Minimal Repairs for Learning Over Incomplete Data

Anonymous Author(s)
Affiliation
Address
email

Abstract

Missing data often exists in real-world datasets, requiring significant time and effort for data repair to learn accurate machine learning (ML) models. In this paper, we show that imputing all missing values is not always necessary to achieve an accurate ML model. We introduce concepts of minimal and almost minimal repair, which are subsets of missing data items in training data whose imputation delivers accurate and reasonably accurate models, respectively. Imputing these sets can significantly reduce the time, computational resources, and manual effort required for learning models. We show that finding these sets is NP-hard for SVM and linear regression and propose efficient approximation algorithms with provable error bounds. Our extensive experiments indicate that our proposed algorithms can substantially reduce the time and effort required to learn on incomplete datasets.

1 Introduction

The performance of an ML model is highly dependent on the quality of its training data. In real-world data, a major data quality issue is missing or incomplete data [12, 18, 17, 8, 13]. There are two common approaches to address missing values in training data. The first approach involves deleting samples with missing values. However, this method can lead to the loss of important information and introduce bias [31]. Another popular approach is data repair or imputation, in which end-users or ML practitioners impute missing values with the correct ones [3, 10, 21, 22, 28, 24, 33, 34]. Accurate repair is often challenging and expensive as it usually requires extensive collaboration with expensive domain experts. It usually must be repeated whenever the dataset evolves.

To reduce the cost of imputation, significant effort has been made to train imputation models on the observed subset of the dataset that predict accurate values for missing data items [6, 19, 21, 29, 36, 38]. State-of-the-art models for data imputation may take a long time to process and predict values for missing data items, and those that use deep neural networks need costly computational resources [25, 38, 36]. As the dataset evolves, the user often has to repeat these steps. Moreover, in domains where important decisions must be made, e.g., healthcare and criminal justice, humans may need to manually verify the predictions of the imputation models [35]. Some users also distrust black-box model-based imputation techniques in critical applications and prefer to reason about missing data themselves using observed features and domain knowledge [2, 30]. In addition, model-based imputation may perform poorly when the ratio of missing data to observed data is too large [9, 14, 15]. In these settings, users may have to manually repair at least parts of the data.

To address these challenges, we introduce the concept of a **minimal repair** for a training dataset with missing values. Generally speaking, this set represents the smallest group of data items with missing values that, once repaired, yields the same model as that trained on a fully and accurately repaired dataset. By finding and imputing this set, users can significantly reduce the time and effort required to manually repair a dataset without sacrificing model accuracy. It also reduces the time and computational resources needed to predict missing values using imputation models and the manual labor required to verify their imputations. Moreover, minimal repair of a dataset pinpoints the subset

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

of the dataset whose uncertainty impacts the effectiveness of the model trained on the dataset. Hence, it simplifies the inspection and debugging of model training, which is often labor intensive [27]. Because incomplete data sets are prevalent and often evolve, a small reduction in time, effort, and computational resources in the preparation of training datasets can save significant resources in the long run. Specifically, our contributions are as follows.

- We define minimal repair for learning support vector machines (SVM) (Section 3) and linear regression (Section 4) over incomplete data. We prove that finding minimal repairs for SVM and linear regression is NP-hard and propose efficient algorithms with provable error bounds to approximate minimal repairs for them.
- Minimal repairs may sometimes be too large or take too long to find. We propose the concept of **almost minimal repair**, which is the minimal subset of data items with missing values whose repair delivers a model with a loss within a given threshold from the model trained over the fully and accurately repaired dataset. We prove that the problem of finding almost minimal repairs is NP-hard for SVM and linear regression and propose algorithms with provable error bounds to approximate almost minimal repairs for them (Section 5).
- We evaluated the scalability of our algorithms on multiple real-world datasets (Section 6). Our empirical results indicate that our proposed algorithms efficiently approximate minimal and almost minimal repairs and deliver models with the same or almost the same accuracy as those trained over fully repaired datasets. Our results also indicate that using minimal and almost minimal repairs can reduce the time of the model-based imputation methods for large data without losing accuracy in the downstream learning task.

2 Background

We model the training data as a table where each row represents a training sample. One column in the table represents labels and others represent the features of the samples. Given that the training data has d features, we denote its features as $[\mathbf{z}_1, \ldots, \mathbf{z}_d]$. The values of each feature belong to the *domain of the feature*, e.g., real numbers. To simplify our analysis, we assume that all the features share the same domain. Our results extend to other settings. A *training set* with n samples is a pair of a feature matrix $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^T$ and a corresponding label vector $\mathbf{y} = [y_1, ..., y_n]^T$. We denote each sample with d features in $\mathbf{X}$ as a vector $\mathbf{x}_i = [x_{i1}, ..., x_{id}]$, where x_{ij} represents the j^{th} feature in the i^{th} sample. Given the training set $(\mathbf{X}, \mathbf{y})$, the target function f, and the loss function L, the goal of training is to find an optimal model $\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathcal{W}} L(f(\mathbf{X}, \mathbf{w}), \mathbf{y})$.

Missing values Any x_{ij} is a missing value if it is unknown (marked by *null*). An *incomplete sample* (*incomplete feature*) is a sample (feature) with at least one missing value. We use *complete feature* and *complete sample* to refer to features and samples that are free of missing values. We denote the set of all missing values in a feature matrix $\mathbf{X}$ as $M(\mathbf{X})$, the set of incomplete samples as $MS(\mathbf{X})$, and the set of incomplete features as $MF(\mathbf{X})$. In this paper, we focus on the case where all missing values are in the feature matrix and the label vector is complete.

Repair A repair is a complete version of an incomplete feature matrix $\mathbf{X}$ where all missing values in $\mathbf{X}$ are replaced with values from their domains and the complete values of $\mathbf{X}$ remain intact. Given the repair $\mathbf{X}^r$ of the feature matrix $\mathbf{X}$, we denote the repair, i.e. imputation, of the sample $\mathbf{x}_i$ in $\mathbf{X}$ by $\mathbf{x}_i^r$. Since the domains of features often contain numerous or infinite values, an incomplete feature matrix usually has many or infinitely many repairs. We denote this set of all repairs of $\mathbf{X}$ by $\mathbf{X}^R$.

3 Minimal Repair (MR) for SVM

We use the concept of certain model [37] to define minimal repair for SVM. A model $\mathbf{w}^*$ is a certain model for the target function f on the training set $(\mathbf{X}, \mathbf{y})$ if for every repair $\mathbf{X}^r \in \mathbf{X}^R$, we have $\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathcal{W}} L(f(\mathbf{X}^r, \mathbf{w}), \mathbf{y})$ where L is the loss function. Intuitively, a certain model minimizes training loss for all repairs of the incomplete feature matrix. Thus, if a certain model exists, one can learn an accurate model over the training set without any repair to the training data, as training over any repair to the dataset, e.g., using randomly selected values, will deliver the same accurate model. This observation holds regardless of the missingness mechanism—Missing Completely at Random (MCAR), Missing at Random (MAR), or Missing Not at Random (MNAR). Given the restrictive definition of certain models, they do not often exist [37]. Thus, we find the minimal amount of repair of an incomplete training set such that the resulting training set has a certain model.

Definition 1 *A set of incomplete samples $\mathbf{S}_{MR}$ in the training set $(\mathbf{X}, \mathbf{y})$ is a minimal repair for learning SVM with the regularization parameter C if we have: 1) a certain model exists when imputing all missing values $\mathbf{S}_{MR}$ regardless of the imputation result, and 2) there is no other set $\mathbf{S}'$ satisfying condition (1) such that $|\mathbf{S}'| < |\mathbf{S}_{MR}|$ where $|\mathbf{S}_{MR}|$ denotes the cardinality of $\mathbf{S}_{MR}$.*

We denote the minimal repair for SVM with the regularization parameter C on the training set $(\mathbf{X}, \mathbf{y})$ as $\mathbf{S}_{MR}(\mathbf{X}, \mathbf{y}, C)$. We have the following property for the minimal repair of SVM.

Theorem 1 *Given training set $(\mathbf{X}, \mathbf{y})$ and regularization parameter C, $\mathbf{S}_{MR}(\mathbf{X}, \mathbf{y}, C)$ is unique.*

3.1 Finding Minimal Repair

Let $SV(\mathbf{X}^{\mathbf{r}}, \mathbf{y}, C)$ be the set of support vectors for the optimal SVM model with regularization parameter C on a repair $\mathbf{X}^{\mathbf{r}}$ of the training set $(\mathbf{X}, \mathbf{y})$.

Lemma 2 *Given the training set $(\mathbf{X}, \mathbf{y})$ and the regularization parameter C, at least one repair $\mathbf{x}_i^r$ of every sample $\mathbf{x}_i \in S_{MR}(\mathbf{X}, \mathbf{y}, C)$ is a support vector in a repair $\mathbf{X}^{\mathbf{r}}$ of $\mathbf{X}$, i.e., $\mathbf{x}_i^r \in SV(\mathbf{X}^{\mathbf{r}}, \mathbf{y}, C)$.*

Hence, to determine if an incomplete sample belongs to the minimal repair, one could materialize every repair of the feature matrix and check if the incomplete sample is a support vector for any of them. However, this process can be extremely inefficient due to the often large number of repairs. Assume that each missing value x_{ij} is bounded by an interval $[x_{ij}^{min}, x_{ij}^{max}]$ based on its domain. $\mathbf{X}^e$ is an *edge repair* to $\mathbf{X}$ if for every missing value x_{ij}, $x_{ij}^e = x_{ij}^{min}$ or x_{ij}^{max}. $\mathbf{X}^E$ denotes the set of all edge repairs for $\mathbf{X}$. Theorem 3 shows that we can use only the edge repair instead of all repairs to check if an incomplete sample belongs to the minimal repair.

Theorem 3 *Given the training set $(\mathbf{X}, \mathbf{y})$ and the regularization parameter C, an incomplete sample $\mathbf{x}_i$ belongs to minimal repair $S_{MR}(\mathbf{X}, \mathbf{y}, C)$ if and only if there is at least one edge repair $\mathbf{X}^e$ of $\mathbf{X}$ such that $\mathbf{x}_i^e \in SV(\mathbf{X}^e, \mathbf{y}, C)$ where $\mathbf{x}_i^e$ is the repair of $\mathbf{x}_i$.*

Based on Theorem 3, we can find the minimal repair following these steps: 1) Initialize an empty minimal repair set, S_{MR}. 2) Iterate over each incomplete sample $\mathbf{x}_i$. At each iteration, materialize all edge repairs $\mathbf{X}^e \in \mathbf{X}^E$, and check if $\mathbf{x}_i$ is a support vector for any of the edge repairs. If it is, add $\mathbf{x}_i$ to S_{MR}, and 3) Finally, return the minimal repair S_{MR}. Despite this optimization, finding the minimal repair remains computationally intractable.

Theorem 4 *Given a training set $(\mathbf{X}, \mathbf{y})$ with missing values, deciding whether an incomplete sample belongs to the minimal repair for SVM on $(\mathbf{X}, \mathbf{y})$ is NP-hard. Consequently, finding the minimal repair for SVM on $(\mathbf{X}, \mathbf{y})$ is NP-hard.*

3.2 Approximating Minimal Repair

We propose an efficient approximation algorithm (Algorithm 1) to find minimal repair for SVM. Its key idea is to test whether each incomplete sample $\mathbf{x}_i$ belongs to minimal repair by constructing an edge repair $\mathbf{X}^e$ that maximizes the likelihood of $\mathbf{x}_i$ becoming a support vector. This construction begins with a random edge repair and iteratively updates each missing value in the dataset to its minimum or maximum bound. At each step, this choice minimizes $y_i \mathbf{w}^\top \mathbf{x}_i$, encouraging $\mathbf{x}_i$ to satisfy the support vector condition $y_i \mathbf{w}^\top \mathbf{x}_i \leq 1$. If this condition holds after the full pass of the data, $\mathbf{x}_i$ is selected for repair. Crucially, this algorithm **does not return any false positive**. Since the algorithm initializes with a randomly selected edge repair, it does not introduce bias towards any specific imputation in learning models.

Theorem 5 *Every sample returned by Algorithm 1 belongs to $S_{MR}(\mathbf{X}, \mathbf{y}, C)$.*

Algorithm 1 Approximating minimal repair for SVM on training set $(\mathbf{X}, \mathbf{y})$

```
$S_{MR} \leftarrow [\ \ ]$
$\mathbf{X}^e \leftarrow$ a random edge repair to the feature matrix $\mathbf{X}$
for $\mathbf{x}_i \in MS(\mathbf{X})$ do
    for $x_{pq} \in M(\mathbf{X})$ do
        $\mathbf{X}^{e_{min}}, \mathbf{X}^{e_{max}} \leftarrow$ two edge repairs by only replacing $x_{pq}$ in $\mathbf{X}^e$ with its min or max value
        $\mathbf{w}_1, \mathbf{w}_2 \leftarrow SVM(\mathbf{X}^{e_{min}}, \mathbf{y}), SVM(\mathbf{X}^{e_{max}}, \mathbf{y})$ {learning SVM models with edge repairs}
        $\mathbf{X}^e \leftarrow$ if $y_i \mathbf{w}_1^\top \mathbf{x}_i^{e_{min}} \leq y_i \mathbf{w}_2^\top \mathbf{x}_i^{e_{max}}$ then $\mathbf{X}^{e_{min}}$ else $\mathbf{X}^{e_{max}}$
    end for
    $\mathbf{w} \leftarrow SVM(\mathbf{X}^e, \mathbf{y})$
    if $y_i \mathbf{w}^\top \mathbf{x}_i \leq 1$ then $S_{MR} \leftarrow S_{MR}.add(\mathbf{x}_i)$
end for
return $S_{MR}$
```

Since each iteration modifies only one missing value, adjacent models $\mathbf{w}_1$ and $\mathbf{w}_2$ differ by only a single feature entry. This allows us to avoid retraining from scratch by applying incremental or decremental SVM updates [7, 20]. These techniques update the model efficiently—typically an order of magnitude faster—by reusing computations from the previous solution.

Algorithm 1 may miss some samples of minimal repair. Thus, we iteratively apply Algorithm 1 to the remaining incomplete samples in the training set to find more samples in the minimal repair of the training set. The process ends when no new samples are selected for repair. The following theorem shows that the probability of not finding samples of minimal repair decreases using this approach.

Theorem 6 *Given the training set $(\mathbf{X}, \mathbf{y})$, let $p_k(\mathbf{x})$ be the probability that an incomplete sample $\mathbf{x}$ in minimal repair of $(\mathbf{X}, \mathbf{y})$ not returned in iteration of $k > 0$ in iterative application of Algorithm 1, $p_k(\mathbf{x}) > p_{k+1}(\mathbf{x})$.*

Corollary 6.1 *If the probability distribution of each missing value is known, and we let $g(x_{ij})$ denote the probability density function of the ground truth value for the missing value x_{ij} in the incomplete training set $(\mathbf{X}, \mathbf{y})$. If missing values in $\mathbf{X}$ are independent, the probability that an incomplete sample $\mathbf{x}_i$ in minimal repair not returned by Algorithm 1 in the main content is:*

$$p(\mathbf{x}_i) = 1 - \frac{\int \cdots \int_{\min(x_{ij}^{\textit{visited}})}^{\max(x_{ij}^{\textit{visited}})} \prod_{x_{ij} \in M(\mathbf{X})} g(x_{ij})\, dx_{ij}}{\int \cdots \int_{x_{ij} \in M(\mathbf{X})} \prod_{x_{ij} \in M(\mathbf{X})} g(x_{ij})\, dx_{ij}} \tag{1}$$

$x_{ij}^{\textit{visited}} \in \{x_{ij}^{min}, x_{ij}^{max}\}$ shows the values used for x_{ij} in Algorithm 1.

4 Minimal Repair for Linear Regression

The minimal repair for linear regression is the smallest set of features that is necessary to repair.

Definition 2 *Given the training set $(\mathbf{X}, \mathbf{y})$, a set of incomplete features in $\mathbf{X}$, denoted as $\mathbf{S}_{MR}(\mathbf{X}, \mathbf{y})$, is a minimal repair for $(\mathbf{X}, \mathbf{y})$ for linear regression if we have: 1) a certain model exists upon imputing all missing values in the $\mathbf{S}_{MR}(\mathbf{X}, \mathbf{y})$ regardless of the imputation result, and 2) there is no set $\mathbf{S}$ satisfying condition (1) and $|\mathbf{S}| < |\mathbf{S}_{MR}|$.*

In linear regression, the optimal linear regression model $\mathbf{w}^*$ consists of the set of linear coefficients for feature vectors. A feature $\mathbf{z}_i$ is considered relevant if the corresponding linear coefficient in the optimal model w_i^* is not zero, and it is irrelevant if w_i^* equals zero. Intuitively, an incomplete feature needs to be repaired if it is relevant (i.e., it plays a role in the optimal model) and does not need to be repaired if it is irrelevant. However, traditional statistical tools, such as the chi-square test, require complete distributions for each feature to assess correlations, which is challenging in the presence of missing values. The minimal repair for linear regression may not be unique.

Theorem 7 *There is a training set with multiple minimal repairs for linear regression. In addition, if all the features in all the repairs of the training set $(\mathbf{X}, \mathbf{y})$ are linearly independent, the minimal repair for linear regression over $(\mathbf{X}, \mathbf{y})$ is unique.*

The following theorem establishes that finding minimal repair for linear regression is intractable.

Theorem 8 *Given a training set* $(\mathbf{X}, \mathbf{y})$ *with incomplete features, finding the minimal repair for linear regression over* $(\mathbf{X}, \mathbf{y})$ *is NP-hard.*

To find minimal repair efficiently, we first propose an equivalent problem in Theorem 9, based on a variant of the well-known sparse linear regression problem [5].

Lemma 9 *Finding the minimal repair for linear regression on training set* $(\mathbf{X}, \mathbf{y})$ *is equivalent to:*

$$\begin{aligned} &\min_{\mathbf{w} \in \mathcal{W}} T_{MF(\mathbf{X})}(\mathbf{w}) \\ &\textit{subject to} \quad \mathbf{w} = \arg\min ||\mathbf{X}^r \mathbf{w} - \mathbf{y}||_2^2, \forall \mathbf{X}^r \in \mathbf{X}^R \end{aligned} \tag{2}$$

where $T_{MF(\mathbf{X})}(\mathbf{w})$ *is the number of non-zero linear coefficient in* $\mathbf{w}$ *whose corresponding feature is incomplete, i.e.,* $T_{MF(\mathbf{X})}(\mathbf{w}) = |\{\mathbf{z}_i \in MF(\mathbf{X}) | w_i! = 0\}|$

The key distinction between our problem and sparse linear regression lies in their objectives: sparse linear regression seeks to minimize the number of non-zero coefficients across all features, whereas we focus on minimizing the number of non-zero coefficients only among incomplete features. Orthogonal Matching Pursuit (OMP) provides an efficient approximation to solve the sparse linear regression problem [32]. This greedy algorithm begins with an empty solution set and initializes the regression residual to the label vector. In each iteration, the algorithm selects the feature most relevant to the current residual (having the largest dot product), adds it to the solution set, retrains a linear regression model, and updates the residual accordingly. It stops when the regression residue is sufficiently small.

We propose a variant of OMP, as outlined in the appendix, to find minimal repair for linear regression. Our algorithm has two major differences compared to the conventional OMP. First, we include all complete features in the regression at the initialization, ensuring that we minimize the number of non-zero coefficients only among incomplete features. Secondly, we define our stopping condition by the maximum relevance (cosine similarity) between the feature and the label being smaller than or equal to a user-defined threshold, instead of relying on a near-zero regression residue. This approach enables our algorithm to work with general datasets without requiring the assumption of an underdetermined linear system, which is typically necessary in conventional OMP.

The time complexity of the algorithm is $\mathcal{O}(T_{train} \cdot |MF(\mathbf{z})|)$, making it significantly more efficient than the baseline algorithm, which trains models on all repairs individually and has a time complexity of $\mathcal{O}(T_{train} \cdot |\mathbf{X}^R|)$. If we use gradient descent, our algorithm has a time complexity of $\mathcal{O}(n \cdot d^3)$, where n is the number of training samples and d is the number of features. In cases where $n < d^2$, the time complexity is reduced to $\mathcal{O}(n \cdot d^2 + n^2 \cdot d)$ under certain conditions by applying incremental learning techniques based on the Sherman-Morrison formula, as outlined in the appendix. The following theorem characterizes the approximation rate of our algorithm.

Theorem 10 *The first* k *incomplete features added to* S_{MR} *in our algorithm for training set* $(\mathbf{X}, \mathbf{y})$ *belong to a minimal repair of* $(\mathbf{X}, \mathbf{y})$ *with a probability of at least* $1 - 1/n$*, provided that: 1)* $\mu < 1/(2k-1)$*, 2) the missing values in the dataset follow independent zero-mean normal distributions (*$\mathcal{N}(0, \sigma_{ij}^2)$*), and 3) all linear coefficients (*$w_i, \mathbf{z}_i \in MF(\mathbf{X})$*) for incomplete features satisfy:*

$$|w_i| \geq \frac{2 \sum_{x_{ij}=null} \sigma_{ij} \sqrt{n + 2\sqrt{n \log n}}}{1 - (2k-1)\mu} \tag{3}$$

where μ *is the mutual incoherence defined by* $\mu = \max_{i \neq j} |\mathbf{z_i}^T \mathbf{z_j}|$.

5 Almost Minimal Repair

Minimal repair might be too large and take a long time to compute for some datasets and learning tasks. Thus, we relax the definition of minimal repair to reduce its size and computation cost. Instead of enforcing exact optimality, we aim for a set whose imputation can deliver a model that is near-optimal for all possible repairs. We use the concept of approximately certain model (ACM) [37] to formalize this notion. For a user-defined error threshold $e \geq 0$, $\mathbf{w}^{\approx}$ is an ACM for the target function f on the training set $(\mathbf{X}, \mathbf{y})$ if for every repair $\mathbf{X}^r$, $L(\mathbf{w}^{\approx}, \mathbf{X}^r, \mathbf{y}) - \min_{\mathbf{w} \in \mathcal{W}} L(\mathbf{w}, \mathbf{X}^r, \mathbf{y}) \leq e$.

Definition 3 *Given a threshold $e \geq 0$, a set S_{AMR} of incomplete samples in the training set $(\mathbf{X}, \mathbf{y})$ is an almost minimal repair (AMR) for the target function f with loss L if: (1) repairing S_{AMR} yields an ACM for f in $(\mathbf{X}, \mathbf{y})$, and (2) no other set S' satisfies (1) with $|S'| < |S_{AMR}|$.*

If $e = 0$, ACM reduces to a certain model. Hence, we can show that computing AMR is also NP-hard for SVM (details are in the appendix).

5.1 Computing AMR

We first propose an iterative algorithm with two main steps. Step 1 (ST1: ACM Optimizer) takes the input dataset in iteration $k > 0$ of the algorithm, $\mathbf{X}^{(k)}$, and finds the model $\mathbf{w}_k^{\approx}$ that minimizes the worst-case suboptimality gap $g_k = \sup_{\mathbf{X}^{(k)r}} \left[L(\mathbf{w}_k^{\approx}, \mathbf{X}^{(k)r}, \mathbf{y}) - \min_{\mathbf{w} \in \mathcal{W}} L(\mathbf{w}, \mathbf{X}^{(k)r}, \mathbf{y}) \right]$.

Step 2 (ST2: Local Repair Set Identifier) examines whether $g_k > e$, and if so, returns the smallest set of currently incomplete samples whose imputation may help further reduce the suboptimality gap in the next iteration.

Theorem 11 *Given the training set $(\mathbf{X}, \mathbf{y})$, each selection made by ST2 belongs to the AMR set S_{AMR} of $(\mathbf{X}, \mathbf{y})$. Thus, the iterative algorithm terminates with an ACM, and the total imputed set $S_{iter\text{-}ACM} \subseteq S_{AMR}$, where $S_{iter\text{-}ACM}$ is the union of all incomplete samples selected across iterations.*

This guarantees that our algorithm converges to an ACM by imputing only a subset of S_{AMR}. The key distinction is that S_{AMR} is defined to guarantee the ACM condition under all possible repairs—it is sufficient without knowledge of any imputation results. In contrast, the iterative algorithm dynamically learns imputation results along the way. This new information may render some samples in S_{AMR} unnecessary for achieving ACM in the current trajectory. Thus, $S_{\text{iter-ACM}}$ can be smaller than S_{AMR} while still ensuring the ACM condition.

5.2 Efficient Approximation

Both ST1 and ST2 are intractable because they require solving min-sup optimization over exponentially many repairs and identifying minimal subsets of incomplete samples whose repair is necessary when an ACM does not yet exist. Specifically, these are the samples whose imputation would further reduce the minimum value of the worst-case suboptimality gap $g(\mathbf{w}) = \sup_{\mathbf{X}^r} h(\mathbf{w}, \mathbf{X}^r)$ toward the user-defined threshold e. Finding such subsets involves understanding how each missing value affects the supremum over all repairs—a problem known to be computationally hard in general due to the nested structure of min-max optimization [4]. We therefore propose efficient approximations of these steps that make the entire algorithm tractable.

Approximating ST1 (ACM Optimizer): ST1 aims to find the model $\mathbf{w}_k^{\approx} = \arg\min_{\mathbf{w}} \sup_{\mathbf{X}^r \in \mathcal{X}^R_{\text{rem}}} h(\mathbf{w}, \mathbf{X}^r)$, where $h(\mathbf{w}, \mathbf{X}^r) = L(\mathbf{w}, \mathbf{X}^r) - \min_{\mathbf{w}'} L(\mathbf{w}', \mathbf{X}^r)$. When the loss function L is convex, each $h(\mathbf{w}, \mathbf{X}^r)$ is convex in $\mathbf{w}$, and so is the pointwise supremum of such functions. Thus, we approximate this by sampling a finite subset of edge repairs $\{\mathbf{X}_1^e, \ldots, \mathbf{X}_s^e\}$ and solving the convex problem $\min_{\mathbf{w}} \max_i h(\mathbf{w}, \mathbf{X}_i^e)$.

However, directly computing $h(\mathbf{w}, \mathbf{X}^e)$ requires solving an inner optimization for each sampled repair to obtain the minimum loss. To make this tractable, we use the subgradient norm $\|\mathbf{g}(\mathbf{w}, \mathbf{X}^e)\|$ as a proxy for the suboptimality gap.

Theorem 12 *If $L(\mathbf{w})$ is convex and has an M-Lipschitz continuous gradient, then any model $\mathbf{w}^{\approx}$ satisfying $\|\nabla_{\mathbf{w}} L(\mathbf{w}, \mathbf{X}^r)\| \leq \sqrt{2Me}$ for all $\mathbf{X}^r$ is an ACM.*

This result implies that for linear regression, which satisfies the convexity and smoothness conditions, we can directly use the gradient norm to check whether a model is an ACM. For non-differentiable models like linear SVM, the hinge loss is not smooth and the subgradient norm is not convex. Nonetheless, we still use the subgradient norm as a practical stopping proxy to assess whether ACM has been achieved.

Approximating ST2 (Local Repair Set Identifier): ST2 must find a small subset of currently incomplete samples whose repair enables further progress toward satisfying the ACM condition. We

Table 1: Details of datasets with injected missing data

Data Set	Task	Features	Training samples
Malware	Classification	6823	1596
Tuadromd	Classification	242	3571
Credit Default	Classification	23	30000
Gas	Regression	129	2566
Superconductivity	Regression	82	21262
Concrete	Regression	8	1030

Table 2: Details of datasets with original missing data

Data Set	Task	Features	Training samples	Missing Factor
Breast Cancer	Classification	10	559	1.97%
Water-Potability	Classification	9	2620	39.00%
Online-Ed	Classification	36	7026	35.48%
Bankruptcy	Classification	64	8402	54.00%
Air Quality	Regression	12	7344	90.80%
Communities	Regression	1954	1595	93.67%
Cancer Rate	Regression	32	3048	81.00%

Table 3: Accuracy and runtime for datasets with injected missing values for SVM

Data Set	% Missing	Ground Truth Accuracy(%)	Time(s)			Accuracy(%)			Impute % of Samples		
			AC	MR	AMR	AC	MR	AMR	AC	MR	AMR
Malware	0.2	95.61	1.36	49.6±3	26.15	93.13	96.49±3	95.48	6.39	34.17	2.19
	0.4	95.04	0.56	98.0±3	26.31	92.20	92.42±3	96.74	3.35	29.78	2.04
	0.6	95.91	0.170	147.8±3	32.32	88.67	96.37±3	94.73	3.28	24.56	2.09
Tuadromd	0.2	98.67	0.68	8.1	2.89	97.53	98.73	98.99	3.78	16.53	2.10
	0.4	98.77	0.54	18.5	4.11	97.42	98.81	98.99	3.53	14.92	2.03
	0.6	98.77	0.34	30.6	5.7	97.50	98.77	98.2	2.48	13.17	2.01
Credit Default	0.2	81.03	11.86	78	5.53	74.61	81.02	78.6	0.19	100	0.08
	0.4	81.03	14.19	186.0	5.92	71.91	81.00	78.7	0.23	99.51	0.05
	0.6	81.03	14.2	309.9	8.0	66.40	81.02	78.43	0.19	99.82	0.03

approximate this by identifying edge repairs $\mathbf{X}^e$ from the sampled set where $\|\mathbf{g}(\mathbf{w}_k^{\approx}, \mathbf{X}^e)\| > \epsilon'$, indicating that ACM is violated under these repairs.

We then inspect each such "problematic" edge repair. For each incomplete sample x_j that currently violates the margin condition (i.e., $y_j(\mathbf{w}_k^{\approx})^T\mathbf{x}_j^e < 1$), we check if there exists a feasible repair where the margin would exceed 1. If so, we assign a score to x_j estimating its potential to reduce the subgradient norm. One option is the maximum hinge loss reduction:

$$\Delta L_{\max} = C \cdot \left[(1 - \text{margin}_j) - \max(0, 1 - \text{margin}_{j,\max})\right], \tag{4}$$

where $\text{margin}_{j,\max}$ is estimated using interval arithmetic over the missing feature bounds. Alternatively, we compute a gradient alignment score based on the inner product between the current subgradient vector and $Cy_j\mathbf{x}_j^e$, estimating the contribution to gradient magnitude.

These scores are aggregated across all high-gradient edge repairs. We then select the top-h highest-ranked incomplete samples for imputation in the next iteration. This procedure effectively approximates the function of ST2, enabling tractable, targeted refinement of the model toward satisfying the ACM condition.

6 Experimental Evaluation

We evaluated our methods on six real-world datasets with injected missingness and seven with naturally occurring missing values, spanning diverse domains and varying in missingness ratios, feature dimensionalities, and sample sizes (Table 2). We evaluate our methods on six real-world datasets with synthetically injected missingness and seven with naturally occurring missing values. These datasets span diverse domains and vary in missingness ratios (proportion of incomplete samples), feature dimensionalities, and sample sizes; Table 2 summarizes them. We first compare our

Table 4: Accuracy and runtime of model-based imputation methods for SVM

Data Set	Method	Time(s)			Accuracy(%)			Impute % of Samples		
		KNN	MICE	TCSDI	KNN	MICE	TCSDI	KNN	MICE	TCSDI
Breast Cancer	MR	0.055±0.002	0.064±0.001	51±3.5	96.30±0.2	96.4±0.2	97.00±1.03	18.2	18.2	18.2
	AMR	0.1237	0.1357	85	96.40	96.40	97.11	54.54	54.54	45.45
	AC	0.065	0.065	84	95.85	96.30	97.87	87.27	87.27	87.27
	Baseline	0.0039	0.046	102	95.78	96.30	97.00	100	100	100
Water-Potability	MR	0.259±0.048	0.135±0.004	473.7±2.3	60.2±1.00	60.3±0.3	62.80±1.38	30	30	30
	AMR	1.531	1.756	27.13	55.69	56.13	60.13	0.18	0.18	0.18
	AC	0.33	0.033	85.32	54.96	56.90	57.00	1.94	1.94	1.94
	Baseline	0.0053	0.0115	1459	96.00	96.30	97.00	100	100	100
Online-Ed	MR	1.606±0.322	0.748±0.318	1087.2±4.1	64.5±0.8	64.5±0.3	65.22±0.01	29.91	29.91	29.91
	AMR	19.58	21.31	561.5	62.79	62.70	63.87	14.79	14.79	14.79
	AC	1.83	1.88	93.76	63.71	60.77	63.60	0.81	0.81	0.81
	Baseline	0.989	1.270	3624	65.23	65.17	65.23	100	100	100
Bankruptcy	MR	2.798±0.09	0.76±0.084	2286.7±5.4	97.22±0.033	97.8±0.01	97.79±0.04	29.9	29.9	29.9
	AMR	21.37	25.13	49.89	96.40	96.40	97.11	0.52	0.53	0.53
	AC	2.24	2.25	101	54.96	56.90	56.95	0.6	0.6	0.6
	Baseline	4.843	22.15	7620	96.00	96.30	97.00	100	100	100
Malware 0.6	MR	18.26±0.642	-	64959.2±4286.7	96.16±1.06	-	97.87±0.45	16.65	-	16.65
	AMR	33.17	-	1442	95.12	-	96.01	18.87	-	18.87
	AC	0.84	-	11085	85.99	-	88.82	3.28	-	3.28
	Baseline	41.209	-	390806	96.16	-	97.87	100	-	100

Table 5: Accuracy and runtime for datasets with injected missing values for Linear Regression

Data Set	% Missing	Ground Truth	Time(s)			MSE			Impute % of Samples or Features		
		MSE	AC	MR	AMR	AC	MR	AMR	AC	MR	AMR
Superconductivity	0.2	0.0088	2.200	2.305	4.88	0.01	0.00888	0.0092	0.24	70.00	0.07
	0.4	0.0088	2.228	2.534	6.22	0.0099	0.00885	0.0092	0.22	75.00	0.07
	0.6	0.0088	1.465	2.476	7.22	0.0103	0.00885	0.0093	0.25	75.00	0.07
Gas	0.2	0.1053	0.0734	0.31	2.98	0.279	0.105	0.127	2.01	65.00	0.58
	0.4	0.1053	0.051	0.3391	3.16	0.296	0.1054	0.167	2.01	65.00	0.58
	0.6	0.1053	0.0332	0.551	3.45	0.355	0.112	0.163	1.78	25.00	0.58
Concrete	0.2	0.0149	0.0126	0.0227	0.0203	0.0267	0.01495	0.0159	6.89	50.00	1.46
	0.4	0.0149	0.0149	0.0202	0.0328	0.0293	0.01495	0.0159	5.63	50.00	1.46
	0.6	0.0149	0.0065	0.0199	0.0449	0.0356	0.01495	0.0162	5.28	50.00	1.46

Table 6: Accuracy and runtime of model-based imputation methods for Linear Regression

Data Set	Method	Time(s)			MSE			Impute % of Samples		
		KNN	MICE	TCSDI	KNN	MICE	TCSDI	KNN	MICE	TCSDI
Cancer Rate	MR	0.153	0.574	5852	0.0045	0.0045	0.0045	0.33	0.33	0.33
	AMR	0.213	0.234	91.23	0.0047	0.0045	0.0045	1.46	1.44	1.46
	AC	0.166	0.133	110	0.0050	0.0051	0.0049	0.70	0.70	0.70
	Baseline	0.584	0.664	6104	0.0045	0.0058	0.0049	100	100	100
Air Quality	MR	1.06	2.46	14976	5.671	5.74	5.82	50	50	50
	AMR	0.815	0.956	901	6.015	5.99	5.98	4.78	4.35	4.90
	AC	0.199	0.0612	95	6.66	6.71	7.138	1.69	1.69	1.69
	Baseline	1.763	2.46	18372	5.672	5.923	5.825	100	100	100
Communities	MR	26.74	28863	-	0.023	0.026	-	75	75	-
	AMR	43.18	679.5	-	0.020	0.024	-	1.98	1.98	-
	AC	-	-	-	-	-	-	-	-	-
	Baseline	26.72	33475	-	0.019	0.024	-	100	100	-
Gas 0.6	MR	0.566	35.05	5098	0.1983	0.1626	0.1166	65.00	65.00	65.00
	AMR	1.58	19.23	33.74	0.202	0.178	0.180	0.58	0.58	0.58
	AC	0.183	16.9	109	0.215	0.212	0.221	1.78	1.78	1.78
	Baseline	0.447	38.67	5227	0.185	0.192	0.2001	100	100	100

methods to *Active Clean (AC)* [17], which integrates data repair with stochastic gradient descent: in each iteration it samples a batch, returns it to the user for repair, and then updates model parameters with the repaired samples. Although *AC* reduces repair cost by prioritizing influential samples, it is unclear whether the resulting repaired data yields an accurate model, since not all samples are ever selected for gradient updates. Accordingly, we compare the accuracy and time overhead of our methods and *AC* in settings where users manually repair data items, as explained in Section 1. We therefore use datasets with injected missingness whose ground truth is available. Details about the hardware used in our experiments are available in appendix.

Table 3 reports SVM classification results for minimal repair (*MR*), almost minimal repair (*AMR*), and *AC*. The results show that *MR* and *AMR* consistently outperform *AC* in accuracy across all datasets and missingness levels. Notably, *AMR* achieves higher accuracy than *AC* while repairing substantially fewer samples. For example, on the Credit Default dataset at 40% missingness, *AMR* repairs only 0.05% of samples versus *AC*'s 0.23%, yet improves accuracy considerably (78.7% for

AMR vs. 71.91% for *AC*). *MR* attains the highest accuracy overall, though it selects more samples for repair.

Table 5 compares regression outcomes for *MR*, *AMR*, and *AC*. Unlike *AC* and *AMR*, which repair entire samples, *MR* imputes individual missing features (see Section 4). Consistent with the classification findings, *MR* and *AMR* again outperform *AC* in terms of mean squared error (MSE) across all datasets and missingness ratios. On the Gas dataset at 60% missingness, *AMR* performs fewer repairs than *AC* yet achieves markedly lower MSE (0.163 vs. 0.355). *MR* achieves the lowest MSE overall, reflecting its more comprehensive repair strategy aimed at closely approximating the optimal model.

We next evaluate the time and effort saved when using model-based imputations. Because imputation cost grows with the number of missing items, (almost) minimal repair can cut both inference time and user effort for inspecting or verifying imputed values. We use three imputation models: *KNN* (predicts missing values from observed samples via a KNN classifier) [23], *MICE* (multivariate regression–based imputation) [6], and *TCSDI* (a diffusion model for imputation) [38], representing diverse approaches. We compare full imputation, imputing (almost) minimal repair, and imputing the samples selected by *AC*, measuring accuracy, running time, and the number of imputed items (a proxy for user effort). We exclude imputation-model training time, which depends only on the observed subset and is identical across methods. Experiments use both datasets with natural missingness and those with injected missingness; due to space, full injected-missingness results appear in appendix.

Comparing the three imputation models in Tables 4 and 6, *TCSDI* consistently achieves higher accuracy, but with longer inference times than *KNN* and *MICE*. This underscores the practical value of *MR* and *AMR*, which substantially reduce inference overhead by limiting imputations, especially when paired with *TCSDI*. For instance, on the Malware dataset with 60% missingness, our methods reduce imputation time by nearly three days relative to fully imputing the dataset, while maintaining comparable accuracy. For the Malware dataset, we omit *MICE* results because it exceeded available memory—an expected limitation given that *MICE* scales poorly with the number of features. On the Bankruptcy dataset (approximately 5% minority vs. 95% majority), *AC* required multiple executions to obtain stable results under severe class imbalance, whereas *MR* and *AMR* remained robust.

We also assess linear regression with model-based imputations. Here, *MR* and *AMR* consistently deliver faster inference than full imputation while maintaining comparable accuracy, despite substantially fewer imputations. In contrast, *AC* encountered computational issues on high-missingness datasets—particularly Communities and Crime—where minimal cleaning occasionally left zero training samples, causing failures in partial fitting. *MR* and *AMR* avoid such failures, demonstrating robustness at substantial missingness ratios.

Finally, while some theoretical results assume conditions such as zero-mean Gaussian noise or M-Lipschitz continuity of loss functions, these assumptions are not required in practice. The datasets in our empirical evaluation do not satisfy these conditions, and SVM models do not satisfy M-Lipschitz continuity; nonetheless, *MR* and *AMR* consistently deliver accurate results.

7 Related Work

Researchers have proposed *stochastic optimization* to find a model by optimizing the expected loss function over the probability distributions of missing data items in training samples [11]. Similarly, *robust optimization* aims to minimize the loss function of a model for the imputation that brings the highest training loss given certain distributions of missing values [1]. However, the distributions of missing data items are not often available. Thus, users may spend significant time and effort discovering or training these distributions, which may require the user to find the causes of missingness in the data and dependencies between the features. Additionally, for a given type of model, users must solve various and possibly challenging optimization problems for many possible (combinations of) distributions of missing values. More importantly, these methods reflect the uncertainty in the training data caused by missing values in the trained model instead of repairing the data to reduce its uncertainty. Hence, they deliver inaccurate models on the dataset with many missing values.

There are methods to detect cases where the imputation of missing data is not necessary to learn accurate models [26, 16, 37]. Although these approaches are useful for some datasets and learning tasks, they ignore a majority of learning tasks in which imputing incomplete samples impacts the quality of the learned model. More discussion about related work is available in appendix.

References

[1] Alireza Aghasi, MohammadJavad Feizollahi, and Saeed Ghadimi. Rigid: Robust linear regression with missing data. *arXiv preprint arXiv:2205.13635*, 2022.

[2] Muhammad Aurangzeb Ahmad, Carly Eckert, and Ankur Teredesai. The challenge of imputation in explainable artificial intelligence models. In *IJCAI Workshop on Artificial Intelligence Safety*, 2019. URL `https://arxiv.org/abs/1907.12669`.

[3] Rebecca R Andridge and Roderick JA Little. A review of hot deck imputation for survey non-response. *International statistical review*, 78(1):40–64, 2010.

[4] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. Robust solutions of optimization problems affected by uncertain data. *Mathematical Programming*, 112(1):1–20, 2008.

[5] Alfred M Bruckstein, David L Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1):34–81, 2009.

[6] Stef Buuren and Catharina Groothuis-Oudshoorn. Mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45, 12 2011. doi: 10.18637/jss.v045.i03.

[7] Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. *Advances in neural information processing systems*, 13, 2000.

[8] Xu Chu, Ihab F. Ilyas, Sanjay Krishnan, and Jiannan Wang. Data cleaning: Overview and emerging challenges. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, page 2201–2206, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450335317. doi: 10.1145/2882903.2912574. URL `https://doi.org/10.1145/2882903.2912574`.

[9] Chapman JR Dettori JR, Norvell DC. The sin of missing data: Is all forgiven by way of imputation? *Global Spine J.*, 8(8):892–894, 2018. doi: doi:10.1177/2192568218811922.

[10] Alireza Farhangfar, Lukasz Kurgan, and Jennifer Dy. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41(12):3692–3705, 2008.

[11] Ravi Ganti and Rebecca M Willett. Sparse linear regression with missing data. *arXiv preprint arXiv:1503.08348*, 2015.

[12] JW Graham. Missing data analysis: making it work in the real world. *Annu Rev Psychol.*, 60: 549–576, 2009.

[13] Nitin Gupta, Shashank Mujumdar, Hima Patel, Satoshi Masuda, Naveen Panwar, Sambaran Bandyopadhyay, Sameep Mehta, Shanmukha Guttula, Shazia Afzal, Ruhi Sharma Mittal, and Vitobha Munigala. Data quality for machine learning tasks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 4040–4041, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325. doi: 10.1145/3447548.3470817. URL `https://doi.org/10.1145/3447548.3470817`.

[14] Janus Christian Jakobsen, Christian Gluud, Jørn Wetterslev, and Per Winkel. When and how should multiple imputation be used for handling missing data in randomised clinical trials –a practical guide with flowcharts. *BMC Medical Research Methodology*, 17(1):162, 2017. doi: 10.1186/s12874-017-0442-1. URL `https://doi.org/10.1186/s12874-017-0442-1`.

[15] K. P. Junaid, Tanvi Kiran, Madhu Gupta, Kamal Kishore, and Sujata Siwatch. How much missing data is too much to impute for longitudinal health indicators? a preliminary guideline for the choice of the extent of missing proportion to impute with multiple imputation by chained equations. *Population Health Metrics*, 23(1):2, 2025.

[16] Bojan Karlaš, Peng Li, Renzhi Wu, Nezihe Merve Gürel, Xu Chu, Wentao Wu, and Ce Zhang. Nearest neighbor classifiers over incomplete information: From certain answers to certain predictions. *arXiv preprint arXiv:2005.05117*, 2020.

[17] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J Franklin, and Ken Goldberg. Activeclean: Interactive data cleaning for statistical modeling. *Proceedings of the VLDB Endowment*, 9(12):948–959, 2016.

[18] Arun Kumar, Matthias Boehm, and Jun Yang. Data management in machine learning: Challenges, techniques, and systems. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 1717–1722, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450341974. doi: 10.1145/3035918.3054775. URL `https://doi.org/10.1145/3035918.3054775`.

[19] Trent Kyono, Yao Zhang, Alexis Bellot, and Mihaela van der Schaar. Miracle: causally-aware imputation via learning missing data mechanisms. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.

[20] Pavel Laskov, Christian Gehl, Stefan Krüger, Klaus-Robert Müller, Kristin P Bennett, and Emilio Parrado-Hernández. Incremental support vector learning: Analysis, implementation and applications. *Journal of machine learning research*, 7(9), 2006.

[21] 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. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/5fe8fdc79ce292c39c5f209d734b7206-Paper.pdf`.

[22] R.J.A. Little and D.B. Rubin. *Statistical analysis with missing data*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 2002. ISBN 9780471183860. URL `http://books.google.com/books?id=aYPwAAAAMAAJ`.

[23] Pierre-Alexandre Mattei and Jes Frellsen. MIWAE: Deep generative modelling and imputation of incomplete data sets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4413–4423. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/mattei19a.html`.

[24] Kristiaan Pelckmans, Jos De Brabanter, Johan AK Suykens, and Bart De Moor. Handling missing values in support vector machine classifiers. *Neural Networks*, 18(5-6):684–692, 2005.

[25] Massimo Perini and Milos Nikolic. In-database data imputation. *Proceedings of the ACM on Management of Data*, 2(1):1–27, 2024.

[26] Jose Picado, John Davis, Arash Termehchy, and Ga Young Lee. Learning over dirty data without cleaning. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 1301–1316, 2020.

[27] Shafaq Siddiqi, Roman Kern, and Matthias Boehm. Saga: A scalable framework for optimizing data cleaning pipelines for machine learning applications. *Proc. ACM Manag. Data*, 1(3), November 2023. doi: 10.1145/3617338. URL `https://doi.org/10.1145/3617338`.

[28] Marek Śmieja, Łukasz Struski, Jacek Tabor, Bartosz Zieliński, and Przemysław Spurek. Processing of missing data by neural networks. *Advances in neural information processing systems*, 31, 2018.

[29] Daniel J. Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 10 2011. ISSN 1367-4803. doi: 10.1093/bioinformatics/btr597. URL `https://doi.org/10.1093/bioinformatics/btr597`.

[30] Lena Stempfle, Arthur James, Julie Josse, Tobias Gauss, and Fredrik D. Johansson. Handling missing values in clinical machine learning: Insights from an expert study, 2025. URL `https://arxiv.org/abs/2411.09591`.

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

[32] Jian Wang, Seokbeop Kwon, and Byonghyo Shim. Generalized orthogonal matching pursuit. *IEEE Transactions on signal processing*, 60(12):6202–6216, 2012.

[33] Steven Euijong Whang, Yuji Roh, Hwanjun Song, and Jae-Gil Lee. Data collection and quality challenges in deep learning: A data-centric ai perspective. *The VLDB Journal*, 32(4):791–813, 2023.

[34] David Williams, Xuejun Liao, Ya Xue, and Lawrence Carin. Incomplete-data classification using logistic regression. In *Proceedings of the 22nd International Conference on Machine learning*, pages 972–979, 2005.

[35] Mohamed Yakout, Ahmed K. Elmagarmid, Jennifer Neville, Mourad Ouzzani, and Ihab F. Ilyas. Guided data repair. *Proc. VLDB Endow.*, 4(5):279–289, February 2011. ISSN 2150-8097. doi: 10.14778/1952376.1952378. URL `https://doi.org/10.14778/1952376.1952378`.

[36] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. GAIN: Missing data imputation using generative adversarial nets. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5689–5698. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/yoon18a.html`.

[37] Cheng Zhen, Nischal Aryal, Arash Termehchy, and Amandeep Singh Chabada. Certain and approximately certain models for statistical learning. *Proc. ACM Manag. Data*, 2(3), May 2024. doi: 10.1145/3654929. URL `https://doi.org/10.1145/3654929`.

[38] Shuhan Zheng and Nontawat Charoenphakdee. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128*, 2022.

Appendix: Minimal Repairs for Learning Over Incomplete Data

Limitations

While our work demonstrates both theoretical and practical advantages in learning over incomplete data, we acknowledge two limitations:

Model Class and Convexity Assumptions. Our proposed minimal repair (MR) algorithms are developed for support vector machines (SVM) and linear regression, while the almost minimal repair (AMR) framework is applicable to a broader class of statistical machine learning models. However, for AMR, we currently provide provable error bounds and efficient approximations only for models with convex loss functions. This stems from our reliance on Step 1 (ST1) in Section 5.1, where we solve a convex optimization problem to find an approximately optimal model $w_{k}^{\approx}$. Extending AMR to models with non-convex loss functions remains an open challenge due to the difficulty of verifying approximate optimality in such settings. Importantly, this limitation reflects the well-known hardness of non-convex optimization itself—since one cannot generally find globally optimal models for non-convex losses, it is also difficult to guarantee that a repaired model is close to a global optimum.

Trade-off Between Computation and Imputation Time. As seen in our experiments, the time required to compute MR or AMR can exceed or roughly match the time needed to fully impute the dataset when simple imputation methods (e.g., mean or KNN) are used. This suggests that MR and AMR may not be the preferred choice in scenarios where users already opt for inexpensive imputation strategies. However, for more complex, resource-intensive and often accurate imputation methods—such as diffusion-based models [23, 14]—we observe substantial time savings by using MR or AMR to reduce the number of imputations. In practice, users may choose to apply MR or AMR when planning to use high-cost imputation models, and directly pursue full imputation when using simpler methods.

Broader Impacts

Our work has potential positive and negative societal impacts, which we outline below.

Positive societal impacts. Our methods can substantially reduce the time and effort needed for data preparation, a phase that often consumes up to 80% of a data scientist's time [13]. By identifying only the essential missing values to repair, our approach streamlines the ML pipeline, lowers costs, and makes ML more accessible for everyone—especially in resource-constrained settings or domains where full imputation is infeasible.

Negative societal impacts. In high-stakes domains (e.g., healthcare, criminal justice), setting a suboptimal error threshold in AMR (either intentionally or unintentionally) may lead to missed repairs of critical data, resulting in biased or unsafe models. Additionally, the selective repair approach may cause developers to overlook the importance of understanding missingness mechanisms or domain context. These risks can be mitigated by involving domain experts and validating models before deployment.

Preprint. Under review.

Related Work

Researchers have proposed *stochastic optimization* to find a model by optimizing the expected loss function over the probability distributions of missing data items in training samples [9]. Similarly, *robust optimization* aims to minimize the loss function of a model for the imputation that brings the highest training loss given certain distributions of missing values [1]. However, the distributions of missing data items are not often available. Thus, users may spend significant time and effort discovering or training these distributions, which may require the user to find the causes of missingness in the data and dependencies between the features. Additionally, for a given type of model, users must solve various and possibly challenging optimization problems for many possible (combinations of) distributions of missing values. More importantly, these methods reflect the uncertainty in the training data caused by missing values in the trained model instead of repairing the data to reduce its uncertainty. Hence, they deliver inaccurate models on the dataset with many missing values.

There are methods to detect cases where the imputation of missing data is not necessary to learn accurate models [15, 11, 22]. Although these approaches are useful for some datasets and learning tasks, they ignore a majority of learning tasks in which imputing incomplete samples impacts the quality of the learned model.

Researchers have proposed methods to reduce the cost of repair [12, 11]. ActiveClean learns models using stochastic gradient descent and greedily chooses samples for repair that may reduce gradient the most [12]. Unlike our methods, it does not provide any guarantees of minimal repair. Due to the inherent properties of stochastic gradient descent, it is challenging to provide such a guarantee. CPClean follows a similar greedy approach but it is limited to learning k nearest neighbor models over missing data and does not support the types of model our approach addresses [11]. It also does not provide any guarantees of minimality for its imputations.

Hardware

We conducted experiments on two hardware platforms. Most experiments ran on an x86_64 machine with 30 Intel(R) Xeon(R) E5-2670 v3 CPU cores (2.30GHz), hosted in a VMware virtualized environment with two NUMA nodes and 30MB L3 cache. However, this system lacked sufficient power for diffusion-based imputation models. For those experiments (TCSDI), we used an Nvidia DGX-2 system with one Nvidia Tesla V100 GPU (32GB VRAM) and 20 CPU cores from 2.70GHz Intel Xeon Platinum 8168 processors with 33MB L3 cache.

Datasets

We evaluate our methods on two types of datasets: those with synthetic missingness and those with real-world missingness. For each dataset, we simulate three levels of missingness: 0.2, 0.4, and 0.6, corresponding to 20%, 40%, and 60% incomplete samples, respectively. These datasets are further divided based on the downstream task: linear regression (LR) and support vector machine classification (SVM).

All datasets are obtained from publicly available repositories. For synthetic missingness, we start with complete datasets and introduce missing values in a controlled manner. For real missingness, we use datasets that naturally contain incomplete entries. This separation allows us to analyze the behavior of our repair methods under both idealized and realistic data corruption scenarios.

Experimental Results

Here we present the complete experimental results, including those omitted from the main content due to space constraints.

Tables 3 and 5 in the Main Content

Tables 3 and 5 in the main content present the results of minimal repair (MR) and almost minimal repair (AMR) from the first iteration for SVM and linear regression, respectively, due to space

Table A: Accuracy and runtime for datasets with injected missing values for SVM

Data Set	% Missing	Ground Truth Accuracy(%)	Time(s)			Accuracy(%)			Impute % of Samples		
			AC	MR	AMR	AC	MR	AMR	AC	MR	AMR
Malware[16]	0.2	95.61	1.36	Figure1	Figure4	93.13	Figure1	Figure4	6.39	Figure1	Figure4
	0.4	95.04	0.56	Figure1	Figure4	92.20	Figure1	Figure4	3.35	Figure1	Figure4
	0.6	95.91	0.170	Figure1	Figure4	88.67	Figure1	Figure4	3.28	Figure1	Figure4
Tuadromd[5]	0.2	98.67	0.68	Figure2	Figure5	97.53	Figure2	Figure5	3.78	Figure2	Figure5
	0.4	98.77	0.54	Figure2	Figure5	97.42	Figure2	Figure5	3.53	Figure2	Figure5
	0.6	98.77	0.34	Figure2	Figure5	97.50	Figure2	Figure5	2.48	Figure2	Figure5
Credit Default[21]	0.2	81.03	11.86	Figure3	Figure6	74.61	Figure3	Figure6	0.19	Figure3	Figure6
	0.4	81.03	14.19	Figure3	Figure6	71.91	Figure3	Figure6	0.23	Figure3	Figure6
	0.6	81.03	14.2	Figure3	Figure6	66.40	Figure3	Figure6	0.19	Figure3	Figure6

Table B: Accuracy and runtime for datasets with injected missing values for Linear Regression

Data Set	% Missing	Ground Truth MSE	Time(s)			MSE			Impute % of Samples or Features		
			AC	MR	AMR	AC	MR	AMR	AC	MR	AMR
Superconductivity[10]	0.2	0.0088	2.200	2.305	Figure7	0.01	0.00888	Figure7	0.24	70.00	Figure7
	0.4	0.0088	2.228	2.534	Figure7	0.0099	0.00885	Figure7	0.22	75.00	Figure7
	0.6	0.0088	1.465	2.476	Figure7	0.0103	0.00885	Figure7	0.25	75.00	Figure7
Gas[8]	0.2	0.1053	0.0734	0.31	Figure8	0.279	0.105	Figure8	2.01	65.00	Figure8
	0.4	0.1053	0.051	0.3391	Figure8	0.296	0.1054	Figure8	2.01	65.00	Figure8
	0.6	0.1053	0.0332	0.551	Figure8	0.355	0.112	Figure8	1.78	25.00	Figure8
Concrete[20]	0.2	0.0149	0.0126	0.0227	Figure9	0.0267	0.01495	Figure9	6.89	50.00	Figure9
	0.4	0.0149	0.0149	0.0202	Figure9	0.0293	0.01495	Figure9	5.63	50.00	Figure9
	0.6	0.0149	0.0065	0.0199	Figure9	0.0356	0.01495	Figure9	5.28	50.00	Figure9

constraints. Tables A and B provide the full results across multiple iterations for SVM and linear regression.As the number of iterations increases, both the runtime and the number of imputed data items increase, reflecting the cumulative nature of the repair process. However, we observe that in the majority of cases, iteration 1 already achieves accuracy that is close to the ground-truth model trained on fully repaired data.

For instance, on the Credit Default dataset with Minimal Repair, the model converges as early as iteration 1 under 20% missingness. For 40% and 60%, convergence is reached by the second iteration. This explains why the graph only shows a single data point for the 20% case—accuracy stabilizes early and remains relatively unchanged in subsequent iterations. In the case of Almost Minimal Repair, the model converges at the second iteration for 20% and 40% missingness, while under 60% missingness, convergence has not yet been reached, reflecting a more gradual imputation process under higher uncertainty.On the Concrete dataset, AMR achieves convergence at the first iteration under 20% missingness, similar to MR on Credit Default. This early convergence indicates that the repaired model already closely approximates the fully imputed ground-truth model, and additional iterations yield negligible improvement.

This finding suggests that while multi-iteration repair can be used to theoretically guarantee convergence to certain or approximately certain models (as discussed in Sections 3 and 5 of the main content), in practice, a single iteration of MR or AMR is often sufficient to obtain a high-performing downstream model. Thus, users may opt for early stopping after one iteration to save time and effort without sacrificing model accuracy in most settings.

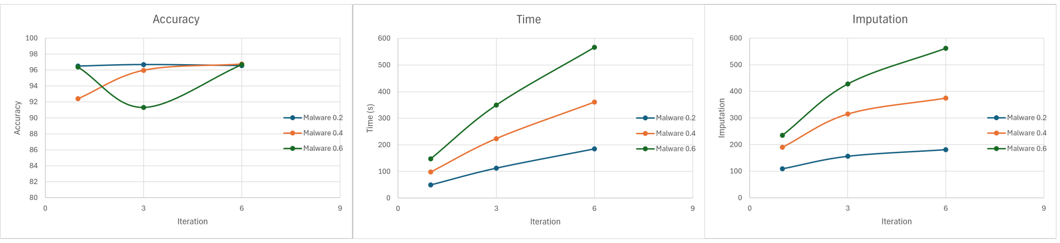

Figure 1: Iterative Minimal Repair results on the Malware dataset. From left to right: classification accuracy, total time (imputation + search), and number of imputed values over iterations. Each line represents a different missingness level (0.2, 0.4, 0.6)

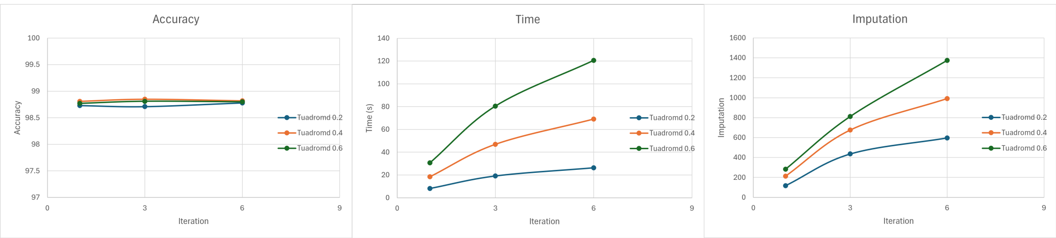

Figure 2: Iterative Minimal Repair results on the Tuadromd dataset. From left to right: classification accuracy, total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

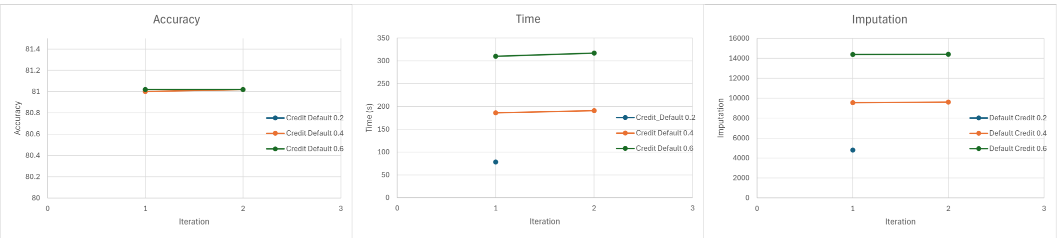

Figure 3: Iterative Minimal Repair results on the Credit Default dataset. From left to right: classification accuracy, total runtime (imputation + search), and number of imputed values across iterations. Each line represents a different missingness level (0.2, 0.4, 0.6).

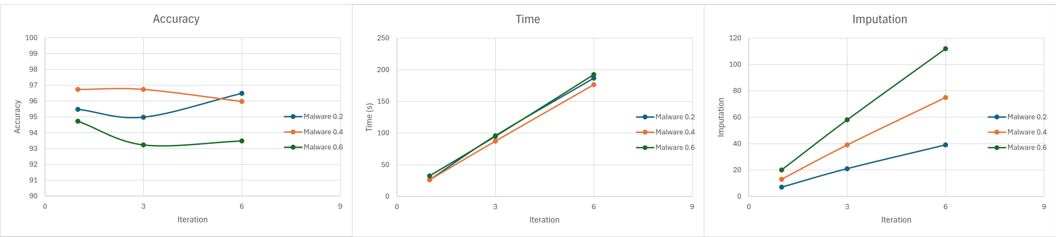

Figure 4: Iterative Almost Minimal Repair results on the Malware dataset. From left to right: classification accuracy, total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

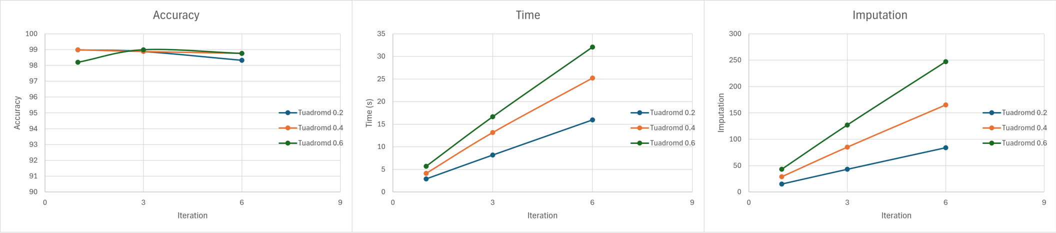

Figure 5: Iterative Almost Minimal Repair results on the Tuadromd dataset. From left to right: classification accuracy, total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

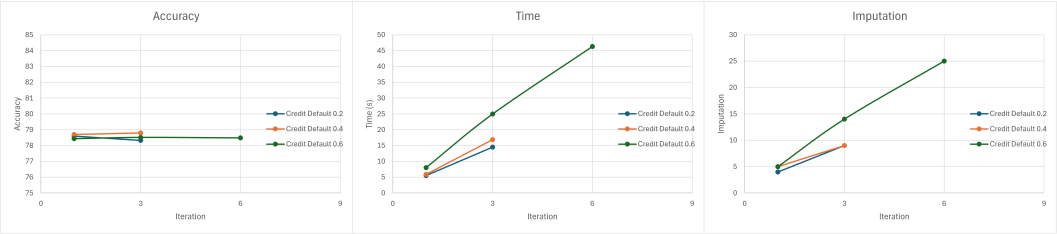

Figure 6: Iterative Almost Minimal Repair results on the Credit Default dataset. From left to right: classification accuracy, total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

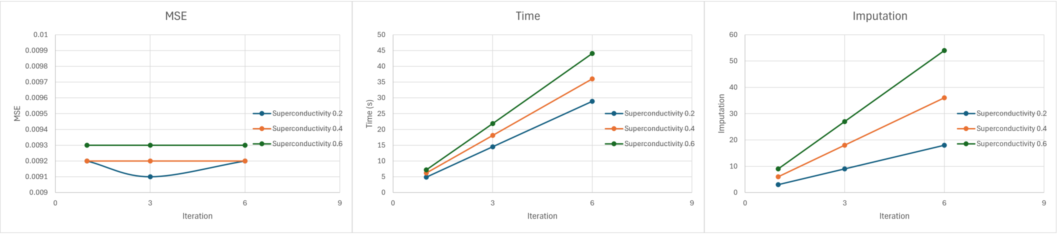

Figure 7: Iterative Almost Minimal Repair results on the Superconductivity dataset. From left to right: mean squared error (MSE), total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

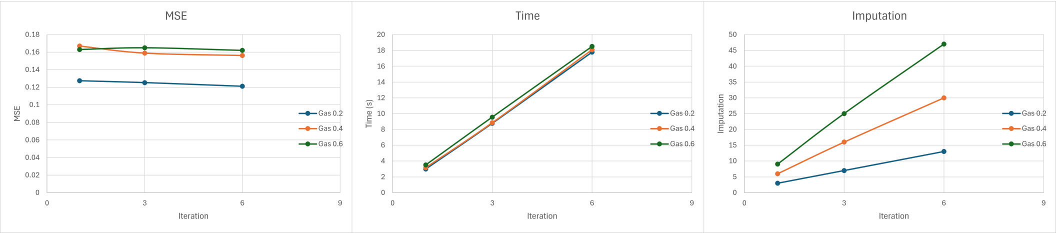

Figure 8: Iterative Almost Minimal Repair results on the Gas dataset. From left to right: mean squared error (MSE), total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

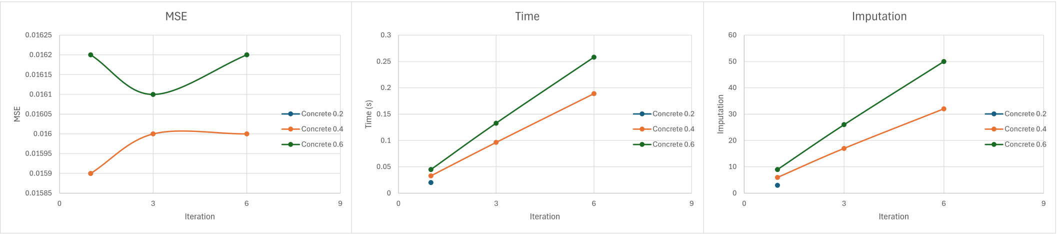

Figure 9: Iterative Almost Minimal Repair results on the Concrete dataset. From left to right: mean squared error (MSE), total runtime (imputation + search), and number of imputed values across iterations. Each curve corresponds to a different missingness level (0.2, 0.4, 0.6).

Table C: Accuracy and runtime of model-based imputation methods for SVM

Data Set	Method	Time(s)			Accuracy(%)			Impute % of Samples		
		KNN	MICE	TCSDI	KNN	MICE	TCSDI	KNN	MICE	TCSDI
Breast Cancer	MR	0.055±0.002	0.064±0.001	51±3.5	96.30±0.2	96.4±0.2	97.00±1.03	18.2	18.2	18.2
	AMR	0.1237	0.1357	85	96.40	96.40	97.11	54.54	54.54	45.45
	AC	0.065	0.065	84	95.85	96.30	97.87	87.27	87.27	87.27
	Baseline	0.0039	0.046	102	95.78	96.30	97.00	100	100	100
Water-Potability	MR	0.259±0.048	0.135±0.004	473.7±2.3	60.2±1.00	60.3±0.3	62.80±1.38	30	30	30
	AMR	1.531	1.756	27.13	55.69	56.13	60.13	0.18	0.18	0.18
	AC	0.33	0.033	85.32	54.96	56.90	57.00	1.94	1.94	1.94
	Baseline	0.0053	0.0115	1459	96.00	96.30	97.00	100	100	100
Online-Ed	MR	1.606±0.322	0.748±0.318	1087.2±4.1	64.5±0.8	64.5±0.3	65.22±0.01	29.91	29.91	29.91
	AMR	19.58	21.31	561.5	62.79	62.70	63.87	14.79	14.79	14.79
	AC	1.83	1.88	93.76	63.71	60.77	63.60	0.81	0.81	0.81
	Baseline	0.989	1.270	3624	65.23	65.17	65.23	100	100	100
Bankruptcy	MR	2.798±0.09	0.76±0.084	2286.7±5.4	97.22±0.033	97.8±0.01	97.79±0.04	29.9	29.9	29.9
	AMR	21.37	25.13	49.89	96.40	96.40	97.11	0.52	0.53	0.53
	AC	2.24	2.25	101	54.96	56.90	56.95	0.6	0.6	0.6
	Baseline	4.843	22.15	7620	96.00	96.30	97.00	100	100	100
Malware 0.2	MR	8.98±0.315	-	24390.8±1604.1	95.6±0.4	-	96.49±0.7	18.68	-	18.68
	AMR	35.18	-	7942	95.42	-	95.78	18.87	-	18.87
	AC	1.17	-	12668	91.20	-	92.30	9.09	-	9.09
	Baseline	13.9	-	130269	96.52	-	96.74	100	-	100
Malware 0.4	MR	14.24±1.07	-	54702.4±10385	96.16±0.4	-	96.23±0.45	21.1	-	21.1
	AMR	39.45	-	32346	95.89	-	96.23	18.87	-	18.87
	AC	0.84	-	11085	88.97	-	89.73	5.32	-	5.32
	Baseline	27.66	-	260537	93.83	-	96.74	100	-	100
Malware 0.6	MR	18.26±0.642	-	64959.2±4286.7	96.16±1.06	-	97.87±0.45	16.65	-	16.65
	AMR	33.17	-	1442	95.12	-	96.01	18.87	-	18.87
	AC	0.84	-	11085	85.99	-	88.82	3.28	-	3.28
	Baseline	41.209	-	390806	96.16	-	97.87	100	-	100
Tuadromd 0.2	MR	1.39	100.8±2.26	287±4.69	98.6±0.07	98.73±0.15	98.43±0.16	11.9	11.9	11.9
	AMR	5.12	6.52	43.74	96.13	96.27	96.89	2.02	2.03	2.10
	AC	1.04	99.86	145	97.58	97.63	97.63	4.06	4.06	4.06
	Baseline	2.374	102.11	1987	98.77	98.77	98.66	100	100	100
Tuadromd 0.4	MR	2.55±0.032	96.26±2.26	466.4±16.4	98.5±0.2	98.4±0.13	98.54±0.17	11.1	11.1	11.1
	AMR	5.97	7.58	83.64	96.15	95.78	96.56	2.01	2.01	2.01
	AC	0.86	99.8	169	96.98	97.12	97.45	3.3	3.3	3.3
	Baseline	4.69	100.14	3882	97.3	97.6	98.66	100	100	100
Tuadromd 0.6	MR	3.62 ± 0.019	79.66 ± 0.47	692.6 ± 52.1	98.3 ± 0.2	98.36 ± 0.2	98.38 ± 0.2	11.8	11.8	11.8
	AMR	6.52	9.13	137.52	95.45	95.25	96.13	2.01	2.01	2.01
	AC	0.679	77.08	170	96.96	96.88	97.3	1.87	1.87	1.87
	Baseline	6.21	100.6	6476	97.6	97.3	98.66	100	100	100
Credit Default 0.2	MR	8.37 ± 0.012	2.93 ± 0.011	2121 ± 56.4	80 ± 0.03	78.1 ± 0.14	79.6 ± 0.1	0.3	0.3	0.3
	AMR	11.05	15.48	21.37	78.10	78.14	78.10	0.08	0.08	0.08
	AC	14.491	15.43	94	78.3	78.16	78.2	0.125	0.125	0.125
	Baseline	23.20	5.14	7071	78.1	80.1	80.3	100	100	100
Credit Default 0.4	MR	11.77 ± 0.012	2.64 ± 0.059	4242.6 ± 74	80.31 ± 0.03	80.1 ± 0.07	80.4 ± 0.03	0.3	0.3	0.3
	AMR	12.37	16.75	29.57	78.14	78.12	78.12	0.08	0.08	0.08
	AC	18.07	18.78	96	79.76	79.1	80.01	0.19	0.19	0.19
	Baseline	38.56	5.05	14263	79.6	78.1	78.08	100	100	100
Credit Default 0.6	MR	13.56 ± 0.014	3.5 ± 0.031	6357 ± 67	79.72 ± 0.02	79.81 ± 0.093	79.75 ± 0.07	0.3	0.3	0.3
	AMR	14.12	15.79	32.15	78.14	78.12	78.12	0.08	0.08	0.08
	AC	20.87	21.31	94	79.4	79.3	79.7	0.21	0.21	0.21
	Baseline	48.04	3.902	21124	0.791	0.796	0.801	100	100	100

Tables 4 and 6 in the Main Content

Tables 4 and 6 in the main content report the results of minimal repair (MR) and almost minimal repair (AMR) on originally incomplete datasets and one synthetically corrupted dataset for SVM and linear regression, respectively, due to space limitations. Tables C and D present the full results across all originally incomplete datasets and all synthetically corrupted datasets for SVM and linear regression, respectively. These additional results are consistent with the conclusions drawn in the main content.

Following our earlier observation that a single iteration of the MR and AMR algorithms is typically sufficient to achieve downstream model performance comparable to that of full imputation, we report results from the first iteration in all these experiments. This choice balances empirical effectiveness with computational efficiency, while still reflecting the overall trends observed across datasets.

Hyperparameter Analysis for Algorithms

Hyperparameters for the Minimal Repair Algorithm

One hyperparameter in the minimal repair algorithm for SVM is the early stopping criterion, which determines whether to terminate the iteration before convergence. In the previous section, we discussed how this setting influences both the overall runtime and the downstream model performance.

Table D: Accuracy and runtime of model-based imputation methods for Linear Regression

Data Set	Method	Time(s)				MSE				Impute % of Samples or Features			
		KNN	MICE	TCSDI	AC	KNN	MICE	TCSDI	AC	KNN	MICE	TCSDI	AC
Cancer Rate[2]	MR	0.153	0.574	5852	0.0212	0.0045	0.0045	0.0045	0.0067	33.33	33.33	33.33	1.69
	AMR	0.213	0.234	91.23		0.0047	0.0045	0.0045		1.46	1.44	1.46	
	Baseline	0.584	0.664	6104		0.0045	0.0058	0.0049		100	100	100	
Air Quality[18]	MR	1.06	2.46	14976	0.031	5.671	5.74	5.817	28.93	50	50	50	0.7
	AMR	0.815	0.956	901		6.015	5.991	5.978		4.78	4.35	4.90	
	Baseline	1.763	2.46	18372		5.672	5.923	5.825		100	100	100	
Communities[17]	MR	26.74	28863	-	-	0.023	0.026	-	-	75	75	-	-
	AMR	43.18	679.5	-		0.020	0.024	-		1.98	1.98	-	
	Baseline	26.72	33475	-		0.019	0.024	-		100	100	-	
Superconductivity 0.2	MR	2.369	33.16	29165	2.2	0.0089	0.00897	0.00904	0.01	70	70	70	0.24
	AMR	4.97	6.03	29.56		0.0088	0.0091	0.0085		0.07	0.07	0.07	
	Baseline	0.0692	34.17	30515		0.0088	0.008939	0.009013		100	100	100	
Superconductivity 0.4	MR	2.599	33.73	29735	2.228	0.0089	0.00924	0.00914	0.0099	75.00	75.00	75.00	0.25
	AMR	6.53	8.12	32.13		0.0087	0.0089	0.0091		0.07	0.07	0.07	
	Baseline	0.067	34.34	30783		0.0089	0.00924	0.00914		100	100	100	
Superconductivity 0.6	MR	2.541	33.24	30659	1.465	0.0089	0.01027	0.00924	0.0103	75.00	75.00	75.00	0.25
	AMR	8.12	12.54	35.46		0.0102	0.0089	0.0093		0.07	0.07	0.07	
	Baseline	0.0681	34.41	30637		0.0089	0.0104	0.00924		100	100	100	
Gas 0.2	MR	0.566	35.05	4160	0.0734	0.1079	0.1069	0.1098	0.279	65	65	65	2.01
	AMR	1.92	14.24	27.13		0.167	0.163	0.127		0.58	0.58	0.58	
	Baseline	0.4472	38.67	4267		0.1056	0.1073	0.1096		100	100	100	
Gas 0.4	MR	0.781	4096	35.8	0.051	0.1121	0.1161	0.1101	0.296	65.00	65.00	65.00	2.01
	AMR	1.42	16.45	31.14		0.197	0.167	0.163		0.58	0.58	0.58	
	Baseline	0.7151	40.64	4290		0.1133	0.1055	0.1109		100	100	100	
Gas 0.6	MR	0.566	35.05	5098	0.033	0.1983	0.1626	0.1166	0.355	65.00	65.00	65.00	1.78
	AMR	1.89	21.99	38.46		0.163	0.173	0.127		0.58	0.58	0.58	
	Baseline	0.447	38.67	5227		0.185	0.192	0.2001		100	100	100	
Concrete 0.2	MR	0.03	0.0501	269	0.0126	0.015	0.0152	0.0155	0.0267	50.00	50.00	50.00	6.89
	AMR	0.081	0.097	4.03		0.0159	0.0159	0.0158		1.46	1.46	1.46	
	Baseline	0.0175	0.0627	273		0.015	0.0152	0.156		100	100	100	
Concrete 0.4	MR	0.0383	0.0501	532	0.0149	0.015	0.0163	0.0161	0.0293	50.00	50.00	50.00	5.63
	AMR	0.124	0.131	8.01		0.0159	0.0159	0.0159		1.46	1.46	1.46	
	Baseline	0.0238	0.0582	536		0.015	0.0162	0.0161		100	100	100	
Concrete 0.6	MR	0.0409	0.0504	715	0.0065	0.0151	0.0197	0.0164	0.0356	50.00	50.00	50.00	5.28
	AMR	0.119	0.157	10.68		0.0159	0.0162	0.0162		1.46	1.46	1.46	
	Baseline	0.0319	0.0561	724		0.0151	0.0191	0.0168		100	100	100	

Table E: Result for Tuadromd dataset with 20% injected missing values (SVM) across AMR iterations with various error thresholds

AMR Iteration	Ground Truth Accuracy (%)	Time(s)			Accuracy(%)			Impute % of Samples		
		$e = 0.1$	$e = 0.2$	$e = 0.5$	$e = 0.1$	$e = 0.2$	$e = 0.5$	$e = 0.1$	$e = 0.2$	$e = 0.5$
1	98.67	3.50	2.89	3.88	99.21	98.99	99.22	2.10	2.10	2.10
3		9.68	9.73	10.52	98.88	98.88	98.89	6.00	6.02	6.02
6		19.32	19.33	20.59	99.10	99.10	99.10	11.76	11.76	11.76

Hyperparameters for the Almost Minimal Repair Algorithm

The almost minimal repair algorithm involves three major tunable hyperparameters: the error threshold e for identifying an approximately certain model, the number of edge repairs s sampled in ST1, and the maximum number of incomplete samples selected for repair in each iteration of ST2, expressed as a ratio r of the current number of incomplete samples. We analyze the impact of these hyperparameters on experimental outcomes using the TUADROMD dataset with a 20% missing rate. The effect of varying e is shown in Table E, that of varying s is presented in Table F, and that of varying r is reported in Table G. For all experiments, we vary one hyperparameter at a time while fixing the others: we fix $s = 20$ and $r = 0.02$ when varying e; fix $e = 0.2$ and $r = 0.02$ when varying s; and fix $e = 0.2$ and $s = 20$ when varying r.

For the error threshold e, we observe that all tested values in the range $[0.1, 0.5]$ produce comparable downstream accuracy and imputation cost by iteration 6, without hitting any convergence limits. This suggests that extremely small error thresholds are not necessary in practice. In fact, decent downstream performance is already observed by iteration 1 across all e values. These results indicate that users need not reach a very small suboptimality gap to gain meaningful repair benefits. We select $e = 0.2$ in the main experiments as a balanced and efficient default. For the number of sampled edge repairs s, increasing s raises the time cost due to more extensive candidate evaluation, while both the downstream accuracy and the percentage of imputed samples remain largely unchanged. This suggests that the range $s \in [10, 50]$ is sufficient to obtain a reliable approximation of $w^{\approx}$, and we use $s = 20$ as an efficient default in our experiments. Finally, for the selection ratio r, increasing r leads to more samples being imputed in each iteration, as expected, but has little effect on runtime or downstream model performance. We select $r = 0.02$ as a practical trade-off between repair effort and computational efficiency.

Table F: Results for Tuadromd dataset with 20% injected missing values (SVM) across AMR iterations with varying number of sampled edge repairs in ST1

AMR Iteration	Ground Truth Accuracy (%)	Time(s)			Accuracy(%)			Impute % of Samples		
		$s = 10$	$s = 20$	$s = 50$	$s = 10$	$s = 20$	$s = 50$	$s = 10$	$s = 20$	$s = 50$
1	98.67	1.14	2.89	14.74	98.76	98.99	98.54	2.10	2.10	2.10
3		3.29	9.73	43.19	98.77	98.88	98.77	6.02	6.02	6.02
6		6.33	19.33	82.31	98.77	99.10	98.10	11.76	11.76	11.76

Table G: Results for Tuadromd dataset with 20% injected missing values (SVM) across AMR iterations with varying selection ratio r for ST2

AMR Iteration	Ground Truth Accuracy (%)	Time(s)			Accuracy(%)			Impute % of Samples		
		$r = 0.01$	$r = 0.02$	$r = 0.05$	$r = 0.01$	$r = 0.02$	$r = 0.05$	$r = 0.01$	$r = 0.02$	$r = 0.05$
1	98.67	3.79	2.89	3.26	98.66	98.99	98.77	1.12	2.10	5.04
3		11.04	9.73	9.20	98.66	98.88	98.77	3.22	6.02	14.42
6		24.56	19.33	17.38	98.43	99.10	98.88	6.16	11.76	26.75

Minimal Repair for Linear Regression

Algorithm for finding minimal repair

Orthogonal Matching Pursuit (OMP) provides an efficient approximation for solving the sparse linear regression problem [19]. Essentially, this greedy algorithm begins with an empty solution set and initializes the regression residual to the label vector. In each iteration, the algorithm selects the feature most relevant to the current residual (i.e., having the largest dot product), adds it to the solution set, retains a linear regression model, and updates the residual accordingly. The program stops when the regression residue is sufficiently small. Therefore, OMP will return a subset of features (the solution set) that are sufficient to achieve an optimal linear regression model.

In this paper, we propose a variant of OMP, as outlined in Algorithm A, to find minimal repair for linear regression. Our algorithm has two major differences compared to the conventional OMP. Firstly, we include all complete features in the regression at the initialization, ensuring that we minimize the number of non-zero coefficients only among incomplete features. Secondly, we define our stopping condition by the maximum relevance (cosine similarity) between the feature and the label being smaller than or equal to a user-defined threshold, instead of relying on a near-zero regression residue. This approach enables our algorithm to work with general datasets without requiring the assumption of an underdetermined linear system, which is typically necessary in conventional OMP.

Algorithm A Approximating minimal repair for linear regression efficiently

```
$S_{min} \leftarrow [\quad]$
$MVF(\mathbf{z}) \leftarrow$ set of incomplete features
$Complete(\mathbf{z}) \leftarrow$ set of complete features
$\mathbf{r} \leftarrow LR(Complete(\mathbf{z}), \mathbf{y})$ {The residue vector from performing linear regression between complete features and label}
$\epsilon \leftarrow$ a user-defined threshold for stopping condition
$MaxCosSim \leftarrow \max_{\mathbf{z} \in MVF(\mathbf{z})} |cos(\mathbf{z}, \mathbf{r})|$
while $MaxCosSim \leq \epsilon$ do
    $S_{min} \leftarrow S_{min}.add(\arg\max_{\mathbf{z} \in MVF(\mathbf{z})} |cos(\mathbf{z}, \mathbf{r})|)$
    $\mathbf{r} \leftarrow LR(Complete(\mathbf{z}) \cup S_{min}, \mathbf{y})$
    $MaxCosSim \leftarrow \max_{\mathbf{z} \in MVF(\mathbf{z})} |cos(\mathbf{z}, \mathbf{r})|$
end while
$res \leftarrow S_{min}$
```

As mentioned in the main content, the time complexity of the algorithm is $\mathcal{O}(T_{train} \cdot |MVF(\mathbf{z})|)$, making it significantly more efficient than the baseline algorithm, which trains models over all repairs individually and has a time complexity of $\mathcal{O}(T_{train} \cdot |\mathbf{X}^R|)$. If a gradient descent algorithm is used, Algorithm A has a time complexity of $\mathcal{O}(n \cdot d^3)$, where n is the number of training samples and d is the number of features. In cases where $n < d^2$, the time complexity can be reduced to $\mathcal{O}(n \cdot d^2 + n^2 \cdot d)$ under certain conditions by applying incremental learning techniques based on the Sherman-Morrison formula, as outlined below.

Optimization for Algorithm A

The primary time cost in Algorithm A arises from the need to completely retrain the linear regression model each time a new imputed feature is added to the feature set. This retraining leads to a time complexity of $\mathcal{O}(n \cdot d^3)$ for the algorithm. To address this inefficiency, we propose an optimization using the Sherman-Morrison formula to update the inverse of the feature matrix incrementally [3]. This method reduces the time complexity of including one new feature to $\mathcal{O}(n^2)$. Consequently, when $n < d^2$, this optimization results in significant time savings.

Given a feature matrix $\mathbf{X}$, a label vector $\mathbf{y}$, and the coefficients $\mathbf{w}$ of the current linear regression model, our objective is to efficiently update $\mathbf{w}$ to incorporate a newly imputed feature vector $\mathbf{x}_{\text{new}}$ into $\mathbf{X}$, forming an updated feature matrix $\mathbf{X}'$, without the necessity of full retraining. When this new feature vector $\mathbf{x}_{\text{new}}$ is added to $\mathbf{X}$, it modifies the original matrix product $\mathbf{X}^T\mathbf{X}$ to $\mathbf{X}^T\mathbf{X} + \mathbf{x}_{\text{new}}\mathbf{x}_{\text{new}}^T$. Applying the Sherman-Morrison formula, the updated inverse of $\mathbf{X}'^T\mathbf{X}'$ (assuming $\mathbf{X}'^T\mathbf{X}'$ is invertible) is given by:

$$(\mathbf{X}'^T\mathbf{X}')^{-1} = (\mathbf{X}^T\mathbf{X})^{-1} - \frac{(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_{\text{new}}\mathbf{x}_{\text{new}}^T(\mathbf{X}^T\mathbf{X})^{-1}}{1 + \mathbf{x}_{\text{new}}^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_{\text{new}}} \tag{1}$$

This formulation enables the efficient update of the regression coefficients $\mathbf{w}$, requiring only $O(n^2)$ operations. Implementing at most d such updates results in a complexity of $\mathcal{O}(d \cdot n^2)$. Including the initial model training $\mathcal{O}(d^2 \cdot n)$, the total computational complexity is thus reduced to $\mathcal{O}(n \cdot d^2 + n^2 \cdot d)$.

Minimal Repair: Feature-wise or Sample-wise

For linear SVM, minimal repair (MR) is defined at the sample level—the algorithm returns a set of samples to repair. This is because the method identifies potential support vectors, which are inherently defined based on individual samples.

In contrast, for linear regression, MR is defined at the feature level—the algorithm selects a subset of features to repair. This stems from the interpretation of linear regression as projecting the residual vector onto the feature space. The approach identifies features that do not contribute to minimizing the training loss, given the current regression residual.

Proof

Proof for Theorem 1

Prove the theorem by contradiction. Assume that given a training set $(\mathbf{X}, \mathbf{y})$ and a regularization parameter C, two minimal repair sets exist ($\mathbf{S}_{min1}(\mathbf{X}, \mathbf{y}, C)$ and $\mathbf{S}_{min2}(\mathbf{X}, \mathbf{y}, C)$). From the definition of minimal repair set, a certain model exists by either imputing all samples in $\mathbf{S}_{min1}(\mathbf{X}, \mathbf{y}, C)$ or $\mathbf{S}_{min2}(\mathbf{X}, \mathbf{y}, C)$, regardless of imputation results. Further, based on the discussion in previous literature [22], a certain model exists when none of the incomplete samples is a support vector in any repair. Therefore, if an incomplete sample is not in the minimal repair set, it is not a support vector in any repair. From the assumption, we can always find an incomplete sample $\mathbf{x}_i$ that $\mathbf{x}_i \notin \mathbf{S}_{min1}(\mathbf{X}, \mathbf{y}, C)$ and $\mathbf{x}_i \in \mathbf{S}_{min2}(\mathbf{X}, \mathbf{y}, C)$. In this scenario, $\mathbf{x}_i$ is not a support vector for any repair of $\mathbf{X}$ because $\mathbf{x}_i \notin \mathbf{S}_{min1}(\mathbf{X}, \mathbf{y}, C)$. Thus, $\mathbf{S}_{min2}(\mathbf{X}, \mathbf{y}, C)$ is not a minimal repair set because removing $\mathbf{x}_i$ from $\mathbf{S}_{min1}(\mathbf{X}, \mathbf{y}, C)$ should construct a smaller set also ensuring the existence of certain models, violating the definition of minimal repair set. Contradicting to the original assumption, Theorem **??** holds.

Proof for Lemma 1

Borrowing the discussion from proving Theorem 1, if an incomplete sample $\mathbf{x}_i$ is not a support vector in any repair of $\mathbf{X}$, it should not be part of the minimal repair set S_{min} (which is unique from Theorem 1). Further, if an incomplete sample $\mathbf{x}_i$ is a support vector in at least one repair of $\mathbf{X}$, it has to be included in the minimal repair set, otherwise certain model does not exist [22].

Proof for Theorem 3

Necessity is trivial based on Lemma 2: if an incomplete sample is a support vector in an edge repair, the incomplete sample is part of the minimal repair set. Then we prove sufficiency by contradiction. Assume that there is an incomplete sample $\mathbf{x}_i$ part of the minimal repair set X_{min} while it is not a support vector in any edge repair $\mathbf{x}^e \in \mathbf{X}^E$. Training an SVM can be interpreted as finding the minimal distance between two reduced convex hulls [4], and if an sample is within the reduced convex hull (not at the boundary), the sample is not a support vector. Because $\mathbf{x}_i$ is not a support vector for any edge repair from the assumption, it is not a support vector for any repair to $\mathbf{X}$. This is because, in the process of changing a value for a missing value (x_{pq}) from one edge repair (x_{pq}^{min}) to another (x_{pq}^{max}) monotonically increase or decrease the coverage of the reduced convex hull. With that being said, if an incomplete sample $\mathbf{x}_i$ is not a support vector for any edge repair (i.e., within the reduced convex hull), the incomplete sample is within the reduced convex hull (i.e., not a support vector) with respect to any repair. This contradicts to the original assumption that $\mathbf{x}_i$ is part of the minimal repair set.

Proof for Theorem 4

We reduce from the NP-complete problem 3-SAT. Let

$$\Phi \;=\; \bigwedge_{j=1}^{m} (C_j)$$

be a 3-SAT formula with k Boolean variables $z_1, z_2, \ldots, z_k$ and m clauses $C_1, \ldots, C_m$, each clause being a disjunction of three literals.

For each variable z_ℓ, we introduce one or more *incomplete* samples whose feature vectors each contain a *missing* coordinate u_ℓ. The imputation set for u_ℓ is $\{-1, +1\}$, corresponding to $\{\text{False}, \text{True}\}$. Thus, any assignment of the z_ℓ corresponds to choosing ± 1 for these missing coordinates.

To enforce that each clause C_j must be satisfied, we add appropriately labeled points (some possibly incomplete) and arrange them in a geometry so that assigning a literal to *false* yields a large penalty term in the soft-margin objective (either by misclassification or forcing the margin to collapse). Intuitively, if a clause were unsatisfied (all literals set to *false*), the SVM would incur a prohibitively large hinge-loss cost, making that repair suboptimal.

We designate one particular incomplete sample $\mathbf{x}_i$ with additional coordinates or constraints so that:

- *If* Φ is satisfiable, then there is an imputation (choosing ± 1 consistently with a satisfying assignment) that maximizes the margin while placing $\mathbf{x}_i$ *exactly on* the decision boundary, making it a support vector.
- *If* Φ is unsatisfiable, then *every* imputation leads to $\mathbf{x}_i$ being off the margin (either strictly inside or otherwise not a support vector). In other words, no selection of $\{\pm 1\}$ for the missing attributes can force $\mathbf{x}_i$ onto the margin.

By suitably tuning the soft-margin parameter C and the placement of the clause-encoding points, we ensure that the SVM will "prefer" to assign ± 1 values in a way that satisfies Φ, whenever possible, in order to avoid a large penalty.

Hence,

$$\Phi \text{ is satisfiable} \quad \Longleftrightarrow \quad \text{there exists a repair making } \mathbf{x}_i \text{ a support vector.}$$

Since deciding satisfiability for Φ (3-SAT) is NP-complete, it follows that deciding whether $\mathbf{x}_i$ can be a support vector under some imputation is NP-hard.

Determining membership of a single incomplete sample $\mathbf{x}_i$ among the possible support vectors is NP-hard. Therefore, listing *all* such samples that can ever appear on the margin is also NP-hard: if we had such a list in polynomial time, we could decide membership in that list in polynomial time, contradicting NP-hardness. Given the proof that finding MR for SVM is NP-hard, deciding whether an incomplete sample belongs to the MR for SVM is also NP hard. To prove, assume that we have a polynomial-time solver for deciding whether an incomplete sample belongs to the MR, then one can linearly scan each incomplete sample and decide its membership in MR (either belongs to or not) by calling the polynomial time subroutine. Therefore, one can find the MR in polynomial time, which contradicts to the NP-hard proof earlier.

Proof for Theorem 5

For any incomplete sample $\mathbf{x}_i$ returned from Algorithm 1 in main content for SVM, the incomplete sample is a support vector in at least one repair to $\mathbf{X}$. Based on Theorem 3, it is part of the minimal repair.

Proof for Theorem 6

Given the iterative algorithm of finding the minimal repair for SVM (Algorithm 1 in the main content), we first characterize the probability that the imputation set returned at iteration k misses one or more incomplete samples that belong to the minimal repair.

Let k be the current iteration index ($k = 0$ represents the initial state before the first run). We define the following: $MS(x)^k$ is the set of incomplete samples remaining at the start of iteration k. $M^k = |MS(x)^k|$ is the number of remaining incomplete samples at the start of iteration k. S_{min}^k is the (unknown) true minimal set of samples within $MS(x)^k$ that must be imputed at the start of iteration k to guarantee a certain model. $s^k = |S_{min}^k|$ is the (unknown) size of this true minimal set; note that we treat s^k as a random variable, and $s^k \leq M^k$. S'^k is the set of samples returned by Algorithm 1 in the main content when run at iteration k on the current data; we know $S'^k \subseteq S_{min}^k$. FN^k is the event that makes at least one false negative error at iteration k, occurring if S'^k is a proper subset of S_{min}^k. $P(FN^k)$ is the probability of event FN^k. We seek a computable upper bound UB'^k such that $P(FN^k) \leq UB'^k$. define p_{fn} as an upper bound on the per-sample false negative probability, $p(\mathbf{x_i})$. We assume that there exists a probability p_{fn} (where $0 \leq p_{fn} \leq 1$) such that for any sample $x_i \in S_{min}^k$, the probability that Algorithm 1 in the main content fails to include x_i in S'^k is bounded above by p_{fn}:

$$P(x_i \notin S'^k | x_i \in S_{min}^k) \leq p_{fn}$$

Then we propose, UB'^k, an upper bound of $P(FN^k)$ as follows:

$$UB'^k = 1 - (1 - p_{fn})^{M^k} \geq P(FN^k)$$

To interpret, when the iteration goes (k becomes larger), M^k and p_{fn} decrease (which we will prove later), UB'^k decreases. This indicates that the upper-bound of probability of under-imputing decreases over iterations.

To prove this bound, we begin by expressing the target probability $P(FN^k)$ using its complement. The event FN^k (at least one false negative) is the complement of the event $NoFN^k$ (no false negatives, i.e., $S'^k = S_{min}^k$). Therefore, conditioned on the true size s^k of the minimal set at iteration k, we have $P(FN^k|s^k) = 1 - P(\text{No FN}^k|s^k)$.

Next, we bound the probability of having no false negatives, $P(\text{No FN}^k|s^k)$. The event $NoFN^k$ occurs if Algorithm 1 in the main content successfully returns all samples in S_{min}^k. Let E_i be the event that Algorithm 1 in the main content fails to return sample x_i. Assuming the failure/success events E_i for different samples $x_i \in S_{min}^k$ within the same iteration k are statistically independent, we can write:

$$P(\text{No FN}^k|s^k) = P(\cap_{x_i \in S_{min}^k} \{\text{not } E_i\}|s^k) = \prod_{x_i \in S_{min}^k} P(\text{not } E_i|s^k)$$

Let $P(E_i|s^k)$ be the probability of failure for x_i. Then $P(\text{not } E_i|s^k) = 1 - P(E_i|s^k)$. Using the definition $P(E_i|s^k) \leq p_{fn}$, we have $1 - P(E_i|s^k) \geq 1 - p_{fn}$. Substituting this lower bound into the product gives:

$$P(\text{No FN}^k|s^k) \geq \prod_{i=1}^{s^k} (1 - p_{fn}) = (1 - p_{fn})^{s^k}$$

Now we can bound $P(FN^k|s^k)$:

$$P(FN^k|s^k) = 1 - P(\text{No FN}^k|s^k) \leq 1 - (1 - p_{fn})^{s^k}$$

The overall probability $P(FN^k)$ is the expectation over the unknown size s^k:

$$P(FN^k) = \mathop{\mathbb{E}}_{s^k}[P(FN^k|s^k)] \leq \mathop{\mathbb{E}}_{s^k}[1 - (1 - p_{fn})^{s^k}]$$

To proceed, we utilize Jensen's inequality. Let $f(s) = 1 - (1 - p_{fn})^s$. We first prove that $f(s)$ is concave for $s \geq 0$. Let $b = 1 - p_{fn}$. Since $0 \leq p_{fn} < 1$, we have $0 < b \leq 1$. The function is $f(s) = 1 - b^s$. The first derivative is $f'(s) = -b^s \ln(b)$. The second derivative is $f''(s) = -(b^s \ln(b)) \ln(b) = -b^s (\ln(b))^2$. Since $b^s > 0$ and $(\ln(b))^2 \geq 0$, the second derivative $f''(s) \leq 0$. Therefore, $f(s)$ is a concave function.

Jensen's inequality for a concave function f states $\mathbb{E}[f(X)] \leq f(\mathbb{E}[X])$. Applying this to our expectation:

$$\mathop{\mathbb{E}}_{s^k}[1 - (1 - p_{fn})^{s^k}] \leq 1 - (1 - p_{fn})^{\mathbb{E}[s^k]}$$

Combining this with the previous inequality gives a theoretical upper bound:

$$P(FN^k) \leq 1 - (1 - p_{fn})^{\mathbb{E}[s^k]}$$

The term $\mathbb{E}[s^k]$ (expected number of truly needed samples) is still unknown. However, we know that the number of needed samples s^k cannot exceed the total number of remaining incomplete samples $M^k = |MS(x)^k|$. Thus, $s^k \leq M^k$. Taking expectations yields $\mathbb{E}[s^k] \leq \mathbb{E}[M^k]$. Since M^k is a known quantity (computable by counting) at the start of iteration k, $\mathbb{E}[M^k] = M^k$. Therefore, we have a computable upper bound for the expectation: $\mathbb{E}[s^k] \leq M^k$.

Finally, we substitute this bound on $\mathbb{E}[s^k]$ into the Jensen result. Let $g(x) = (1 - p_{fn})^x$. Since $0 < (1 - p_{fn}) \leq 1$, $g(x)$ is a non-increasing function. Applying g to the inequality $\mathbb{E}[s^k] \leq M^k$ reverses the inequality direction:

$$(1 - p_{fn})^{\mathbb{E}[s^k]} \geq (1 - p_{fn})^{M^k}$$

Multiplying by -1 and adding 1 (reversing the inequality twice):

$$1 - (1 - p_{fn})^{\mathbb{E}[s^k]} \leq 1 - (1 - p_{fn})^{M^k}$$

Combining the inequalities $P(FN^k) \leq 1 - (1 - p_{fn})^{\mathbb{E}[s^k]}$ and $1 - (1 - p_{fn})^{\mathbb{E}[s^k]} \leq 1 - (1 - p_{fn})^{M^k}$, we arrive at the final upper bound UB'^k:

$$P(FN^k) \leq 1 - (1 - p_{fn})^{M^k}$$

and

$$UB'^k = 1 - (1 - p_{fn})^{|MS(x)^k|}$$

Now the only problem is to compute p_{fn} and understand how it changes over iterations. The Multiple Random Starts method provides an empirical approach. First, select a set of incomplete samples MS_{probe} (e.g., $MS(x)^0$) and choose the number of repetitions T (e.g., $T = 10$ or 20). For each $x_i \in MS_{probe}$, initialize a success count $t_i = 0$. Repeat T times: generate a new random edge repair $X^e_{start,t}$ for the current dataset state; run the greedy construction part of Algorithm 1 in the main content starting from $X^e_{start,t}$ to get $X^e_{final,i,t}$; train $w_{final,i,t} = SVM(X^e_{final,i,t}, y)$; check if $y_i (w_{final,i,t})^T (x_i \text{ part of } X^e_{final,i,t}) \leq 1$. If yes, increment t_i.

Also, if the probability distribution of each incomplete sample is known, and we let $g(x_{ij})$ denote the probability density function of the ground truth value for the missing value x_{ij} in the incomplete training set $(\mathbf{X}, \mathbf{y})$. If missing values in $\mathbf{X}$ are independent, the probability that an incomplete sample $\mathbf{x}_i$ in minimal repair not returned by Algorithm 1 in the main content is:

$$p(\mathbf{x}_i) = 1 - \frac{\int \cdots \int_{\min(x_{ij}^{\text{visited}})}^{\max(x_{ij}^{\text{visited}})} \prod_{x_{ij} \in M(\mathbf{X})} g(x_{ij})\, dx_{ij}}{\int \cdots \int_{x_{ij} \in M(\mathbf{X})} \prod_{x_{ij} \in M(\mathbf{X})} g(x_{ij})\, dx_{ij}}$$

$x_{ij}^{\text{visited}} \in \{x_{ij}^{min}, x_{ij}^{max}\}$ shows the values used for x_{ij} in Algorithm 1 in the main content. It shows that the more edge repairs Algorithm 1 explores, the lower the false negative probability for each sample. One can find p_{fn} by computing $p(\mathbf{x}_i)$ for each incomplete sample and take the maximum as p_{fn}. p_{fn} decreases over iterations because each iteration explores additional edge repairs. This expands the domain of the numerator in the expression increasing the integral value and thereby lowering $p(\mathbf{x}_i)$ for every sample5. Since p_{fn} is an upper bound over all such $p(\mathbf{x}_i)$, it decreases as well.

Proof for Theorem 7

Prove the possibility of having multiple minimal repair sets first. Because linear regression can have multiple non-trivial optimal models in general, multiple minimal repair sets can exist, and each multiple imputation set corresponds to an optimal linear regression model. For example, when we have the dataset:

$$X = \begin{bmatrix} 1 & \text{null} & \text{null} & \text{null} \\ 0 & 1 & 2 & 3 \\ 0 & 4 & 3 & 2 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

We denote features from left to right as $\mathbf{z}_1 \ldots \mathbf{z}_4$. In this example, there are at least two MRs, $MR_1 = \{\mathbf{z}_2, \mathbf{z}_3\}$ and $MR_2 = \{\mathbf{z}_3, \mathbf{z}_4\}$. To prove, we first show that imputing either MR_1 or MR_2, and training a linear regression model with imputed features and the originally complete feature ($\mathbf{z}_1$) leads to a zero (minimal) regression loss in all repairs of X. Let us first consider MR_1. The two incomplete features ($\mathbf{z}_2$ and $\mathbf{z}_3$) with the complete one ($\mathbf{z}_1$) cover the full 3-dimensional space in all repairs because the three features are linearly independent in all repairs. We show the linear independence by computing the determinant of the matrix A consisting of $\mathbf{z}_1$, $\mathbf{z}_2$, and $\mathbf{z}_3$.

$$A = \begin{bmatrix} 1 & \text{null} & \text{null} \\ 0 & 1 & 2 \\ 0 & 4 & 3 \end{bmatrix}$$

The determinant of the matrix A is non-zero regardless of how the null values in $\mathbf{z}_2$ and $\mathbf{z}_3$ are imputed.

$$\begin{aligned} \det(A) = \det(A^T) = 1 \cdot \det\begin{pmatrix} 1 & 4 \\ 2 & 3 \end{pmatrix} - 0 \cdot \det\begin{pmatrix} null & 4 \\ null & 3 \end{pmatrix} + 0 \cdot \det\begin{pmatrix} null & 1 \\ null & 2 \end{pmatrix} \\ = 1 \cdot ((1)(3) - (2)(4)) \\ = 1 \cdot (3 - 8) \\ = -5 \end{aligned}$$

Because $\mathbf{z}_1$, $\mathbf{z}_2$, and $\mathbf{z}_3$ are linearly independent, for every repair of A, there is a linear regression model that achieves zero (minimal) loss with the feature matrix A and the label vector y. Let $v(\mathbf{z}_2)$ and $v(\mathbf{z}_3)$ denote a repair of columns (features) $\mathbf{z}_2$ and $\mathbf{z}_3$ in A, respectively. Every repair of the matrix X with $v(\mathbf{z}_2)$ and $v(\mathbf{z}_3)$ for its second and third columns, no matter what the imputation of missing value in $\mathbf{z}_4$ is, will have zero regression loss for the label vector y.

Similarly, for MR_2, we show that the two incomplete features ($\mathbf{z}_3$ and $\mathbf{z}_4$) along with the complete $\mathbf{z}_1$ cover the full 3-dimensional space in all repairs because the three features are linearly independent in all repairs. We show this by computing the determinant of the matrix B consisting of $\mathbf{z}_1$, $\mathbf{z}_3$, and $\mathbf{z}_4$.

$$B = \begin{bmatrix} 1 & \text{null} & \text{null} \\ 0 & 2 & 3 \\ 0 & 3 & 2 \end{bmatrix}$$

The determinant of B is non-zero in all repairs.

$$\begin{aligned} \det(B) = \det(B^T) = 1 \cdot \det\begin{pmatrix} 2 & 3 \\ 3 & 2 \end{pmatrix} - 0 \cdot \det\begin{pmatrix} null & 3 \\ null & 2 \end{pmatrix} + 0 \cdot \det\begin{pmatrix} null & 2 \\ null & 3 \end{pmatrix} \\ = 1 \cdot ((2)(2) - (3)(3)) \\ = 1 \cdot (4 - 9) \\ = -5 \end{aligned}$$

Therefore, similar to our argument for MR_1, the regression loss for every repair of the features of MR_2 in the linear regression with feature matrix X and label vector y is zero (minimal) no matter what the imputation of the missing value in $\mathbf{z}_2$ is.

To close the proof for MR_1 and MR_2 being minimal repairs, we also show that there is no smaller subset (with only one incomplete feature) such that by imputing the subset and training a linear regression model with the imputed feature and the originally complete feature $\mathbf{z}_1$ leads to the minimal regression loss in all repairs. By scanning every single incomplete feature, no one can achieve the minimal regression loss along with the complete feature ($\mathbf{z}_1$) in all repairs. Therefore, the size of MR should be 2, which concludes the proof that MR_1 and MR_2 are both minimal repairs in this example dataset. However, when all features in $\mathbf{X}$ are linearly independent in all repairs, the optimal linear regression model is unique for every repair. Therefore, a certain model is unique when it exists in this scenario, and the minimal repair set is also unique to reach a certain model.

Proof for Theorem 8

To prove that finding the linear regression solution that is most sparse over a subset of features is NP-hard, we reduce the known NP-hard problem of finding the most sparse linear regression solution to it [6]. Consider the original problem where given a feature matrix $\mathbf{X}$ and a label vector $\mathbf{y}$, the goal is to find the optimal model $\mathbf{w}^*$ that minimizes the number of non-zero entries. In the new problem, given a subset of features, i.e., the incomplete features, denoted as $MVF(\mathbf{X})$, we seek the optimal model $\mathbf{w}^*$ that minimizes the number of non-zero entries in the coefficients within $MVF(\mathbf{X})$. To reduce the original problem to this new one, set $MVF(\mathbf{X})$ as the entire feature set. Solving the new problem in this special case is equivalent to solving the original sparse linear regression problem, which is NP-hard. Therefore, the new problem must also be NP-hard, as it generalizes the original problem.

Proof for Lemma 9

Based on the previous literature about certain model [22], when a certain model $\mathbf{w}^*$ exists for linear regression, $w_i = 0$ for every $\mathbf{z}_i \in MVF(\mathbf{X})$. Therefore, finding a minimal repair set in linear regression is equivalent to finding a regression model that has the maximal number of zero model parameters (linear coefficients) and is optimal for all repairs. Further, the problem is equivalent to minimizing the number of non-zero linear coefficients in $\mathbf{w}$ whose corresponding feature is incomplete.

Proof for Theorem 10

When each missing value in the dataset follows an independent zero-mean normal distribution, training a linear regression model based on the incomplete dataset is equivalent to training linear regression with a zero-mean Gaussian noise ϵ as below:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \epsilon$$

Based on previous literature [7], in the presence of a Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$, the first k features returned from OMP method is correct with a probability of at least $1 - 1/n$ when the following two conditions are satisfied: 1. $\mu < 1/(2k-1)$, and 2.

$$|w_i| \geq \frac{2\sigma_{ij}\sqrt{n + 2\sqrt{nlogn}}}{1 - (2k-1)\mu}$$

As a result, the features returned by the OMP algorithm in our paper is correct with a probability of at least $1 - 1/n$ given the conditions in Theorem 10.

Proof for Theorem 11

The proof has two parts: (1) showing that any set of samples S'_k selected by ST2 at iteration k is a subset of S_{AMR}, implying $S_{\text{iter-ACM}} = \bigcup_k S'_k \subseteq S_{\text{AMR}}$; and (2) showing the algorithm terminates with an ACM ($g_k \leq e$).

Part 1: Each selection S'_k by ST2 belongs to S_{AMR}

S_{AMR} is the smallest set of incomplete samples in $\mathbf{X}$ whose robust imputation guarantees $g \leq e$, irrespective of specific repair values. Consider iteration k: ST1 operates on $\mathbf{X}^{(k)}$ (where $S^{(k-1)}_{\text{iter-ACM}} =$

$\bigcup_{i<k} S'_i$ are imputed) yielding g_k. If $g_k > e$, ST2 returns S'_k, the minimal set of currently incomplete samples in $\mathbf{X}^{(k)}$ necessary to enable $g < g_k$ in the next iteration.

Let $x_j \in S'_k$. Assume, for contradiction, $x_j \notin S_{\text{AMR}}$. If $x_j \notin S_{\text{AMR}}$, then S_{AMR} (not containing x_j) robustly guarantees $g \leq e$ for the original problem $(\mathbf{X}, \mathbf{y})$. So, x_j is not required for this global robust guarantee. At iteration k, ST2 identifies x_j as part of the minimal set S'_k in $\mathbf{X}^{(k)}$ needed to reduce g_k. This implies x_j is locally indispensable for progress from $\mathbf{X}^{(k)}$.

Let $S^*_{\text{AMR}} = S_{\text{AMR}} \cap U^{(k)}$ be the S_{AMR} samples still incomplete in $\mathbf{X}^{(k)}$. By induction ($S^{(0)}_{\text{iter-ACM}} = \emptyset \subseteq S_{\text{AMR}}$), all $S^{(k-1)}_{\text{iter-ACM}} \subseteq S_{\text{AMR}}$. If S_{AMR} (excluding x_j) robustly guarantees ACM for $\mathbf{X}$, and $S^{(k-1)}_{\text{iter-ACM}} \subseteq S_{\text{AMR}}$, then any local impasse $g_k > e$ must be resolvable by further imputing only samples from S^*_{AMR}. So, some $P \subseteq S^*_{\text{AMR}}$ must exist to allow g to decrease. Since ST2 returns the *minimal* set for progress, if such P exists, ST2 would select $S'_k \subseteq P \subseteq S^*_{\text{AMR}} \subseteq S_{\text{AMR}}$. This means $x_j \in S_{\text{AMR}}$, contradicting $x_j \notin S_{\text{AMR}}$.

Thus, if ST2 selects x_j (assumed $x_j \notin S_{\text{AMR}}$) as part of S'_k, it means no $P \subseteq S^*_{\text{AMR}}$ alone allows progress, and x_j is also needed. This implies x_j is locally indispensable even if all of S^*_{AMR} were imputed. This contradicts the global sufficiency of S_{AMR} (which excludes x_j). The perfection of ST2 ensures it doesn't select a globally redundant x_j if progress is possible via samples in S^*_{AMR}. So, $x_j \notin S_{\text{AMR}}$ is false. Thus, any $x_j \in S'_k$ is in S_{AMR}, meaning $S'_k \subseteq S_{\text{AMR}}$ for all k. Consequently, $S_{\text{iter-ACM}} = \bigcup_k S'_k \subseteq S_{\text{AMR}}$.

Part 2: Algorithm Termination with an ACM

If $g_k > e$, ST2 identifies a non-empty S'_k for imputation. (If S'_k was empty while $g_k > e$, it would contradict the existence of S_{AMR} as a solution or the ideal functioning of ST1/ST2). Imputing S'_k creates $\mathbf{X}^{(k+1)}$. The number of incomplete samples is finite. ST2 selects un-imputed samples necessary for reducing g_k. Assuming perfect ST1/ST2, the algorithm progresses towards $g_k \leq e$. It cannot impute distinct samples indefinitely nor cycle with $g_k > e$ as each ST2 selection resolves a current bottleneck. Thus, it must reach $g_k \leq e$ and terminate, achieving ACM.

The assertion $S_{\text{iter-ACM}} \subset S_{\text{AMR}}$ is consistent: S_{AMR} ensures robustness for *all* repairs. The algorithm uses specific repairs and may achieve ACM before all of S_{AMR} (needed for worst-case robustness) are imputed.

Proof for Theorem 12

We assume that the loss function $L(\mathbf{w})$ is convex and has an M-Lipschitz continuous gradient. Formally, this means for all $\mathbf{w}, \mathbf{w}' \in \mathcal{W}$:

$$\|\nabla L(\mathbf{w}) - \nabla L(\mathbf{w}')\| \leq M\|\mathbf{w} - \mathbf{w}'\|.$$

Since $L(\mathbf{w})$ is convex with an M-Lipschitz continuous gradient, the following standard inequality from convex optimization theory holds:

$$L(\mathbf{w}) \leq L(\mathbf{w}') + \nabla L(\mathbf{w}')^\top (\mathbf{w} - \mathbf{w}') + \frac{M}{2}\|\mathbf{w} - \mathbf{w}'\|^2, \quad \forall \mathbf{w}, \mathbf{w}' \in \mathcal{W}.$$

Let $\mathbf{w}^*$ be an optimal solution (thus $\nabla L(\mathbf{w}^*) = 0$), and set $\mathbf{w}' = \mathbf{w}^*$, then we have:

$$L(\mathbf{w}^{\approx}) \leq L(\mathbf{w}^*) + \frac{M}{2}\|\mathbf{w}^{\approx} - \mathbf{w}^*\|^2.$$

Next, due to convexity of $L(\mathbf{w})$, we have:

$$L(\mathbf{w}^*) \geq L(\mathbf{w}^{\approx}) + \nabla L(\mathbf{w}^{\approx})^\top (\mathbf{w}^* - \mathbf{w}^{\approx}).$$

Combining the two inequalities, we get:

$$L(\mathbf{w}^{\approx}) - L(\mathbf{w}^*) \leq \frac{M}{2}\|\mathbf{w}^{\approx} - \mathbf{w}^*\|^2 \leq \frac{1}{2M}\|\nabla L(\mathbf{w}^{\approx})\|^2,$$

where the last step follows from the Lipschitz continuity of the gradient, which implies that:

$$\|\nabla L(\mathbf{w}^{\approx})\| \geq M\|\mathbf{w}^{\approx} - \mathbf{w}^*\|.$$

Hence, the optimality gap is explicitly bounded by the norm of the gradient:

$$L(\mathbf{w}^{\approx}) - L(\mathbf{w}^*) \leq \frac{1}{2M}\|\nabla L(\mathbf{w}^{\approx})\|^2.$$

Therefore, to guarantee for all $\mathbf{X}^r \in \mathbf{X}^R$ that:

$$L(f(\mathbf{X}^r, \mathbf{w}^{\approx}), \mathbf{y}) - \min_{\mathbf{w}\in\mathcal{W}} L(f(\mathbf{X}^r, \mathbf{w}), \mathbf{y}) \leq e,$$

it is sufficient to require:

$$\|\nabla_{\mathbf{w}} L(f(\mathbf{X}^r, \mathbf{w}^{\approx}), \mathbf{y})\| \leq \sqrt{2Me}, \quad \forall \mathbf{X}^r \in \mathbf{X}^R.$$

This completes the derivation.

Code Repository

Link: https://anonymous.4open.science/r/Submission_2025-A1C0/README.md

References

[1] Alireza Aghasi, MohammadJavad Feizollahi, and Saeed Ghadimi. Rigid: Robust linear regression with missing data. *arXiv preprint arXiv:2205.13635*, 2022.

[2] AI Planet. Datasets. `https://github.com/aiplanethub/Datasets`. Accessed: 2025-05-22.

[3] Marco Angioli, Marcello Barbirotta, Abdallah Cheikh, Antonio Mastrandrea, Francesco Menichelli, and Mauro Olivieri. Efficient implementation of linearucb through algorithmic improvements and vector computing acceleration for embedded learning systems. *arXiv preprint arXiv:2501.13139*, 2025.

[4] Kristin P Bennett and Erin J Bredensteiner. Duality and geometry in svm classifiers. In *ICML*, volume 2000, pages 57–64. Citeseer, 2000.

[5] Parthajit Borah and Dhruba K. Bhattacharyya. TUANDROMD (Tezpur University Android Malware Dataset). UCI Machine Learning Repository, 2020. DOI: https://doi.org/10.24432/C5560H.

[6] Alfred M Bruckstein, David L Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1):34–81, 2009.

[7] T Tony Cai and Lie Wang. Orthogonal matching pursuit for sparse signal recovery with noise. *IEEE Transactions on Information theory*, 57(7):4680–4688, 2011.

[8] Trefor Evans and Erin Grant. uci_datasets. `https://github.com/treforevans/uci_datasets`. Accessed: 2025-05-22.

[9] Ravi Ganti and Rebecca M Willett. Sparse linear regression with missing data. *arXiv preprint arXiv:1503.08348*, 2015.

[10] Kam Hamidieh. Superconductivty Data. UCI Machine Learning Repository, 2018. DOI: https://doi.org/10.24432/C53P47.

[11] Bojan Karlaš, Peng Li, Renzhi Wu, Nezihe Merve Gürel, Xu Chu, Wentao Wu, and Ce Zhang. Nearest neighbor classifiers over incomplete information: From certain answers to certain predictions. *arXiv preprint arXiv:2005.05117*, 2020.

[12] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J Franklin, and Ken Goldberg. Activeclean: Interactive data cleaning for statistical modeling. *Proceedings of the VLDB Endowment*, 9(12):948–959, 2016.

[13] Felix Neutatz, Binger Chen, Ziawasch Abedjan, and Eugene Wu. From cleaning before ml to cleaning for ml. *IEEE Data Eng. Bull.*, 44(1):24–41, 2021.

[14] Yidong Ouyang, Liyan Xie, Chongxuan Li, and Guang Cheng. Missdiff: Training diffusion models on tabular data with missing values. *arXiv preprint arXiv:2307.00467*, 2023.

[15] Jose Picado, John Davis, Arash Termehchy, and Ga Young Lee. Learning over dirty data without cleaning. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 1301–1316, 2020.

[16] M. L. de Lima Sidney Murilo Sérgio Albuquerque Edison Souza Danilo Monteiro Thyago Lopes Petrônio Lima Rafael Oliveira Jemerson Pinheiro, Ricardo and Sthéfano Silva. REJAFADA. UCI Machine Learning Repository, 2019. DOI: https://doi.org/10.24432/C5HG8D.

[17] Michael Redmond. Communities and Crime. UCI Machine Learning Repository, 2002. DOI: https://doi.org/10.24432/C53W3X.

[18] Saverio Vito. Air Quality. UCI Machine Learning Repository, 2008. DOI: https://doi.org/10.24432/C59K5F.

[19] Jian Wang, Seokbeop Kwon, and Byonghyo Shim. Generalized orthogonal matching pursuit. *IEEE Transactions on signal processing*, 60(12):6202–6216, 2012.

[20] I-Cheng Yeh. Concrete Compressive Strength. UCI Machine Learning Repository, 1998. DOI: https://doi.org/10.24432/C5PK67.

[21] I-Cheng Yeh. Default of Credit Card Clients. UCI Machine Learning Repository, 2009. DOI: https://doi.org/10.24432/C55S3H.

[22] Cheng Zhen, Nischal Aryal, Arash Termehchy, and Amandeep Singh Chabada. Certain and approximately certain models for statistical learning. *Proc. ACM Manag. Data*, 2(3), May 2024. doi: 10.1145/3654929. URL `https://doi.org/10.1145/3654929`.

[23] Shuhan Zheng and Nontawat Charoenphakdee. Diffusion models for missing value imputation in tabular data. *arXiv preprint arXiv:2210.17128*, 2022.