Counterfactual Residual Data Augmentation for Regression

Hossein Mohebbi [1 2] **Oliver Schulte** [3] **Ke Li** [3] **Pascal Poupart** [1 2]

Abstract

Data-driven modeling in real-world regression tasks often suffers from limited training samples, high collection costs, and noisy observations. Inspired by the impact of data augmentation in vision and language, we propose a novel *Counterfactual Residual Data Augmentation (CRDA)* technique for tabular regression. Our key insight is that once a regressor has modeled the systematic component of the data, the remaining noise can be viewed as an invariant residual that remains stable under small perturbations of carefully selected features. We exploit this residual invariance to generate new, yet realistic, training samples, effectively expanding the dataset without requiring additional real data. Our method is model-agnostic and readily applicable to various types of regressors. In experiments across datasets from a variety of benchmark repositories, on average, *CRDA* reduces an *MLP Regressor*'s MSE by **22.9%** and an *XGBoost Regressor*'s MSE by **6.4%**. When compared to existing state-of-the-art data generators and augmentation techniques, CRDA consistently outperforms in MSE reduction. By adding principled counterfactual variations to the training data, our method offers a simple and efficient remedy for noise-prone, small-sample regression settings.

1. Introduction

Data scarcity and noise are frequently encountered obstacles in regression tasks across domains such as medicine, finance, and manufacturing. Collecting large-scale, high-quality data can be expensive or impractical, and existing data augmentation techniques, while well developed in computer vision and NLP, often do not translate naturally to tabular regression. As a result, many supervised learning models fail to fully capture the underlying behavior of real-world processes when only limited training examples are available.

In this paper, we propose **Counterfactual Residual Data Augmentation (CRDA)**, a simple and flexible method to bolster regression performance under small data constraints. The core idea is straightforward: *(i)* we train a base predictor (e.g., MLP or XGBoost) on a dataset, *(ii)* identify one or more features whose perturbations do not alter the residual (prediction error) distribution significantly, and *(iii)* generate new samples by modifying those features while preserving the original "noise" or residual component. To illustrate, consider a house price prediction task. A model captures systematic value drivers like location and square footage, while the residual captures unobserved factors like a *bidding war* driven by a specific buyer's urgency. Our key insight is that varying a secondary feature, such as garage finish, changes the systematic price but is unlikely to alter the specific buyer's urgency. CRDA exploits this independence to synthesize a residual-preserving counterfactual: a house with a different garage finish, an updated systematic price, but the exact same "bidding war" residual.

This pattern recurs across domains: in clinical recovery-time prediction, perturbing the amount of physical therapy changes a patient's recovery time but does not alter their unobserved physiology; in crop-yield regression, perturbing fertilizer dose shifts the predicted yield while leaving the field's unmeasured pest pressure untouched; in home energy forecasting, perturbing the thermostat setpoint moves predicted consumption but not whether a window was left open that day.

Motivation and Benefits. Our motivation stems from the difficulty of acquiring sufficient labeled data in many practical applications, coupled with the risk of overfitting when sample sizes are small. A major attraction of *CRDA* is its ability to insert new data points *without* assuming domain-specific transformations or heuristics. Instead, it relies on a learned predictor to separate systematic behavior from noise, then conserves the latter across minor interventions of selected features. As a result, the augmented samples remain consistent with the underlying distributional assumptions,

[1]University of Waterloo [2]Vector Institute [3]Simon Fraser University. Correspondence to: Hossein Mohebbi <hossein.mohebbi@uwaterloo.ca>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

Project page with reproducible experiment code and Python package: https://crda-project.github.io/.

improving model fit and reducing variance. Empirically, we observe double-digit percent reductions in test error for small-sample regression tasks, demonstrating the utility of our approach across a variety of dataset types.

Our work makes the following key contributions:

(1) New Data Augmentation Framework. We introduce a model-agnostic strategy for augmenting tabular regression data, centered on counterfactual reasoning and residual invariance.

(2) Residual Invariance Principle. We formalize how certain features can be perturbed without corrupting the noise structure, providing insights to guide feature selection.

(3) Empirical Validation. We evaluate CRDA on synthetic benchmarks and real-world datasets from standard repositories (e.g., UCI, PMLB), illustrating consistent improvements across neural and ensemble models.

2. Related Work

General Data Augmentation. Data augmentation refers to the strategy of enlarging or diversifying a training set via synthetic transformations. While central to success in computer vision and NLP (Zhang et al., 2018; Yun et al., 2019; Cubuk et al., 2019), these techniques often rely on *label-preserving* symmetries (e.g. image rotations) or domain-specific invariances (e.g. back-translation in text). However, applying these methods to tabular regression remains non-trivial.

Tabular and Regression-Specific Augmentation. Classical oversampling approaches include SMOTE (Chawla et al., 2002) and its regression extensions (Branco et al., 2017), which interpolate between samples but do not necessarily preserve higher-order feature interactions or heteroskedastic noise. Recent advances seek to formalize regression augmentation through geometric properties. For example, RegMix (Hwang & Whang, 2021) optimizes Mixup policies to generate samples within high-density regions of the data manifold, aiming to preserve the underlying structure. Conversely, C-Mixup (Yao et al., 2022) addresses the risk of manifold intrusion by restricting mixing to sample pairs with high label similarity. Closely related is Anchor Data Augmentation (ADA) (Schneider et al., 2023), which extends Anchor Regression to augmentation. ADA identifies "anchors" (by clustering) and generates samples using a first-order Taylor approximation, effectively assuming local linearity within clusters. While these methods enforce geometric regularity (linearity or manifold density), they can struggle in highly non-linear or sparse regimes where local linearity assumptions fail. CRDA aims to avoid this by enforcing *statistical* regularity (residual invariance) instead.

Deep Generative Models. Deep generative models offer an alternative by learning the joint distribution to sample entirely new rows. Approaches like CTGAN (Xu et al., 2019), TVAE (Xu et al., 2019), and TabDDPM (Kotelnikov et al., 2023) have shown promise in privacy-preserving data synthesis. However, these models typically treat the target variable as just another column, failing to preserve an instance's specific residual noise. This often leads to "realistic" looking samples that degrade predictive performance.

Residual Bootstrapping. In statistical literature, residuals have been leveraged extensively for *uncertainty quantification* rather than data augmentation. For example, the residual bootstrap (Efron, 1992) and conformal prediction methods (Barber et al., 2021) resample or reuse residuals to construct confidence intervals. Our work repurposes this mechanism for augmentation.

Causal and Counterfactual Data Augmentation. Data augmentation typically ignores the generative process behind the data, risking unrealistic synthetic examples. Causal-based approaches (Kocaoglu et al., 2018; Arjovsky et al., 2019) propose integrating structural assumptions so that augmentations preserve invariant relationships across environments. This has been explored in computer vision through interventions on object attributes, and in language by editing tokens in a `do`-intervention style. Closely related at the framing level, concurrent work by Akbar et al. (2025) formalizes outcome-invariant augmentation as a soft intervention on the treatment-generating mechanism, requiring that the augmentation preserves the outcome function itself ($f(gx) = f(x)$). CRDA takes a complementary route where instead of assuming a known symmetry of the predictor, it preserves the *instance-specific noise term* under perturbations of selected features. This noise-preservation principle draws a direct parallel to work in reinforcement learning. Lu et al. (2020) showed that next-state samples remain identifiable under mild assumptions (monotonicity and independence in the noise term). Their *Theorem 1* establishes that, once the observed outcome fixes a particular noise quantile, reusing that noise in a "what-if" scenario yields a valid counterfactual next-state. Similarly, CRDA treats the calculated residual as an exogenous noise variable that is assumed to be independent of the features being perturbed. This allows us to produce new data in a more systematic and grounded way than purely generative or interpolation-based techniques.

3. Background

In this section, we provide an overview of the key concepts that motivate our proposed approach.

3.1. Counterfactual Reasoning and Structural Causal Models

A *structural causal model* (SCM) (Pearl, 2009; Peters et al., 2017) formalizes how observed variables are generated by underlying data-generating processes (DGP). Formally, an SCM is specified by a tuple $\big(\mathcal{X}, \mathcal{Z}, F, P_{\mathcal{Z}}\big)$, where:

- $\mathcal{X} = \{X_1, \ldots, X_m\}$ is the set of endogenous (observed) variables,
- $\mathcal{Z} = \{Z_1, \ldots, Z_m\}$ is the set of exogenous (noise) variables with distribution $P_{\mathcal{Z}}$,
- $F = \{f_i\}_{i=1}^m$ is a collection of structural equations, each of the form

$$x_i \leftarrow f_i\big(pa_i, z_i\big)$$

where $\mathrm{Pa}_i \subseteq \mathcal{X} \setminus \{X_i\}$ denotes the parents of X_i.

An SCM induces a graph, which encodes causal relationships (i.e. who influences whom), and the exogenous noise terms capture stochasticity.

Interventions and causal effects. A central notion in causal inference is that of an *intervention*, written $\mathrm{do}(\cdot)$ (Pearl, 2009). By applying $\mathrm{do}(X = x')$, one replaces the original structural equation $X \leftarrow f_X(\mathrm{Pa}, Z)$ with a constant assignment $X \leftarrow x'$. This operation severs incoming edges to X, thus altering the downstream (child) variables but leaving other aspects of the system intact. Interventions enable us to reason about population-level outcomes under a forced assignment.

Counterfactuals. While interventions ask about population-level effects, counterfactual reasoning answers instance-level "what if" questions: *given this specific outcome I observed, what would have happened if some feature had been different?* (Pearl, 2009; Peters et al., 2017). Concretely, one first infers the actual setting of $Z = z$, then imagines how the outcome Y would change under a hypothetical intervention $\mathrm{do}(X = x')$. This process involves:

(i) **abduction**, where we infer $\mathbf{z}$ from the observed data,

(ii) **action**, where we intervene and reassign X,

(iii) **prediction**, where we propagate z through the modified system to obtain the counterfactual outcome distribution $P(Y'|X = x', Z = z)$.

3.2. Main Assumptions and Theory

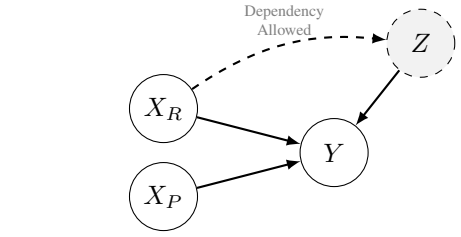
Throughout the paper, we assume that the data-generating process follows an **additive-noise structural causal model (SCM)**, which decomposes the outcome Y into a systematic component and a residual component. The systematic component is the conditional expectation estimated by a base regressor g:

$$Y = g(X_P, X_R) + Z \tag{1}$$

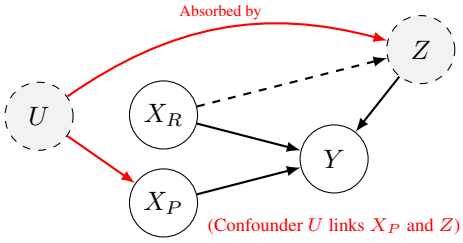
The core theoretical principle underpinning CRDA is the assumption of *residual invariance*. This principle posits that for the true conditional-expectation regressor, the residual noise term remains distributionally constant under interventions on a specific subset of features. We formalize this as follows.

Assumption 3.1. Let the feature vector X be partitioned into two disjoint subsets, $X = (X_P, X_R)$, where X_P are the features we intend to perturb (the *perturbable* coordinates) and X_R are the features we hold fixed (the *remaining* coordinates). Let $g(X) = \mathbb{E}[Y|X]$ be the true conditional expectation function, and let $Z = Y - g(X)$ be the corresponding structural noise term. Residual invariance requires that the noise Z is conditionally independent of the perturbable features X_P given the fixed features X_R:

$$\mathbb{P}(Z \mid X_P, X_R) = \mathbb{P}(Z \mid X_R) \tag{2}$$



(a) Assumption Satisfied



(b) Assumption Violated

Figure 1. **Causal Visualization of Residual Invariance. (a)** The structure satisfies Assumption 3.1. **(b)** A violation due to unobserved confounding. Here, a latent variable U causes both X_P and Y. Since the residual Z absorbs the variation of U, a dependency is created between X_P and Z, invalidating the augmentation.

Causal Interpretation and Hidden Confounding. To better understand what underlying DGPs meet Assumption 3.1, Figure 1a visualizes a causal structure that satisfies it. The assumption requires that the features selected for perturbation, X_P, are *exogenous* with respect to the residual mechanism Z. Note that Z is allowed to depend on the fixed features X_R (e.g., heteroscedasticity or confounding on X_R), as long as it remains independent of X_P. Appendix G

shows that the causal (d-separation) analogue of Assumption 3.1 implies the following: if Z is the only latent direct cause of Y, a perturbed feature $X \in X_P$ is either itself a direct cause of Y or is independent of (d-separated from) Y given the observed direct causes. Thus Assumption 3.1 ensures that the only perturbed features used to predict Y are direct causes of Y (cf. Figure 1a).

Recent work highlights that unobserved confounding is a primary driver of distribution shift failures in real-world tabular data (Prashant et al., 2025; Reddy et al., 2026). When a latent confounder U exists, its influence on Y that is not explained by X is absorbed into the residual term Z. Consequently, Z acts as a noisy proxy for U. Figure 1b illustrates the case where U causes both X_P and Y, thereby creating a "backdoor path" $X_P \leftarrow U \rightarrow Y$. Here, a statistical dependence arises between X_P and Z, violating Assumption 3.1. Therefore, the validity of counterfactual augmentation depends on identifying and excluding such confounded features from the perturbation set.

3.3. From Population to Empirical Residuals

The discussion so far has been at the population level. We now consider what happens when CRDA operates on finite-sample estimates of g and Z. Assumption 3.1 is stated in terms of the true conditional expectation $g(X) = \mathbb{E}[Y \mid X]$ and the true structural noise $Z = Y - g(X)$. In practice, CRDA does not have access to g; it uses a finite-data estimate $\hat{g}$ and computes *empirical* residuals $\hat{Z} = Y - \hat{g}(X)$. The relationship between empirical and true residuals is

$$\hat{Z} \;=\; Z + \big(g(X) - \hat{g}(X)\big) \;=\; Z + e(X_P, X_R),$$

where $e(X_P, X_R)$ is the pointwise approximation error of the base model. The validity of CRDA on empirical residuals depends on how this error behaves on the perturbable coordinates.

Well-specified regime. When $\hat{g}$ is sufficiently accurate, $e \to 0$ uniformly and $\hat{Z} \to Z$. Assumption 3.1 transfers to the empirical residuals: $\hat{Z} \perp X_P \mid X_R$.

Misspecified regime. When $\hat{g}$ fails to capture the systematic effect of X_P, either because the base model underfits or because X_P has a complex non-linear contribution that requires more data than is available, e retains systematic dependence on X_P even after conditioning on X_R. Then $\hat{Z} \not\perp X_P \mid X_R$, and Assumption 3.1 fails on the empirical residuals.

This split governs when CRDA's residual-reuse construction is justified at the empirical level; the practical filtering and validation mechanisms that target the misspecified regime are introduced in Section 4.

4. Method

In this section, we detail our proposed *Counterfactual Residual Data Augmentation (CRDA)* procedure for tabular regression. The main goal is to augment a limited dataset by synthesizing new samples that remain true to the original noise distribution. Algorithm 1 outlines the full workflow. For a visual depiction of the pipeline and how the residual and systematic components interact, see Figure 4 in Appendix A.

4.1. Algorithmic Overview

Algorithm 1 Counterfactual Residual Data Augmentation (CRDA)

Require: Dataset $\mathcal{D}$, baseline regressor $\hat{g}(\cdot)$.
Hyperparameters: PERTURBATIONRANGE (p), AUGDATASIZEFACTOR (M).

Split $\mathcal{D}$ into $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$.
Train baseline $\hat{g}(\cdot)$ on $\mathcal{D}_{\text{train}}$.
for each $(\mathbf{x}_i, y_i)$ in $\mathcal{D}_{\text{train}}$ **do**
  $\hat{z}_i \leftarrow y_i - \hat{g}(\mathbf{x}_i)$ ▷ Compute residual
Select partition (X_P, X_R):
- PC algorithm to remove features directly connected to $\hat{Z}$.
- Correlation check to remove features strongly associated with $\hat{Z}$.

if $X_P = \emptyset$ **then return** g ▷ No partition found
$\mathcal{D}_{\text{aug}} \leftarrow \emptyset$
for each $(\mathbf{x}_i, y_i) \in \mathcal{D}_{\text{train}}$ **do**
  for $m = 1$ to M **do**
    $\mathbf{x}'_{i,P} \leftarrow \mathbf{x}_{i,P} \cdot (1 + \text{Unif}[-p, p])$
    $\mathbf{x}'_i \leftarrow (\mathbf{x}'_{i,P}, \mathbf{x}_{i,R})$ ▷ Keep X_R fixed
    $y'_i \leftarrow \hat{g}(\mathbf{x}'_i) + \hat{z}_i$ ▷ Preserve residual
    $\mathcal{D}_{\text{aug}} \leftarrow \mathcal{D}_{\text{aug}} \cup \{(\mathbf{x}'_i, y'_i)\}$
Validation: Perform K-fold CV comparing *unaugmented* vs. *augmented* models.
Collect validation errors $\{e^{(k)}_{\text{unaug}}, e^{(k)}_{\text{aug}}\}_{k=1}^{K}$.
p-value $\leftarrow$ WilcoxonSignedRank$\big(\{e^{(k)}_{\text{unaug}}, e^{(k)}_{\text{aug}}\}\big)$
if p-value $< \alpha$ **then**
  Train new regressor $\hat{g}'$ on $\mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{aug}}$
  return $\hat{g}'$
else
  return $\hat{g}$ ▷ Augmentation rejected

Step 1: Data Splitting and Baseline Training. We begin by splitting the original dataset $\mathcal{D}$ into training and test sets. Let $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ and $\mathcal{D}_{\text{test}} = \{(\mathbf{x}_j, y_j)\}_{j=N+1}^{n}$. We then fit a *baseline* regressor $\hat{g}(\cdot)$ on $\mathcal{D}_{\text{train}}$. In practice, this model can be chosen from a variety of families (e.g. MLP, XGBoost) depending on the user's preference.

Step 2: Residual Computation. For each training sample

$(\mathbf{x}_i, y_i)$, we compute the *residual*

$$\hat{z}_i \;=\; y_i \;-\; \hat{g}(\mathbf{x}_i).$$

Intuitively, $\hat{z}_i$ captures the latent factors not explained by $\hat{g}$.

Step 3: Feature Selection. To satisfy Assumption 3.1, we must identify which features belong to X_P (perturbable) and which must remain in X_R (fixed). Since the true causal graph is rarely known, we sequentially apply two filters to screen for (approximate) conditional independence between features and the residual $\hat{z}$:

1. A **causal graph check** applies the Peter-Clark (PC) algorithm (Spirtes et al., 2000) to the joint set of variables $\{X, Y, \hat{Z}\}$. Features that have a direct edge to $\hat{Z}$ in the discovered graph are flagged as dependent and assigned to X_R. The remaining features are still potential candidates for X_P.
2. A **Pearson correlation check** assigns features strongly correlated with $\hat{Z}$ (above a threshold) to X_R.

The surviving coordinates form X_P; the complement forms X_R, satisfying Assumption 3.1. (If none survive, we skip augmentation and return the baseline $\hat{g}$.)

Step 4: Counterfactual Generation. For each training sample $(\mathbf{x}_i, y_i)$, we generate M synthetic samples (controlled by the hyperparameter AUGDATASIZEFACTOR). For each synthetic sample $m \in \{1, \ldots, M\}$:

1. We perturb the features in X_P using a uniform scaling factor $\delta \sim \text{Unif}[-p, p]$, where $p \in (0, 1)$ is the PERTURBATIONRANGE:
$$\mathbf{x}'_{i,P} = \mathbf{x}_{i,P} \cdot (1 + \delta).$$
2. We construct the new feature vector $\mathbf{x}'_i = (\mathbf{x}'_{i,P}, \mathbf{x}_{i,R})$, keeping X_R fixed.
3. We calculate the counterfactual label by passing the *new* input through the baseline model and adding the *old* residual:
$$y'_i = \hat{g}(\mathbf{x}'_i) + \hat{z}_i.$$

Crucially, this preserves the original residual $\hat{z}_i$, thus keeping the overall noise structure intact under the perturbation. The newly generated samples $\left(\mathbf{x}'_i,\, y'_i\right)$ form an augmented set $\mathcal{D}_{\text{aug}}$.

Step 5: Safety Validation. Before committing to a final retraining, we evaluate whether the augmented samples *significantly* improve generalization. Concretely, we run K-fold cross-validation on the *original* training data, comparing two models:

1. **Unaugmented model**: trained on the fold's training portion as is.
2. **Augmented model**: trained on the fold's training portion *plus* its augmented points (generated using the same procedure above).

We collect the validation errors (e.g. MSE) across the K folds for both models and perform a nonparametric *Wilcoxon signed-rank test* (Wilcoxon, 1945) on the paired errors. If the resulting p-value is below a chosen significance level (e.g. $\alpha = 0.05$), we conclude that augmentation yields a statistically significant improvement; otherwise, we revert to the baseline $\hat{g}$.

Step 6: Combined Dataset and Retraining. If the Wilcoxon test finds a significant improvement, we combine $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{aug}}$ into a single augmented dataset

$$\mathcal{D}^{\text{aug}}_{\text{train}} \;=\; \mathcal{D}_{\text{train}} \;\cup\; \mathcal{D}_{\text{aug}}.$$

We then retrain a *final* model $\hat{g}'(\cdot)$ with this expanded dataset to use on the test set.

4.2. Discussion of Key Design Choices

Two Views of CRDA: Statistical and Counterfactual.

Statistical (non-causal) view. For soundness, CRDA requires only Assumption 3.1: $Z \perp X_P \mid X_R$. Given an estimated regressor $\hat{g}$, empirical residuals $\hat{z}_i = y_i - \hat{g}(x_i)$, a screened subset X_P, and a perturbed input $x'_i = (x'_{i,P}, x_{i,R})$, the augmented label $y'_i = \hat{g}(x'_i) + \hat{z}_i$ is a sample whose conditional noise distribution matches the original's by Assumption 3.1. No causal claim is required to motivate the procedure.

Counterfactual view (analogy under additive noise). When the decomposition $Y = g(X_P, X_R) + Z$ acts as an additive-noise SCM and X_P is exogenous with respect to Z, reusing $\hat{z}_i$ while changing $x_{i,P}$ to $x'_{i,P}$ mirrors Pearl (2009)'s abduction–action–prediction template (Section 3.1): abduct $\hat{z}_i$ from the observed sample, act on X_P, predict with the same noise. In Appendix G we establish a precise relationship between CRDA and counterfactuals in Pearl's sense. Assuming that the true causal graph satisfies the d-separation analogue of Assumption 3.1 and that the noise term Z is the only unobserved parent of Y, CRDA estimates the counterfactual probability $P(Y_{X=x'} \mid X = x, Y = y)$ at the population level. The finite-sample caveats of Section 3.3 continue to apply.

Reusing the Residual. Setting $y'_i = \hat{g}(x'_i)$ alone would implicitly fix the noise to zero for every synthetic sample, teaching the model that the data manifold is deterministic and erasing the natural heteroscedasticity of the target. Reusing $\hat{z}_i$ preserves the instance-specific noise structure and distinguishes CRDA from naive feature perturbation.

Residual Invariance Filters. The practical steps in Algorithm 1, such as the PC algorithm and correlation checks, are *empirical screens* designed to identify a feature subset X_P for which Assumption 3.1 is likely to hold. The choice of the filters themselves is a design decision and can be swapped for similar techniques. In Appendix H.1, we empirically validate our filters by showing that rejected features possess significantly higher Mutual Information with the residuals than selected features. When we reuse a specific residual $\hat{z}_i$ to construct a counterfactual label $y' = \hat{g}(x') + \hat{z}_i$, we make a practical choice that avoids the need to explicitly model the entire noise distribution.

Model-Agnostic Nature. Although our algorithm is illustrated with a neural or tree-based baseline, the same counterfactual logic applies to *any* parametric or nonparametric regressor: each training outcome is viewed as a sum of a learned systematic component and a noise term, and perturbations occur in the input space while the residual stays anchored to its original data point. While CRDA can be applied to any regression model, it does not *guarantee* improvement on every such model. To guard against cases where augmentation would not help, CRDA defaults to the untouched baseline whenever its safeguards trigger.

5. Experiments

5.1. Experimental Setup and Protocol

High-Level Goal. Our primary goal is to investigate whether CRDA can reduce test MSE on tabular regression tasks, particularly under conditions of data scarcity. We evaluate performance across nine benchmark datasets and a synthetic task with a known ground-truth DGP.

Comparisons and Baselines.

1. **No Augmentation (Baseline)**: Train the model on the raw data only.
2. **CRDA Augmentation**: Train the same model class on the union of the raw data and the counterfactual samples generated by CRDA.
3. **Generative Model Augmentation**: The baseline model architecture trained on the union of the original data and synthetic samples produced by state-of-the-art tabular data generators

Hyperparameter Search. We tune *MLPRegressor* and *XGBoostRegressor* with scikit-learn's `RandomizedSearchCV` (Pedregosa et al., 2011) (3-fold) for 20 trials each, totaling *60 fits* per baseline. *CRDA* also introduces three additional knobs: AUGDATASIZEFACTOR, PERTURBATIONRANGE and MAXNUMFEATURESTOPERTURB. We tune these via *Optuna* (TPE sampler) (Akiba et al., 2019) for up to *30* trials, similarly minimizing validation error [1].

Datasets and Preprocessing. We consider nine regression datasets from the University of California, Irvine (UCI) Machine Learning Repository, the Penn Machine Learning Benchmarks (PMLB) collection, and Kaggle [2]. These were chosen to represent a variety of numeric, tabular domains and sample sizes. We drop duplicate rows and NaN values, then apply standardization per feature. For each dataset, we produce five training subsets, ranging from $n/5$ up to n. All subsets are split 80–20 into training and test sets.

Evaluation Metrics and Significance Tests. We report the **MSE** (mean-squared error on the held-out test set) for our settings and their relative change (negative values indicate improvement).

$$\mathbf{\Delta}\% = 100\,(\mathrm{MSE}_{\mathrm{CRDA}} - \mathrm{MSE}_{\mathrm{baseline}})/\mathrm{MSE}_{\mathrm{baseline}}$$

Significance is assessed with a Wilcoxon signed-rank test across 10 CV folds (Wilcoxon, 1945).

5.2. Results and Analysis

Our main findings are presented in Table 1 and summarized in Figure 2. We see that CRDA provides consistent and substantial reductions in test MSE across nearly all datasets and training set sizes. As shown in Figure 2, the **MLP Regressor benefits most significantly**, achieving an average MSE reduction of **22.9%** across all nine datasets. This highlights CRDA's ability to provide valuable signal for data-hungry neural models. For instance, on the *Parkinson's Monitoring* and *House Price* datasets, MLP models augmented with CRDA see their error reduced by over 30%. The **XGB Regressor** also shows consistent improvement, with an average MSE reduction of **6.4%**.

Table 1 details the performance at different data scales, averaged on 15 seeds [3]. Green cells indicate cases in which the Wilcoxon signed-rank test would have approved augmentation, and red cells indicate cases where it would have abstained. This built-in safeguard prevents CRDA from being applied where it might not be beneficial. While infrequent, we do note that this filter can be prone to error, particularly depending on sample size. Per-cell significance heat-maps are provided in Appendix I.

We further validate the necessity of our specific design choices in Appendix H.2, showing that CRDA outperforms naive global perturbation and label-invariant baselines.

[1] Complete search spaces and selected hyperparameters for the base regressors, together with the configurations used for all augmentation baselines, are detailed in Appendix E.

[2] Summary statistics and sources of datasets are in Appendix C.

[3] Full per-seed results with standard errors are in Appendix L

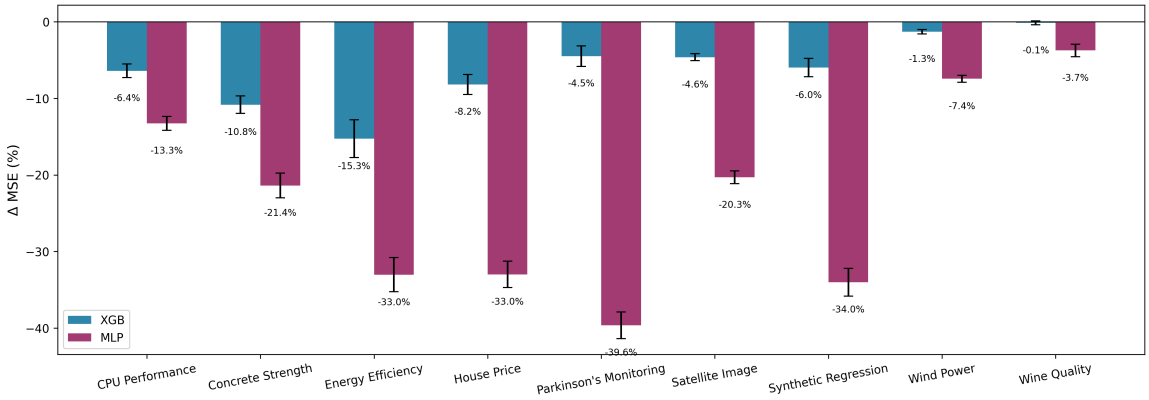


Figure 2. MSE percentage change for each dataset averaged over the five different training-subset sizes reported in Table 1 with error bars corresponding to standard error. Lower is better.

Table 1. Average change in MSE after applying CRDA for XGB and MLP across 15 seeds. Cells are green when data augmentation was selected to proceed more often based on the Wilcoxon signed rank test and red otherwise. The reported Δ values are computed with augmentation applied in all cases, independent of this decision. Lower is better ↓.

Dataset	Sample Size	XGB MSE Δ %	MLP MSE Δ %
CPU Performance (Olson et al., 2017a)	1638	-6.99	-20.24
	3276	-9.47	-14.03
	4914	-6.20	-11.31
	6552	-4.13	-10.48
	8190	-5.19	-10.23
Satellite Image (Olson et al., 2017b)	1287	-4.54	-18.36
	2574	-3.73	-16.69
	3861	-4.79	-23.14
	5148	-4.73	-23.72
	6435	-5.31	-19.66
Wind Power (Haslett & Raftery, 1989)	1314	-2.82	-7.22
	2628	0.20	-9.17
	3942	-1.33	-9.03
	5256	-1.40	-6.15
	6570	-1.08	-5.56
Synthetic Regression (Olson et al., 2017c)	200	-12.00	-28.80
	400	-3.16	-36.93
	600	-7.94	-27.91
	800	-2.23	-34.12
	1000	-4.59	-42.33
Concrete Strength (Yeh, 1998)	201	-8.01	-17.80
	402	-8.43	-19.83
	603	-9.75	-17.64
	804	-15.72	-24.77
	1005	-12.19	-26.90
Energy Efficiency (Tsanas & Xifara, 2012)	153	-13.33	-25.10
	306	-12.20	-28.13
	459	-10.55	-42.98
	612	-19.35	-40.71
	765	-20.96	-28.31
House Price (Montoya & DataCanary, 2016)	200	-14.23	-40.57
	400	-5.39	-37.02
	600	-4.87	-30.14
	800	-9.86	-30.32
	1000	-6.50	-26.97
Parkinson's Monitoring (Tsanas & Little, 2009)	1175	-8.40	-36.17
	2350	-6.60	-31.82
	3525	-2.79	-36.60
	4700	-6.26	-46.40
	5875	1.65	-47.23
Wine Quality (Cortez et al., 2009)	1063	0.31	-0.34
	2126	1.01	-5.24
	3189	-0.33	-3.63
	4252	-0.61	-4.44
	5315	-1.08	-4.99

To better understand how CRDA's effectiveness scales with sample size in a controlled setting, we conducted an experiment on a synthetic DGP with a known ground-truth independence structure:

$$Y = X_1^2 + X_2 X_3 + Z, \qquad \text{where } Z \perp (X_1, X_2, X_3)$$

We generated 50,000 samples and applied CRDA at various sample sizes. The results, shown in Figure 3, reveal that at very low sample sizes (<2.5k), CRDA offers minimal benefit because the base predictor is too inaccurate to produce meaningful residuals. Conversely, at very high sample sizes (>30k), the baseline model is already so accurate that there is little room for improvement. The *greatest MSE reduction occurs in a "sweet spot"* (between 2.5k and 20k samples), where the baseline model has learned the main signal but still benefits from the localized exploration of the feature space that CRDA provides. This dependency on model capacity is further validated in Appendix J using *CatBoost* (Prokhorenkova et al., 2018). Being more data-efficient, CatBoost's utility peaks at even lower sample sizes (peaking around $N \approx 500$ for datasets like *Parkinson's* and *Energy Efficiency*), confirming that the optimal CRDA window shifts earlier as the base learner becomes stronger.

Finally, we benchmarked CRDA against a comprehensive suite of baselines, including regression augmentation methods (C-Mixup (Yao et al., 2022), ADA (Schneider et al., 2023)) and deep generative models (TabDDPM (Kotelnikov et al., 2023), TVAE (Xu et al., 2019), CTGAN (Xu et al., 2019)). As shown in Table 2, CRDA demonstrates superior stability and performance. For a fair comparison, CRDA's safety gate is disabled so that augmentation is applied unconditionally, exactly as the competing methods are; its advantage therefore comes from the augmentation mechanism itself. While geometric methods like ADA and C-Mixup provide gains in specific settings, they exhibit catastrophic failure modes in others (e.g., increasing MSE by over 100% on *Synthetic Regression* and *Parkinson's*). Similarly, deep

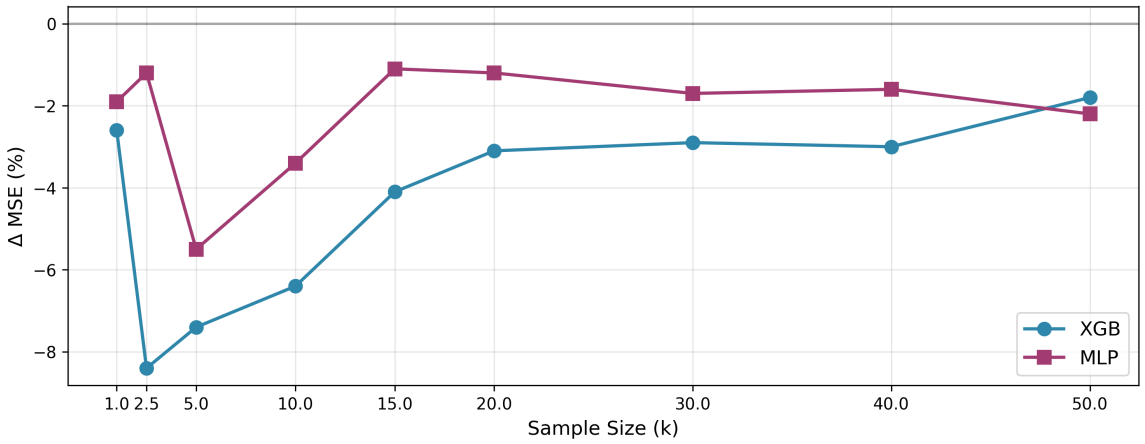


Figure 3. Synthetic sample-size-scaling experiment with a DGP of known independence for the residuals Z and features X. For both XGB and MLP base models, we observe a "sweet spot" where CRDA yields the largest MSE reduction (typically at a lower sample size).

Table 2. The percent MSE change for XGB and MLP base regressors. We compare CRDA against specialized regression augmentations (C-Mixup (Yao et al., 2022), ADA (Schneider et al., 2023)) and generative models (TabDDPM (Kotelnikov et al., 2023), TVAE (Xu et al., 2019), CTGAN (Xu et al., 2019)). Averaged across 10 seeds, reporting standard error. Lower is better ↓.

Dataset	**Model**	**% MSE Change ↓**					
		$\Delta_{\text{C-Mixup}}$	Δ_{ADA}	Δ_{TabDDPM}	Δ_{TVAE}	Δ_{CTGAN}	Δ_{CRDA}
CPU Performance (Olson et al., 2017a)	XGB	1.7 ± 1.5	1.9 ± 0.6	36.5 ± 4.0	24.2 ± 2.6	41.6 ± 2.8	**-1.4 ± 0.7**
	MLP	-0.9 ± 1.0	-0.6 ± 1.0	30.2 ± 7.1	36.6 ± 4.6	108.0 ± 13.3	**-12.2 ± 1.0**
Satellite Image (Olson et al., 2017b)	XGB	6.4 ± 1.7	1.4 ± 1.1	10.7 ± 1.4	12.5 ± 2.1	11.0 ± 1.1	**-0.7 ± 0.7**
	MLP	-0.8 ± 2.0	3.3 ± 2.4	9.9 ± 3.0	18.8 ± 3.3	55.1 ± 3.2	**-23.5 ± 1.5**
Wind Power (Haslett & Raftery, 1989)	XGB	-1.9 ± 0.6	-0.2 ± 0.3	-0.4 ± 0.5	4.8 ± 0.9	7.1 ± 0.9	**-2.6 ± 0.3**
	MLP	4.9 ± 3.4	14.7 ± 2.0	-7.1 ± 1.4	5.5 ± 1.9	13.7 ± 1.9	**-8.0 ± 1.1**
Synthetic Regression (Olson et al., 2017c)	XGB	141.5 ± 31.8	18.3 ± 3.6	25.0 ± 8.2	116.5 ± 20.2	164.1 ± 25.8	**2.3 ± 1.8**
	MLP	78.1 ± 19.2	16.7 ± 6.4	-19.2 ± 2.7	71.6 ± 15.4	198.9 ± 27.2	**-33.3 ± 3.8**
Concrete Strength (Yeh, 1998)	XGB	**-2.8 ± 1.7**	-0.1 ± 1.4	-1.1 ± 3.0	7.9 ± 2.7	26.1 ± 4.3	-1.7 ± 1.9
	MLP	-4.8 ± 2.9	-3.8 ± 1.5	-5.9 ± 2.9	36.8 ± 8.2	142.3 ± 17.6	**-15.4 ± 2.3**
Energy Efficiency (Tsanas & Xifara, 2012)	XGB	-18.0 ± 9.1	-20.7 ± 3.5	3.4 ± 7.7	-19.4 ± 8.1	**-24.7 ± 6.2**	-10.7 ± 3.8
	MLP	11.9 ± 14.2	-22.9 ± 4.1	11.1 ± 11.1	127.7 ± 33.6	360.2 ± 51.5	**-32.5 ± 5.4**
House Price (Montoya & DataCanary, 2016)	XGB	-12.8 ± 12.6	**-42.9 ± 7.4**	-13.4 ± 12.9	298.6 ± 94.8	824.1 ± 139.6	-12.8 ± 3.6
	MLP	-51.0 ± 5.1	**-52.6 ± 5.8**	-4.3 ± 16.1	992.5 ± 263.7	3032.9 ± 374.3	-42.3 ± 3.4
Parkinson's Monitoring (Tsanas & Little, 2009)	XGB	105.1 ± 13.2	89.5 ± 11.1	286.7 ± 25.9	405.1 ± 53.2	565.3 ± 49.3	**-0.3 ± 1.9**
	MLP	116.1 ± 37.5	28.7 ± 15.1	184.9 ± 36.3	659.0 ± 109.0	1454.2 ± 221.8	**-48.2 ± 6.0**
Wine Quality (Cortez et al., 2009)	XGB	-2.0 ± 0.5	-0.0 ± 0.6	-2.6 ± 0.7	-0.5 ± 0.6	0.3 ± 0.8	**-2.8 ± 0.4**
	MLP	13.1 ± 5.2	23.6 ± 2.5	**-5.5 ± 0.6**	-0.8 ± 0.9	2.1 ± 0.6	-2.9 ± 0.8

generative models significantly degrade performance more often, likely due to difficulties in capturing the precise conditional distribution $P(Y|X)$ required for regression. In contrast, CRDA's residual-preserving mechanism ensures that synthetic samples remain faithful to the underlying noise structure. Across all datasets, CRDA is the only method that reliably improves performance for both XGBoost and MLP models without the risk of significant degradation.

6. Limitations

CRDA is currently designed for regression tasks; extending its principles to classification, where residuals are not straightforwardly defined, is a direction for future work.

The core of our method's validity hinges on a key assumption: the model's residual noise, Z, is conditionally independent of the features we choose to perturb, X_P, given the features we hold fixed, X_R (Assumption 3.1). In practice, verifying this assumption from finite data is a primary challenge. A poorly fitted base predictor can produce residuals

that retain dependencies on all input features, causing interventions on X to break the required noise invariance. To address this, CRDA employs a two-layer screen.

First, we use the PC algorithm and a Pearson correlation test as a *risk-control heuristic* to filter for candidate features that are likely to satisfy Assumption 3.1. We acknowledge this screen is imperfect; the PC algorithm can fail in the presence of unobserved confounders and scale poorly with high feature counts, while correlation tests may not detect non-linear dependencies. However, these filters are not intended to be infallible but rather a practical first line of defense. The theoretical guarantees of the PC algorithm, for instance, are well-studied; under standard assumptions, its error probability of incorrectly identifying an edge decays exponentially with sample size (Kalisch & Bühlmann, 2007). This sample consistency suggests that the risk of our filter admitting a feature that violates Assumption 3.1 diminishes as more data becomes available.

Moreover, CRDA incorporates a second, decisive *safety gate*: the Wilcoxon signed-rank test. This evaluates the realized impact of augmentation on a validation set. If the generated samples do not yield a statistically significant improvement, we return the original baseline. This mechanism ensures that we either improve the baseline or abstain from augmentation, thereby mitigating the risk of performance degradation from an imperfect initial screen.

To address concerns that a strong base learner is required, we include *linear regression* experiments in Appendix K. In every dataset/fold, the Wilcoxon gate produced $p > 0.05$, so CRDA *abstained*. If one *ignores* the gate and forces augmentation, performance generally degrades or does not improve, illustrating that the safety checks are beneficial and block harmful augmentation for weak baselines.

Finally, the performance of CRDA is sensitive to both dataset size and the choice of the base predictor. For very large datasets, the need for augmentation diminishes, and CRDA offers little benefit. Conversely, if a dataset is too small, the base model may be too weak to produce meaningful residuals that are even approximately independent of the features, and our statistical tests will lack power. This can be seen in our sample-size-scaling experiment in Figure 3.

7. Conclusion

We described a new data augmentation technique for regression called CRDA. CRDA is model-agnostic and it does not assume any domain knowledge such as specific transformations that preserve labels. Instead, it leverages counterfactual reasoning and the invariance of the residual noise distribution. We demonstrated the effectiveness of CRDA in data scarce regression tasks where it helped improve predictions made by representative base predictors including XGBoost and multi-layer perceptrons. We also displayed substantially stronger and more reliable results when compared to state-of-the-art tabular data generators.

Several directions for future work remain. The first is to extend CRDA to classification tasks. The key challenge is due to the non-numeric nature of residuals, though embedding-based transformations offer a potential path. The second would be to explore alternative methods in our feature partitioning step, such as formal proxy-based causal inference techniques or confounder-robust causal discovery algorithms like Fast Causal Inference (FCI) (Spirtes et al., 2000). This may enable CRDA to better adjust for hidden factors rather than simply discarding confounded features.

To support adoption on real tabular datasets, which often contain categorical columns outside the scope of the multiplicative perturbation analyzed here, we additionally release a reference implementation as a Python package. It is available via `pip install crda` (https://pypi.org/project/crda/) and extends the paper's method by supporting uniform category resampling for one-hot-encoded categorical features. We also replace our feature independence tests with distance-correlation (Székely et al., 2007) for continuous features and mutual information for categorical features. This variant still shares all other logic including the same residual-reuse construction and Wilcoxon outcome gate.

Impact Statement

This is foundational work that aims to improve regression in data scarce scenarios, which can span a wide range of important domains. As discussed in Section 6 of the paper, CRDA could worsen predictive accuracy instead of improving it, leading to negative consequences in high impact applications. To mitigate such outcomes, CRDA has filters in place that detect and prevent harm.

Acknowledgment

We acknowledge funding from the Canada CIFAR AI Chair program, discovery grants to Poupart, Li and Schulte from the Natural Sciences and Engineering Research Council of Canada and a grant from IITP & MSIT of Korea (No. RS-2024-00457882, AI Research Hub Project). Computational resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute https://vectorinstitute.ai/about/current-partners/.

References

Akbar, U., Kilbertus, N., Shen, H., Muandet, K., and Dai, B. An analysis of causal effect estimation using outcome invariant data augmentation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=C1LVIInfZO.

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Barber, R. F., Candes, E. J., Ramdas, A., and Tibshirani, R. J. Predictive inference with the jackknife+. *The Annals of Statistics*, 49(1):486–507, 2021.

Branco, P., Torgo, L., and Ribeiro, R. P. SMOGN: a pre-processing approach for imbalanced regression. In *First international workshop on learning with imbalanced domains: Theory and applications*, pp. 36–50. PMLR, 2017.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Chen, T. and Guestrin, C. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., and Reis, J. Wine Quality, 2009. URL https://archive.ics.uci.edu/dataset/186/wine%2Bquality. UCI Machine Learning Repository.

Cubuk, E. D., Zoph, B., Mané, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 113–123, 2019. doi: 10.1109/CVPR.2019.00020.

Efron, B. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics: Methodology and distribution*, pp. 569–593. Springer, 1992.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1):3133–3181, 2014.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Haslett, J. and Raftery, A. E. Irish Wind Speed (Malin Head, 1961–1978), 1989. URL https://www.rdocumentation.org/packages/gstat/topics/wind. Daily average wind speeds at 12 Irish stations.

Hwang, S.-H. and Whang, S. E. RegMix: Data mixing augmentation for regression. *arXiv preprint arXiv:2106.03374*, 2021.

Kalisch, M. and Bühlmann, P. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8(22), 2007.

Kocaoglu, M., Snyder, C., Dimakis, A. G., and Vishwanath, S. CausalGAN: Learning causal implicit generative models with adversarial training. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJE-4xW0W.

Kotelnikov, A., Baranchuk, D., Rubachev, I., and Babenko, A. TabDDPM: Modelling tabular data with diffusion models. In *International conference on machine learning*, pp. 17564–17579. PMLR, 2023.

Kraskov, A., Stögbauer, H., and Grassberger, P. Estimating mutual information. *Physical Review E-Statistical, Nonlinear, and Soft Matter Physics*, 69(6):066138, 2004.

Lu, C., Huang, B., Wang, K., Hernández-Lobato, J. M., Zhang, K., and Schölkopf, B. Sample-efficient reinforcement learning via counterfactual-based data augmentation. *CoRR*, abs/2012.09092, 2020. URL https://arxiv.org/abs/2012.09092.

Montoya, A. and DataCanary. House prices - advanced regression techniques, 2016. URL https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques. Kaggle.

Olson, R. S., Cava, W. L., Orzechowski, P., Urbanowicz, R. J., and Moore, J. H. PMLB Dataset 227_cpu_small, 2017a. URL https://github.com/EpistasisLab/pmlb. Penn Machine Learning Benchmarks, version 2025-05-16.

Olson, R. S., Cava, W. L., Orzechowski, P., Urbanowicz, R. J., and Moore, J. H. PMLB Dataset 294_satellite_image, 2017b. URL https://github.com/EpistasisLab/pmlb. Penn Machine Learning Benchmarks, version 2025-05-16.

Olson, R. S., Cava, W. L., Orzechowski, P., Urbanowicz, R. J., and Moore, J. H. PMLB Dataset 623_fri_c4_1000_10, 2017c. URL https://github

.com/EpistasisLab/pmlb. Synthetic Friedman #4 variant; Penn Machine Learning Benchmarks.

Pearl, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2 edition, 2009.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Peters, J., Janzing, D., and Schölkopf, B. *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press, 2017.

Prashant, P. P., Khatami, S. B., Ribeiro, B., and Salimi, B. Scalable out-of-distribution robustness in the presence of unobserved confounders. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL https://openreview.net/forum?id=eIyOtZ9tgl.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. CatBoost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 2018. URL https://papers.nips.cc/paper/7898-catboost-unbiased-boosting-with-categorical-features.

Reddy, A. G., Rubio-Madrigal, C., Burkholz, R., and Muandet, K. When shift happens - confounding is to blame. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=sFjxg8cyJS.

Schneider, N., Goshtasbpour, S., and Perez-Cruz, F. Anchor data augmentation. *Advances in Neural Information Processing Systems*, 36:74890–74902, 2023.

Spirtes, P., Glymour, C., and Scheines, R. *Causation, Prediction, and Search*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2nd edition, 2000.

Sun, X. and Schulte, O. Cause-effect inference in location-scale noise models: Maximum likelihood vs. independence testing. *Advances in Neural Information Processing Systems*, 36:5447–5483, 2023.

Székely, G. J., Rizzo, M. L., and Bakirov, N. K. Measuring and testing dependence by correlation of distances. *The Annals of Statistics*, 35(6):2769–2794, 2007.

Tsanas, A. and Little, M. A. Parkinsons Telemonitoring, 2009. URL https://archive.ics.uci.edu/ml/datasets/parkinsons%2Btelemonitoring. UCI Machine Learning Repository.

Tsanas, A. and Xifara, A. Energy Efficiency, 2012. URL https://archive.ics.uci.edu/ml/datasets/energy%2Befficiency. UCI Machine Learning Repository.

Wilcoxon, F. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. Modeling tabular data using conditional GAN. *Advances in Neural Information Processing Systems*, 32, 2019.

Yao, H., Wang, Y., Zhang, L., Zou, J. Y., and Finn, C. C-Mixup: Improving generalization in regression. *Advances in Neural Information Processing Systems*, 35: 3361–3376, 2022.

Yeh, I. Concrete Compressive Strength, 1998. URL https://archive.ics.uci.edu/ml/datasets/concrete%2Bcompressive%2Bstrength. UCI Machine Learning Repository.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. CutMix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1Ddp1-Rb.

A. Methodological Diagram

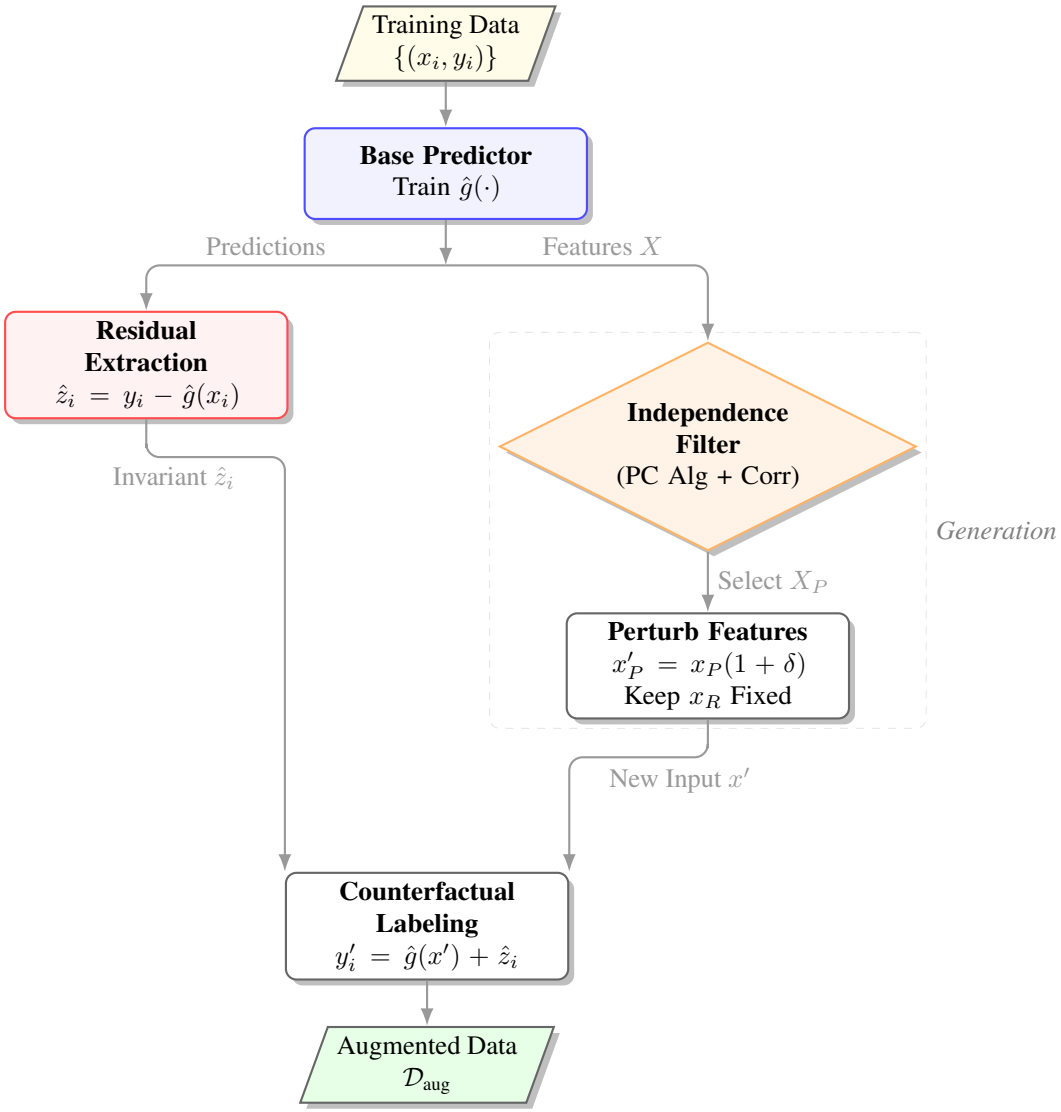


Figure 4. The Counterfactual Residual Data Augmentation (CRDA) Pipeline. The workflow proceeds top-to-bottom; the base model component $\hat{g}(\cdot)$ first isolates the residual noise $\hat{z}_i$. Simultaneously, the Independence Filter identifies safe features x_P. These are perturbed and recombined with the preserved residual to generate valid counterfactual samples.

B. Implementation Details

All experiments were conducted in `Python`, leveraging standard libraries for machine learning and hyperparameter optimization. We used `scikit-learn` for the `MLPRegressor` baseline (Pedregosa et al., 2011), `XGBoost` for the `XGBoostRegressor` baseline (Chen & Guestrin, 2016), and `Optuna` for tuning CRDA's specific hyperparameters (Akiba et al., 2019).

Our experimental protocol uses 10-fold cross-validation (CV), and we seed all random number generators to ensure reproducibility. Computations were performed on a single **AWS c7i.24xlarge** instance equipped with 96 vCPUs and 192 GB of RAM. To facilitate the complete reproduction of our findings, we have made the full study logs, JSON configuration files, and source code available in our project repository, `https://github.com/mhmohebbi/CRDA/`.

C. Dataset Summary

Table 3 lists the nine tabular-regression benchmarks used in the paper, with their total sample count, dimensionality (# numeric columns), and provenance repository.

Table 3. Basic statistics of the evaluation datasets, including number of samples (n_{samp}) and number of features (n_{features}).

Dataset	n_{samp}	n_{features}	Source
CPU Performance	8192	12	PMLB (Olson et al., 2017a)
Satellite Image	6435	36	PMLB (Olson et al., 2017b)
Wind Power	6574	14	UCI (Haslett & Raftery, 1989)
Synthetic Regression	1000	10	PMLB (Olson et al., 2017c)
Concrete Strength	1005	8	UCI (Yeh, 1998)
Energy Efficiency	768	9	UCI (Tsanas & Xifara, 2012)
House Price	1000	7	Kaggle (Montoya & DataCanary, 2016)
Parkinson's Monitoring	5875	20	UCI (Tsanas & Little, 2009)
Wine Quality	5318	11	UCI (Cortez et al., 2009)

D. Complete Experimental Protocol

Configuration Objects. We centralize all experiment settings (e.g. dataset path, model type, global seed, CRDA knobs) in a Python class `Config`. Each run instantiates a `Config` with specific arguments and passes it to our `Experiment` harness, which saves the resulting configuration to a JSON file for reproducibility.

Listing 1. Example truncated config file for an XGB run.

```
{
  "baseline": "xgboost",
  "dataset_path": "../data/WineQuality.csv",
  "sample_sizes": [1063, 2126, 3189, 4252, 5315],
  "ignore_filter": true,
  "hyperparam_tune": true,
  "results_dir": "../experiments/WineQuality",
  "...": "More fields omitted (test_size, num_seeds, p_wilcoxon_threshold, etc.)"
}
```

Key Fields and Usage

- **Model parameters:** `baseline` can be set to "mlp" or "xgboost"; we do not alter other hyper-parameters (those are tuned via `RandomizedSearchCV`).
- **CRDA knobs:** `aug_data_size_factor`, `max_n_features_to_perturb` and `max_perturb_percent`. These are also tunable parameters (via `Optuna`). They specify how many counterfactual samples to generate, how many perturbable features we perturb and by how much; see Section F.
- **Data splits:** `sample_sizes` enumerates partial subsets of a dataset (e.g. $\frac{n}{5}, \ldots, n$), and `test_size` sets the final train–test ratio.
- **Miscellaneous toggles:** `hyperparam_tune` (whether to run a cross-validated search), `ignore_filter` (bypass CRDA's feature independence checks), `save_plots`, etc.

For each experiment, the `Experiment` class reads the `config` object, runs the pipeline (training, augmentation, evaluation), and dumps logs plus final results in a timestamped directory. By reloading `config.json` via `Config.from_dict`, one can exactly reproduce the same run.

E. Hyper-parameters and Search Spaces

E.1. Base Regressor Configurations

The two baseline families–**MLPRegressor** and **XGBoostRegressor**–share a hybrid strategy: we *fix* well-established architectural or optimisation knobs to textbook defaults, while *searching* over the handful of hyper-parameters that most strongly drive bias–variance trade-offs. This mirrors common practice in tabular ML benchmarks (Fernández-Delgado et al., 2014) and keeps the search budget (20 trials per 3-fold, per dataset, per baseline) focused on the levers that matter.

Why these choices? For MLPs we retain the ReLU–Adam recipe that has been shown to be robust for small/medium tabular tasks (Goodfellow et al., 2016). We enable *adaptive* learning-rate and early stopping to guard against over-training, and explore only depth/width ('hidden_layer_sizes') and three learning-dynamics scalars (α, learning_rate_init, tol). For XGBoost we follow the histogram grow policy ("`tree_method=hist`") that is memory-friendly on CPUs, fix 1 000 boosting rounds (with early stopping inside the CRDA loop), and search the usual five knobs that govern tree shape, sampling and shrinkage. Median wall-clock per trial on a `c7i.24xlarge` is ∼9s (MLP) and ∼4s (XGB).[4] Tables 4 and 5 enumerate the *search priors* together with the *modal* best value across datasets.

Table 4. MLPRegressor hyper-parameters searched with RANDOMIZEDSEARCHCV. Ranges use log-uniform (LogU) or categorical priors.

Parameter	Prior / Range	Modal best
hidden_layer_sizes	$\{(128,64,32),(128,64),(64,32),(64,)\}$	(128,64,32)
α (L2)	$\mathrm{LogU}(10^{-5},10^{-3})$	0.00040
learning_rate_init	$\mathrm{LogU}(10^{-3},10^{-2})$	0.00942
tol	$\mathrm{LogU}(10^{-5},10^{-4})$	0.00009
Fixed for all runs		
activation	relu	
solver	adam	
batch_size	32	
max_iter	1 000	
learning_rate	adaptive	
early_stopping	true	
validation_fraction	0.10	
n_iter_no_change	20	

Table 5. XGBoostRegressor hyper-parameters searched with RANDOMIZEDSEARCHCV. Log-spaces are base-10.

Parameter	Prior / Range	Modal best
learning_rate	$\log_{10}[10^{-3},10^{-1}]$ (10 pts)	0.02154
max_depth	$\{3,4,6\}$	6
min_child_weight	$\{1,5\}$	5
subsample	$\{0.7,1.0\}$	0.7
colsample_bytree	$\{0.7,1.0\}$	0.7
reg_lambda	$\log_{10}[10^{-3},10^{1}]$ (6 pts)	0.03981
Fixed for all runs		
objective	reg:squarederror	
tree_method	hist	
n_estimators	1 000	
reg_alpha	0.0	
early_stopping_rounds	20	

[4] Full timing logs are available in `experiments/full_reproduction.ipynb` of the code repository.

E.2. Data Augmentation Baseline Configurations

For comparability and reproducibility of the Table 2 benchmark, we report the full configuration used for each augmentation baseline. In all cases we followed the recommended defaults from the method's original paper and its public reference implementation; no method-specific tuning was performed on our datasets. Table 6 summarizes the values.

Table 6. Hyperparameter configurations for the augmentation baselines used in Table 2. Values follow the recommended defaults of each method's reference implementation.

Method	**Hyperparameter**	**Value**
C-Mixup	alpha	1.0
	KDE bandwidth	1.0
	mixtype	`kde`
	KDE type	`gaussian`
ADA	gammas	$\{0.5, 0.75, 1.5, 2.0\}$
	number of anchors	10
TabDDPM	learning rate	10^{-3}
	batch size	4096
	diffusion timesteps	1000
	training iterations	1000
	MLP layers	2
	MLP layer width	256
	dropout	0.0
CTGAN	batch size	500
	embedding dimension	128
	generator / discriminator dims	$(256, 256)$
	learning rate (G / D)	2×10^{-4}
	weight decay	10^{-6}
	pac	10
TVAE	embedding dimension	128
	encoder dims	$(128, 128)$
	decoder dims	$(128, 128)$
	L2 scale	10^{-5}
	loss factor	2

F. CRDA Knob Selection & Sensitivity

CRDA exposes three *augmentation knobs*. During a 30-trial OPTUNA–TPE search (per dataset, per baseline) we sample from the priors in Table 7; all other implementation details are inherited from Algorithm 1 (main paper).

- **`max_n_features_to_perturb`** controls *how many* invariant features are jointly edited, trading off sample realism against diversity.
- **`aug_data_size_factor`** decides the # of counterfactuals per real point; values <1 mitigate class-imbalance–style bias, whereas >1 favors variance reduction.
- **`max_perturb_percent`** sets the half-width of the $[-p, +p]$ uniform scaling band; larger p injects broader *counterfactual sweep* but risks violating local linearity assumptions of the residual.

These three parameters explain the vast majority of between-trial variance in validation MSE, so limiting OPTUNA to a small budget remains effective. Median trial time is $\sim$7.5s (MLP) and $\sim$3.7s (XGB).

Table 7 reports the *modal* best-value across the nine benchmarks and their partitions.

Table 7. CRDA augmentation knobs: search priors and modal best values.

Knob	Search prior / range	Modal best
max_n_features_to_perturb	$\{1, 2, 3, 4, 5\}$	2
aug_data_size_factor	$\{0.50, 0.75, 1.00, 1.25, 1.50\}$	1.25
max_perturb_percent	$\{0.10, 0.20, \ldots, 1.00\}$	0.7

One-dataset sweep (House Price)

For illustration, we fix two of the three knobs at their modal best values (from Table 7) and systematically vary the remaining knob. Figures 5 and 6 show the resulting percentage change in MSE (*lower* is better) on the *House Price* dataset, averaged over five random seeds. We make several observations:

- **Augmenting data size** (`aug_data_size_factor`) appears more beneficial for MLP, presumably because additional training samples reduce overfitting; by contrast, XGB sees weaker or even mixed effects here, consistent with the notion that tree ensembles can already leverage smaller sets effectively.
- **Number of perturbed features** (`max_n_features_to_perturb`) shows an opposite preference: XGB yields stronger gains when more features are jointly modified, whereas MLP performance degrades if we perturb too many simultaneously (likely hurting the local consistency of the residual).
- **Perturbation magnitude** (`max_perturb_percent`) also diverges across baselines: larger scales help XGB discover more diverse synthetic points, but MLP tends to prefer smaller shifts in order to maintain stable gradients in training.

In short, although *both* models benefit from CRDA overall, their ideal hyper-parameter configurations differ. This shows the importance of model-aware tuning for effective data augmentation.

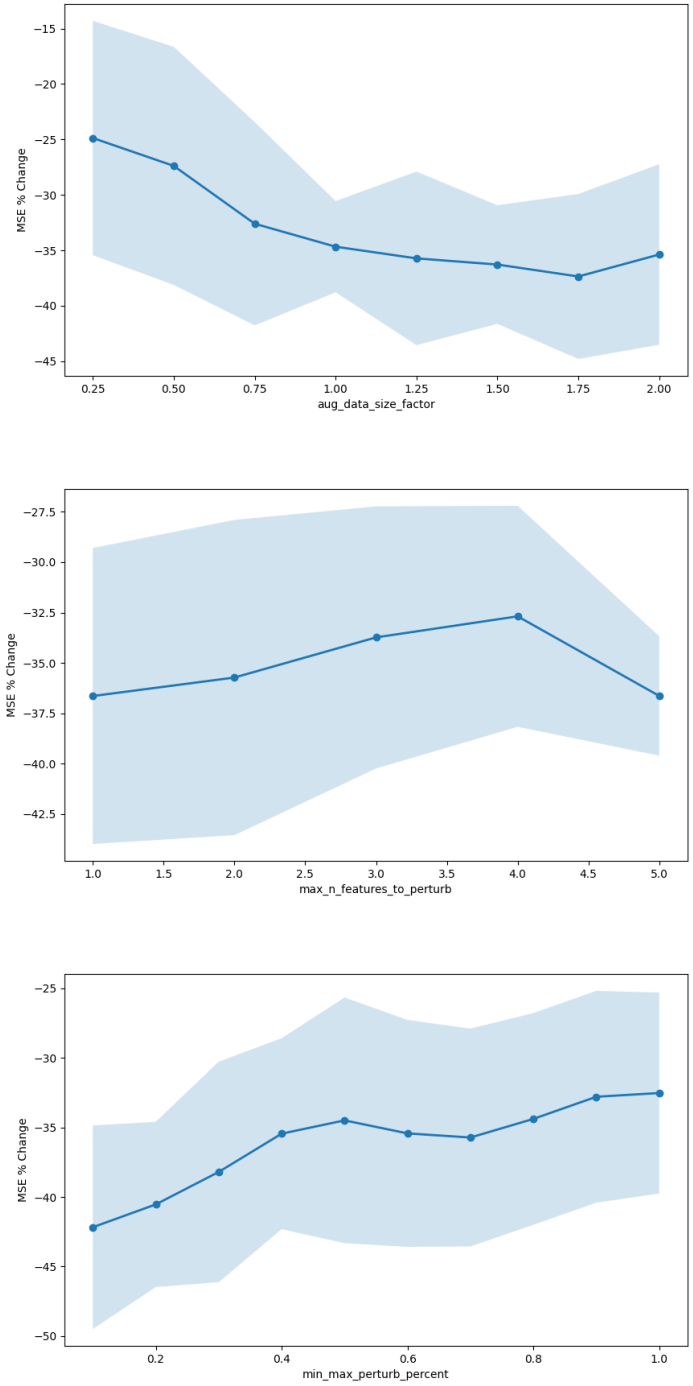


Figure 5. CRDA knob–sensitivity on the **MLP** baseline (HousePrice dataset).

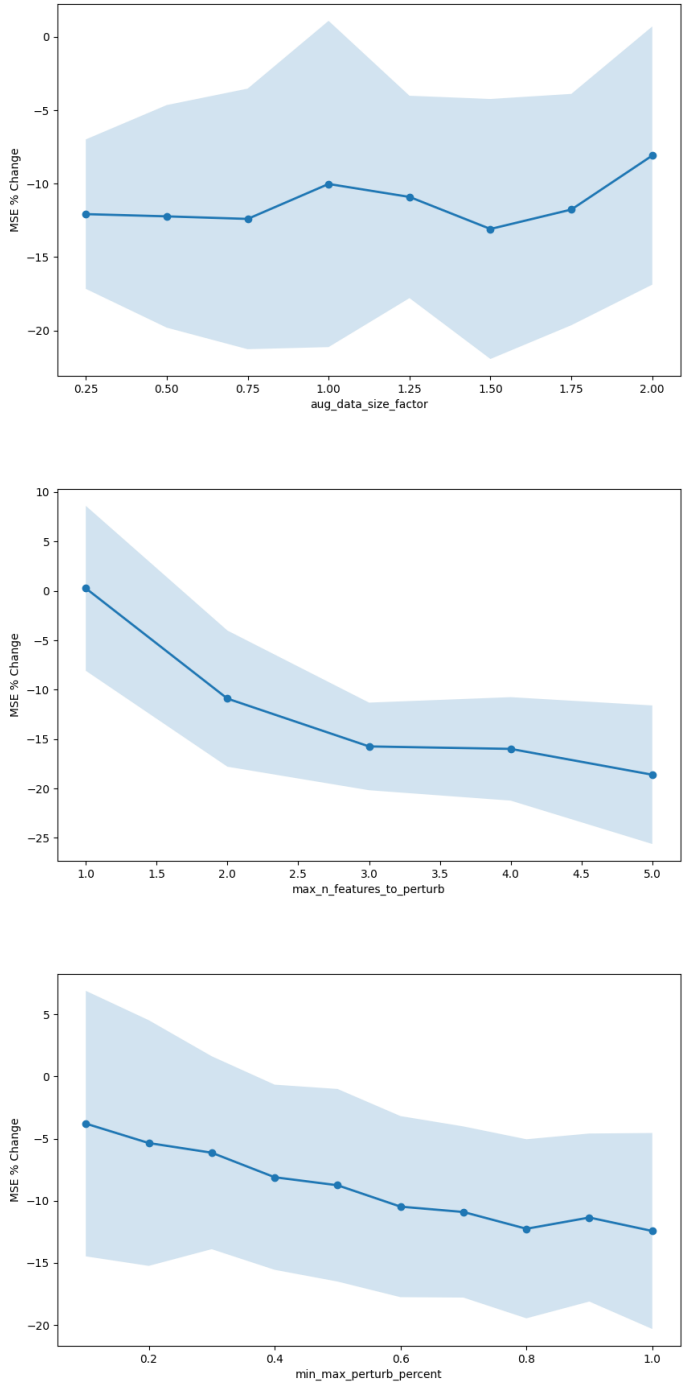


Figure 6. CRDA knob–sensitivity on the **XGB** baseline (HousePrice dataset).

G. Causal Implications of Assumption 3.1

This section gives causal analogues of our statistical independent assumptions and derives consequences for the causal relationship between the perturbed features X_P and the target variable Y.

Background: d-separation. We briefly recall the graphical notion underlying our assumptions (Pearl, 2009; Peters et al., 2017). A *path* between two nodes is any sequence of distinct adjacent edges, regardless of their orientation. A node W on a path is a *collider* if both adjacent edges point into it ($\rightarrow W \leftarrow$). Given a conditioning set S, a path is *blocked* if either (i) it contains a non-collider that is in S, or (ii) it contains a collider that is neither in S nor has a descendant in S. Two nodes are *d-separated given* S if every path between them is blocked, and *d-connected given* S otherwise. The central property we use is that d-separation in the graph implies conditional independence in any distribution that factorizes according to it: if A and B are d-separated given S, then $A \perp B \mid S$ (Pearl, 2009). The causal (d-separation) analogue of Assumption 3.1 is therefore that Z is d-separated from X_P given X_R.

We make the following assumption about the true data-generating causal graph G: (1) Z is the only latent parent of Y, representing the common interpretation of Z as summarizing unobserved causes influencing Y and (2) Z is d-separated from X_P given X_R, which implies Condition 3.1. In the following, write $PA(Y)$ for the parents of target variable Y.

Lemma G.1. *Let G be any causal graph whose variables comprise X_P, X_R, Z, Y, and possibly other latent variables U, such that Z is the only latent parent of Y and Z is d-separated from X_P given X_R. Then every perturbed nonparent $X \in (X_P - PA(Y))$ is d-separated from Y given $(PA(Y) \cap X_P) \cup X_R$.*

Proof. For contradiction, consider a path $X_0 - X_1 - \ldots - X_n - Y$ that d-connects $X_0 \in (X_P - PA(Y))$ with Y given $(PA(Y) \cap X_P) \cup X_R$. We use - to indicate a generic edge directed either way.

Let $X_k \in X_P$ be the last perturbed variable on the path. That is, $k \geq 0$ is the largest index such that $X_k \in X_P$. Since subpaths of a d-connecting path are also d-connecting paths, the subpath $X_k - \ldots - X_n - Y$ d-connects X_k to Y given $(PA(Y) \cap X_P) \cup X_R$. Since all variables between X_k and X_n are in X_R, we have that (*) the subpath d-connects X_k to X_n given X_R (only).

Case 1: The subpath is of the form $X_k - \ldots - X_n \rightarrow Y$. Then X_n is not an observed variable, since otherwise conditioning on $(PA(Y) \cap X_P) \cup X_R$ would block the path. Since Z is the only latent parent of Y, this means that $X_n = Z$. Therefore by the property (*), the subpath d-connects X_k to Z given X_R, which contradicts Assumption (2).

Case 2: The subpath is of the form $X_k - \ldots - X_n \leftarrow Y$. Therefore Y is not a collider on the path $X_k - \ldots - X_n \leftarrow Y \leftarrow Z$, and this path d-connects Z to $X_k \in X_P$ given X_R, which again contradicts Assumption (2).

Since both possible cases result in a contradiction, there is no path that d-connects a non-parent X_k to the target variable Y given the perturbed parents and X_R, which was to be proven. □

Proposition G.2. *Let G be any causal graph whose variables comprise X_P, X_R, Z, Y, and possibly other latent variables U, such that Z is the only latent parent of Y and Z is d-separated from X_P given X_R. Let G' be the truncated graph in which the variables in X_P have no parents. Then $P_G(Y|X) = P_{G'}(Y|X)$. Therefore the CRDA procedure of Section 4 computes the counterfactual $P(Y_{X=x'}|X = x, Y = y)$.*

Proof. Let $X_0 \in (X_P \cap PA(Y))$ be a perturbed parent of Y. The difference between the original graph G and the truncated graph G' is that the truncated graph removes backdoor paths that d-connect X_0 to Y given $(PA(Y) \cap X_P) \cup X_R$. We show that there is no such backdoor path in the original graph.

For contradiction, consider any backdoor path $X_0 \leftarrow X_1 - \ldots - X_n - Y$ that d-connects $X_0 \in (X_P \cap PA(Y))$ with Y given $(PA(Y) \cap X_P) \cup X_R$. As in the proof of Lemma G.1, let $X_k \in X_P$ be the last perturbed variable on the path. As in the previous proof, we have that (*) the subpath d-connects X_k to X_n given X_R (only).

Case 1: The subpath is of the form $X_k - \ldots - X_n \rightarrow Y$. Since Z is the only latent parent, there are only two possible cases.

Case 1a: $X_n = Z$. Then the subpath d-connects X_k to Z given X_R, which contradicts Assumption (2).
Case 1b: X_n is an observed variable. Then conditioning on $(PA(Y) \cap X_P) \cup X_R$ blocks a path ending in $X_n \rightarrow Y$ unless $X_n = X_0$. But then the path is of the form $X_0 \rightarrow Y$ and is not a backdoor path.

Case 2: The subpath is of the form $X_k - \ldots - X_n \leftarrow Y$. Then as in Case 2 of Lemma G.1, the path $X_k - \ldots - X_n \leftarrow Y \leftarrow Z$ d-connects X_k to Z given X_R, which contradicts Assumption 2.

Since there are no backdoor paths from perturbed parents in the non-truncated graph G, the conditional probabilities for perturbed parents are the same on both graphs:

$$P_G(Y|(PA(Y) \cap X_P) \cup X_R) = P_{G'}(Y|(PA(Y) \cap X_P) \cup X_R) \tag{3}$$

Since truncating only increases d-separation which implies independence, we conclude the following from Equation (3) and Lemma G.1:

$$P_G(Y|X_P, X_R) = P_G(Y|(PA(Y) \cap X_P) \cup X_R) = P_{G'}(Y|(PA(Y) \cap X_P) \cup X_R) = P_{G'}(Y|X_P, X_R) \tag{4}$$

Equation (4) shows that the conditional probabilities coincide in the original and truncated graphs, for every assignment of values to X, including $X = x'$. Therefore the regressor g fit on observational data estimates the correct interventional response. Under the additive-noise model, observing $(X = x, Y = y)$ further identifies the realized noise $Z = y - g(x)$, and reusing it while setting $X = x'$ yields $g(x') + z$; this is precisely CRDA's augmented label and equals the counterfactual outcome $Y_{X=x'}$, establishing that CRDA computes $P(Y_{X=x'} \mid X = x, Y = y)$. □

Remarks We note that Lemma G.1 and Proposition G.2 above condition on $(PA(Y) \cap X_P) \cup X_R$ rather than on X_R alone. The additional set $PA(Y) \cap X_P$ comprises the perturbed features that are themselves direct causes of Y; conditioning on them is necessary so that the remaining perturbed non-causes are blocked from Y, isolating the causal contribution of the perturbation. This is consistent with Assumption 3.1, which constrains the residual Z (the latent parent) and places no requirement on the observed parents of Y.

Our argument does not assume knowledge of the true causal graph G. The argument says that if we did know the true causal graph G, and used it to compute the counterfactual probability $P(Y_{X=x'}|X = x, Y = y)$, we would obtain the same result as estimating conditional probabilities $P(Y|X = x')$ after updating our posterior over the latent variable Z given the observations x, y.

The assumption that Z is the only latent parent of Y can be relaxed by allowing other latent parents but strengthening Assumption (2) to require that X_P is d-separated from all latent parents of Y given X_R.

Proposition G.2 is purely structural and does not depend on the particular form of the additive noise model. For example, it also applies to location-scale noise models (Sun & Schulte, 2023).

H. Validation of Assumptions and Component Analysis

In this section, we provide a deeper analysis of the CRDA framework. First, we empirically validate the core residual independence assumption using Mutual Information. Second, we perform ablation studies to demonstrate the benefits of applying CRDA compared to simplified baselines.

H.1. Empirical Validation of Residual Independence

A core theoretical assumption of CRDA (Assumption 3.1) is that the residual noise Z is conditionally independent of the features selected for perturbation (X_P), i.e., $P(Z|X_P, X_R) = P(Z|X_R)$. To validate this assumption and assess the effectiveness of our PC-algorithm/Correlation filter, we conducted an analysis measuring the Mutual Information (MI) between the residuals and the features. Mutual Information is an empirical estimator of the KL-Divergence $D_{KL}(P(Z, X)||P(Z)P(X))$; a value of zero indicates perfect independence.

We performed this evaluation across all 9 benchmark datasets using the XGBoost regressor over 15 random seeds. For each run, we calculated the MI (using the Kraskov KSG estimator (Kraskov et al., 2004)) for the set of features *Selected* (X_P) by CRDA versus the set of features *Rejected* (X_R).

The results are presented in Table 8. We observe that for datasets showing stronger feature-residual dependence (e.g., Energy Efficiency, House Price), the features rejected by our filter display significantly higher Mutual Information with the residuals

(up to $\approx 3\times$ higher) than the selected features. This confirms that the filter effectively identifies and removes features that would violate the independence assumption. For other datasets (e.g., Wine Quality, Wind Power), the MI scores for both selected and rejected features are uniformly low, indicating that the residuals are naturally independent of the features in these domains, and the filter correctly permits a wider range of perturbations.

Table 8. Evaluation of Feature-Residual Independence via Mutual Information (MI). We report the MI (in nats) between the model residuals Z and the features X, comparing features **Selected** by CRDA vs. those **Rejected**. Results are averaged over 15 seeds with standard errors. The **Ratio** column highlights the effectiveness of the filter in reducing divergence (higher is better).

Dataset	**Selected Features** (X_P) (Lower is better)	**Rejected Features** (X_R) (Higher implies dependence)	**Divergence Ratio** (MI_{Rej}/MI_{Sel})
House Price	$\mathbf{0.0056 \pm 0.0020}$	0.0155 ± 0.0023	$\mathbf{2.75}\times$
Energy Efficiency	$\mathbf{0.0054 \pm 0.0012}$	0.0160 ± 0.0023	$\mathbf{2.94}\times$
Parkinson's Monitoring	$\mathbf{0.0054 \pm 0.0006}$	0.0103 ± 0.0007	$\mathbf{1.92}\times$
Synthetic Regression	$\mathbf{0.0073 \pm 0.0013}$	0.0136 ± 0.0025	$\mathbf{1.85}\times$
Concrete Strength	0.0203 ± 0.0024	0.0320 ± 0.0035	$1.58\times$
CPU Performance	0.0136 ± 0.0010	0.0144 ± 0.0014	$1.06\times$
Wine Quality	0.0110 ± 0.0009	0.0136 ± 0.0011	$1.23\times$
Wind Power	0.0065 ± 0.0007	0.0084 ± 0.0007	$1.29\times$
Satellite Image	0.1031 ± 0.0013	0.1149 ± 0.0009	$1.11\times$

H.2. Ablation Studies

To verify that the independence assumption verified above translates to performance gains, we compare CRDA against two simplified ablation baselines:

- **Global Perturbation:** All features are perturbed randomly ($X_P = X$), ignoring the PC-algorithm and correlation checks.
- **Label Invariance:** Features are perturbed, but the label is kept fixed ($y' = y$), rather than recalculating $y' = \hat{g}(x') + \hat{z}$.

Table 9 presents the percentage change in MSE ($\Delta\%$) relative to the unaugmented base regressor across 3 representative datasets. CRDA consistently yields the largest error reduction. Notably, simple baselines often yield negligible improvements or even degrade performance (positive $\Delta\%$).

Table 9. Ablation results on Synthetic Regression, Energy Efficiency, and Parkinson's Monitoring datasets. Values represent the percentage change in MSE ($\Delta\%$) relative to the unaugmented baseline (lower is better). Results are averaged over 5 seeds with standard errors.

Dataset	**Model**	**MSE $\Delta\%$ Change ($\downarrow$)**		
		Global Perturbation	Label Invariance	CRDA
Synthetic Regression	MLP	-16.12 ± 4.30	-12.44 ± 32.43	$\mathbf{-38.94 \pm 4.02}$
	XGB	$+1.21 \pm 2.10$	-1.02 ± 1.33	$\mathbf{-3.62 \pm 1.93}$
Energy Efficiency	MLP	-14.50 ± 3.86	-2.65 ± 2.47	$\mathbf{-38.84 \pm 5.99}$
	XGB	-7.15 ± 9.75	-5.28 ± 8.78	$\mathbf{-17.45 \pm 5.16}$
Parkinson's Monitoring	MLP	-13.50 ± 8.90	$+0.55 \pm 19.04$	$\mathbf{-58.40 \pm 5.16}$
	XGB	$+0.36 \pm 1.57$	-3.09 ± 4.35	$\mathbf{-7.82 \pm 2.47}$

I. Statistical Significance Tests

For every dataset × training–set–fraction of our main experiment we did a 10-fold cross validation comparison of CRDA's augmented MSE against the corresponding raw unaugmented MSE with a two–sided Wilcoxon signed–rank test ($n_{\text{folds}} = 10$, $n_{\text{seeds}} = 15$ per cell). The heat-maps in Figures 7 and 8 visualize the outcome.

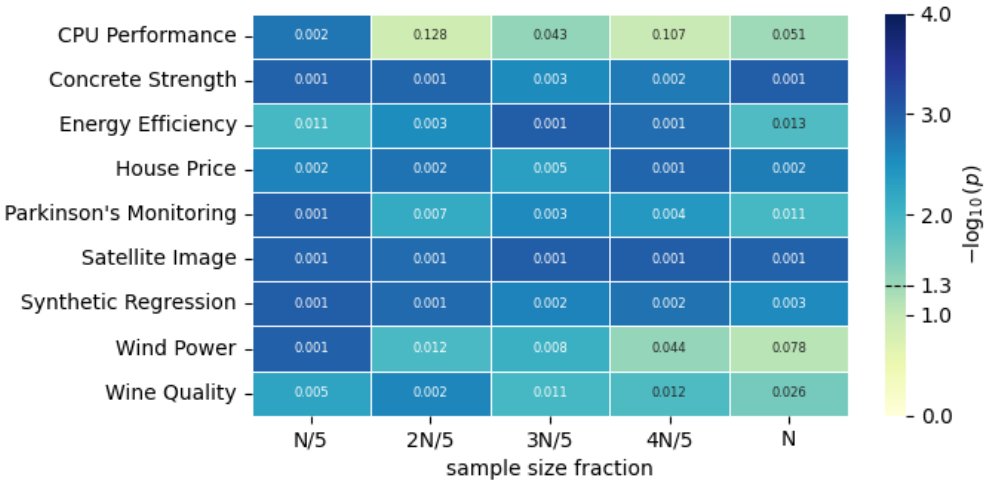


Figure 7. **MLP baseline.** Colour encodes $-\log_{10}(p)$; numbers are the mean p across 15 seeds. The dashed line on the colour-bar marks the $\alpha = 0.05$ threshold ($-\log_{10} p \approx 1.3$).

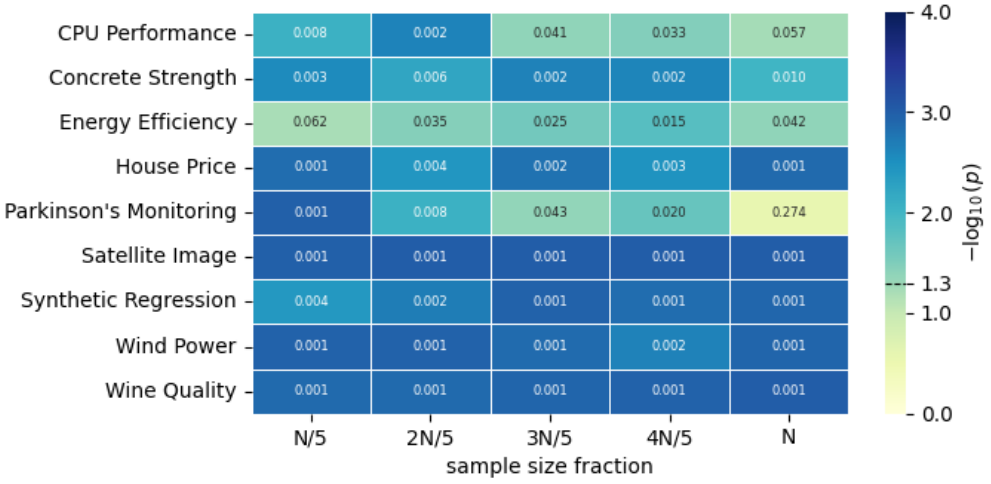


Figure 8. **XGB baseline.** Same layout and colour scale as Figure 7.

Brief analysis. Across *both* baselines the majority of cells are darker than the $\alpha = 0.05$ cut-off, indicating that CRDA delivers a statistically significant reduction in test–MSE for most dataset/size combinations. Significance is strongest for smaller training sets and occasionally weakens as the full dataset is used (e.g. `CPU Performance` and `Wind Power` for MLP, `Parkinson's Monitoring` for XGB), but even at n the method remains significant in 7/9 datasets with at least one baseline. These results support the robustness of the performance gains reported in the main paper.

J. Additional Baseline: CatBoost Analysis

To assess CRDA's robustness against stronger tree-based ensembles, we performed an additional evaluation using CatBoost (Prokhorenkova et al., 2018). CatBoost is often considered better than XGBoost due to its oblivious trees and robustness to overfitting, making it a challenging predictor to improve upon.

We evaluated performance at three fixed sample sizes ($N = \{300, 500, 700\}$) to observe behavior across different data availabilities.

Table 10 presents the percentage change in MSE ($\Delta\%$). We observe three distinct behaviors:

- **Consistent Gains:** On *House Price* and *Wind Power*, CRDA significantly reduces MSE across all sample sizes (peaking at -22.8% for *House Price*), demonstrating that CRDA behaves robustly for these tasks regardless of sample size.
- **Late-Stage Gains:** *CPU Performance* requires a sufficient number of samples to model the residual. It shows no benefit at $N = 300$ but improves substantially as data increases, reaching -13.0% at $N = 700$.
- **Sweet-Spot Behavior:** Datasets such as *Parkinson's Monitoring*, *Energy Efficiency*, and *Synthetic Regression* exhibit a "sweet spot" around $N = 500$, where the augmentation provides the most benefit ($\approx$ 4-5% reduction) before CatBoost potentially saturates the signal at larger sample sizes.

Table 10. Average percentage change MSE ($\Delta\%$) for *CatBoost* at fixed sample sizes across 15 seeds $\pm$ standard error.

Dataset	$N = 300$	$N = 500$	$N = 700$
House Price	-22.80 $\pm$ 1.97	-18.87 $\pm$ 1.69	-14.11 $\pm$ 1.64
CPU Performance	1.92 $\pm$ 2.99	-7.34 $\pm$ 1.91	-13.04 $\pm$ 2.96
Parkinson's Monitoring	-1.44 $\pm$ 1.99	-5.13 $\pm$ 1.97	-1.33 $\pm$ 1.49
Energy Efficiency	-1.65 $\pm$ 2.83	-4.95 $\pm$ 3.32	-1.89 $\pm$ 2.96
Synthetic Regression	-2.31 $\pm$ 1.86	-4.01 $\pm$ 1.97	-1.15 $\pm$ 2.16
Wind Power	-3.78 $\pm$ 1.23	-2.54 $\pm$ 0.79	-2.47 $\pm$ 0.95
Satellite Image	-0.95 $\pm$ 1.74	-2.08 $\pm$ 1.34	-1.95 $\pm$ 0.97
Wine Quality	-0.09 $\pm$ 1.28	-1.62 $\pm$ 0.96	-1.94 $\pm$ 0.68
Concrete Strength	-0.95 $\pm$ 1.85	-0.69 $\pm$ 1.29	0.46 $\pm$ 1.26

K. Linear Regression Base Predictor Study

In order to test our method against weaker base predictors; where separate systematic signal cannot be cleanly separated from noise, possibly violating our assumptions; we selected linear regression. Using the same 15 seeds and data settings as the main experiment, we conducted this study to observe how CRDA behaves.

Table 11 reports the averages and standard errors for baseline MSE, CRDA MSE, their percentage change (Δ %) as well as the $p - values$ from the Wilcoxon signed-rank test for every dataset and sample size subset.

We see that CRDA's filters *rejected* every single fold. Recall that for the Wilcoxon signed-rank test, if the p-value is above the 0.05 threshold, CRDA stops. We still report the Δ % if we had ignored the filter and observe how CRDA hurts here. CRDA therefore protects against weaker baselines, further illustrating how model-agnostic does not imply *always helpful*.

Table 11. Augmentation results for Linear Regression. Cells are green when data augmentation was selected to proceed according to the Wilcoxon signed rank test and red otherwise. Lower is better for the Δ MSE % change $\downarrow$.

Dataset	**Size**	**Linear Regression**			
		$\text{MSE}_{\text{baseline}}$	MSE_{CRDA}	Δ % $\downarrow$	p-value
CPU Performance	1638	0.011094 ± 0.000901	0.011066 ± 0.000871	0.04 ± 0.60	0.461 ± 0.036
	3276	0.010935 ± 0.000596	0.011012 ± 0.000625	0.58 ± 0.42	0.506 ± 0.039
	4914	0.009789 ± 0.000305	0.009784 ± 0.000306	-0.05 ± 0.24	0.452 ± 0.039
	6552	0.009834 ± 0.000372	0.009887 ± 0.000374	0.55 ± 0.19	0.515 ± 0.030
	8190	0.009705 ± 0.000359	0.009709 ± 0.000358	0.05 ± 0.10	0.456 ± 0.042
Satellite Image	1287	0.042119 ± 0.000676	0.042185 ± 0.000681	0.16 ± 0.15	0.240 ± 0.038
	2574	0.041148 ± 0.000522	0.041131 ± 0.000521	-0.04 ± 0.07	0.264 ± 0.033
	3861	0.040646 ± 0.000402	0.040666 ± 0.000407	0.05 ± 0.05	0.275 ± 0.026
	5148	0.040154 ± 0.000315	0.040183 ± 0.000306	0.08 ± 0.05	0.393 ± 0.038
	6435	0.040492 ± 0.000301	0.040509 ± 0.000299	0.04 ± 0.05	0.347 ± 0.032
Wind Power	1314	0.007335 ± 0.000271	0.007339 ± 0.000270	0.06 ± 0.11	0.501 ± 0.044
	2628	0.006363 ± 0.000134	0.006368 ± 0.000135	0.08 ± 0.06	0.468 ± 0.029
	3942	0.006580 ± 0.000102	0.006583 ± 0.000102	0.04 ± 0.04	0.493 ± 0.030
	5256	0.006583 ± 0.000087	0.006584 ± 0.000087	0.00 ± 0.03	0.498 ± 0.025
	6570	0.006175 ± 0.000052	0.006175 ± 0.000051	-0.00 ± 0.02	0.528 ± 0.034
Synthetic Regression	200	0.023265 ± 0.001023	0.023317 ± 0.001013	0.29 ± 0.63	0.306 ± 0.030
	400	0.022073 ± 0.000571	0.022101 ± 0.000572	0.13 ± 0.27	0.365 ± 0.033
	600	0.021332 ± 0.000500	0.021358 ± 0.000507	0.12 ± 0.16	0.385 ± 0.040
	800	0.015924 ± 0.000396	0.015945 ± 0.000400	0.12 ± 0.09	0.381 ± 0.039
	1000	0.015908 ± 0.000378	0.015911 ± 0.000373	0.03 ± 0.13	0.419 ± 0.032
Concrete Strength	201	0.016621 ± 0.001084	0.016592 ± 0.001080	-0.15 ± 0.29	0.362 ± 0.036
	402	0.017469 ± 0.000896	0.017431 ± 0.000917	-0.33 ± 0.32	0.352 ± 0.032
	603	0.017323 ± 0.000489	0.017336 ± 0.000507	0.03 ± 0.26	0.406 ± 0.029
	804	0.016711 ± 0.000571	0.016726 ± 0.000582	0.06 ± 0.16	0.452 ± 0.029
	1005	0.015719 ± 0.000358	0.015722 ± 0.000360	0.01 ± 0.08	0.477 ± 0.025
Energy Efficiency	153	0.003620 ± 0.000327	0.003650 ± 0.000326	1.02 ± 0.95	0.413 ± 0.044
	306	0.002928 ± 0.000122	0.002926 ± 0.000122	-0.03 ± 0.37	0.398 ± 0.039
	459	0.002872 ± 0.000124	0.002873 ± 0.000123	0.04 ± 0.23	0.453 ± 0.042
	612	0.002615 ± 0.000086	0.002621 ± 0.000084	0.29 ± 0.31	0.452 ± 0.036
	765	0.002667 ± 0.000056	0.002673 ± 0.000059	0.21 ± 0.20	0.436 ± 0.026
House Price	200	0.000103 ± 0.000006	0.000103 ± 0.000006	0.77 ± 0.54	0.344 ± 0.030
	400	0.000100 ± 0.000004	0.000101 ± 0.000004	0.10 ± 0.19	0.463 ± 0.039
	600	0.000100 ± 0.000004	0.000100 ± 0.000004	0.07 ± 0.13	0.515 ± 0.024
	800	0.000104 ± 0.000003	0.000104 ± 0.000003	0.03 ± 0.09	0.476 ± 0.047
	1000	0.000103 ± 0.000003	0.000104 ± 0.000003	0.22 ± 0.17	0.445 ± 0.042
Parkinson's Monitoring	1175	0.004750 ± 0.000112	0.004754 ± 0.000112	0.08 ± 0.10	0.425 ± 0.043
	2350	0.004672 ± 0.000096	0.004681 ± 0.000099	0.18 ± 0.14	0.333 ± 0.026
	3525	0.004651 ± 0.000096	0.004650 ± 0.000095	-0.02 ± 0.06	0.355 ± 0.028
	4700	0.004618 ± 0.000077	0.004620 ± 0.000076	0.05 ± 0.05	0.449 ± 0.030
	5875	0.004655 ± 0.000054	0.004655 ± 0.000054	0.01 ± 0.03	0.429 ± 0.024
Wine Quality	1063	0.021526 ± 0.000658	0.021544 ± 0.000670	0.07 ± 0.24	0.393 ± 0.026
	2126	0.015126 ± 0.000321	0.015131 ± 0.000318	0.04 ± 0.05	0.452 ± 0.033
	3189	0.015448 ± 0.000206	0.015461 ± 0.000204	0.09 ± 0.05	0.443 ± 0.027
	4252	0.014830 ± 0.000270	0.014839 ± 0.000269	0.06 ± 0.04	0.487 ± 0.035
	5315	0.014894 ± 0.000168	0.014896 ± 0.000170	0.01 ± 0.03	0.505 ± 0.024

L. Complete Per-Dataset Scores

Table 12 reports the baseline MSE, CRDA MSE and their percentage change (Δ %) for every dataset and sample size subset. It is a more comprehensive version of Table 1 in the main paper. These results are the averages across 15 different seed runs and so we include their standard errors.[5]

Table 12. Complete results with standard errors for XGB and MLP across 15 seeds for each of the 9 datasets. Lower is better ↓.

Dataset	Sample Size	XGB ↓			MLP ↓		
		$\text{MSE}_{\text{baseline}}$	MSE_{CRDA}	Δ %	$\text{MSE}_{\text{baseline}}$	MSE_{CRDA}	Δ %
CPU Performance	1638	0.00097 ± 0.00005	0.00089 ± 0.00004	-7.0 ± 2.8	0.00112 ± 0.00007	0.00087 ± 0.00003	-20.2 ± 3.3
	3276	0.00088 ± 0.00003	0.00079 ± 0.00002	-9.5 ± 2.3	0.00100 ± 0.00002	0.00085 ± 0.00002	-14.0 ± 1.6
	4914	0.00077 ± 0.00002	0.00072 ± 0.00001	-6.2 ± 1.3	0.00093 ± 0.00002	0.00082 ± 0.00001	-11.3 ± 1.6
	6552	0.00073 ± 0.00002	0.00069 ± 0.00001	-4.1 ± 1.7	0.00090 ± 0.00003	0.00079 ± 0.00001	-10.5 ± 2.2
	8190	0.00074 ± 0.00002	0.00070 ± 0.00001	-5.2 ± 2.0	0.00087 ± 0.00001	0.00078 ± 0.00001	-10.2 ± 0.8
Satellite Image	1287	0.01778 ± 0.00047	0.01697 ± 0.00053	-4.5 ± 1.5	0.02031 ± 0.00104	0.01629 ± 0.00059	-18.4 ± 3.2
	2574	0.01636 ± 0.00037	0.01576 ± 0.00041	-3.7 ± 0.9	0.01747 ± 0.00039	0.01455 ± 0.00041	-16.7 ± 1.5
	3861	0.01460 ± 0.00035	0.01390 ± 0.00035	-4.8 ± 0.8	0.01585 ± 0.00054	0.01211 ± 0.00037	-23.1 ± 1.8
	5148	0.01366 ± 0.00033	0.01300 ± 0.00031	-4.7 ± 1.1	0.01415 ± 0.00045	0.01076 ± 0.00030	-23.7 ± 1.3
	6435	0.01254 ± 0.00030	0.01186 ± 0.00027	-5.3 ± 0.8	0.01232 ± 0.00035	0.00989 ± 0.00029	-19.7 ± 1.1
Wind Power	1314	0.00742 ± 0.00029	0.00721 ± 0.00029	-2.8 ± 1.2	0.00752 ± 0.00025	0.00697 ± 0.00025	-7.2 ± 1.6
	2628	0.00602 ± 0.00012	0.00603 ± 0.00013	0.2 ± 0.6	0.00621 ± 0.00017	0.00562 ± 0.00011	-9.2 ± 1.5
	3942	0.00586 ± 0.00008	0.00578 ± 0.00008	-1.3 ± 0.4	0.00593 ± 0.00008	0.00539 ± 0.00008	-9.0 ± 0.7
	5256	0.00570 ± 0.00006	0.00562 ± 0.00007	-1.4 ± 0.4	0.00567 ± 0.00009	0.00533 ± 0.00009	-6.2 ± 0.4
	6570	0.00528 ± 0.00005	0.00522 ± 0.00005	-1.1 ± 0.3	0.00530 ± 0.00004	0.00500 ± 0.00004	-5.6 ± 0.5
Synthetic Regression	200	0.00652 ± 0.00044	0.00564 ± 0.00032	-12.0 ± 3.8	0.01993 ± 0.00178	0.01387 ± 0.00163	-28.8 ± 6.5
	400	0.00327 ± 0.00027	0.00312 ± 0.00023	-3.2 ± 2.5	0.00610 ± 0.00037	0.00384 ± 0.00032	-36.9 ± 3.1
	600	0.00264 ± 0.00009	0.00242 ± 0.00007	-7.9 ± 2.2	0.00321 ± 0.00027	0.00228 ± 0.00019	-27.9 ± 3.1
	800	0.00165 ± 0.00008	0.00161 ± 0.00009	-2.2 ± 2.1	0.00223 ± 0.00017	0.00140 ± 0.00008	-34.1 ± 4.1
	1000	0.00152 ± 0.00005	0.00145 ± 0.00006	-4.6 ± 2.9	0.00220 ± 0.00014	0.00123 ± 0.00007	-42.3 ± 3.2
Concrete Strength	201	0.00777 ± 0.00071	0.00701 ± 0.00065	-8.0 ± 3.8	0.01033 ± 0.00106	0.00793 ± 0.00052	-17.8 ± 5.9
	402	0.00493 ± 0.00036	0.00453 ± 0.00038	-8.4 ± 2.8	0.00635 ± 0.00055	0.00496 ± 0.00038	-19.8 ± 2.9
	603	0.00473 ± 0.00025	0.00427 ± 0.00025	-9.7 ± 2.2	0.00602 ± 0.00015	0.00494 ± 0.00014	-17.6 ± 2.5
	804	0.00365 ± 0.00017	0.00307 ± 0.00015	-15.7 ± 1.9	0.00497 ± 0.00026	0.00361 ± 0.00013	-24.8 ± 4.2
	1005	0.00290 ± 0.00010	0.00256 ± 0.00013	-12.2 ± 2.1	0.00422 ± 0.00025	0.00306 ± 0.00017	-26.9 ± 1.8
Energy Efficiency	153	0.00399 ± 0.00050	0.00344 ± 0.00052	-13.3 ± 8.0	0.00583 ± 0.00049	0.00426 ± 0.00048	-25.1 ± 7.0
	306	0.00233 ± 0.00015	0.00206 ± 0.00015	-12.2 ± 3.3	0.00321 ± 0.00015	0.00233 ± 0.00021	-28.1 ± 5.0
	459	0.00165 ± 0.00012	0.00143 ± 0.00012	-10.5 ± 6.1	0.00188 ± 0.00016	0.00106 ± 0.00013	-43.0 ± 4.8
	612	0.00128 ± 0.00008	0.00100 ± 0.00006	-19.3 ± 5.5	0.00091 ± 0.00008	0.00052 ± 0.00006	-40.7 ± 4.0
	765	0.00097 ± 0.00006	0.00076 ± 0.00007	-21.0 ± 4.5	0.00053 ± 0.00008	0.00035 ± 0.00003	-28.3 ± 4.4
House Price	200	0.00079 ± 0.00009	0.00064 ± 0.00005	-14.2 ± 4.9	0.00102 ± 0.00012	0.00057 ± 0.00007	-40.6 ± 5.0
	400	0.00033 ± 0.00002	0.00031 ± 0.00002	-5.4 ± 2.3	0.00041 ± 0.00003	0.00025 ± 0.00001	-37.0 ± 3.8
	600	0.00027 ± 0.00002	0.00026 ± 0.00002	-4.9 ± 2.8	0.00029 ± 0.00002	0.00020 ± 0.00002	-30.1 ± 3.9
	800	0.00024 ± 0.00001	0.00022 ± 0.00001	-9.9 ± 2.1	0.00023 ± 0.00001	0.00016 ± 0.00001	-30.3 ± 4.2
	1000	0.00020 ± 0.00001	0.00018 ± 0.00001	-6.5 ± 1.9	0.00019 ± 0.00001	0.00014 ± 0.00001	-27.0 ± 2.6
Parkinson's Monitoring	1175	0.00079 ± 0.00003	0.00072 ± 0.00003	-8.4 ± 2.5	0.00165 ± 0.00012	0.00101 ± 0.00006	-36.2 ± 4.0
	2350	0.00034 ± 0.00002	0.00032 ± 0.00001	-6.6 ± 2.9	0.00080 ± 0.00005	0.00054 ± 0.00003	-31.8 ± 2.6
	3525	0.00021 ± 0.00001	0.00020 ± 0.00001	-2.8 ± 3.5	0.00048 ± 0.00003	0.00030 ± 0.00002	-36.6 ± 4.1
	4700	0.00015 ± 0.00001	0.00014 ± 0.00001	-6.3 ± 2.5	0.00042 ± 0.00003	0.00021 ± 0.00001	-46.4 ± 4.2
	5875	0.00011 ± 0.00001	0.00011 ± 0.00001	1.7 ± 3.9	0.00026 ± 0.00002	0.00013 ± 0.00001	-47.2 ± 4.8
Wine Quality	1063	0.02057 ± 0.00058	0.02062 ± 0.00056	0.3 ± 0.8	0.02291 ± 0.00091	0.02284 ± 0.00133	-0.3 ± 3.4
	2126	0.01416 ± 0.00030	0.01429 ± 0.00030	1.0 ± 0.8	0.01539 ± 0.00027	0.01458 ± 0.00034	-5.2 ± 1.6
	3189	0.01391 ± 0.00020	0.01386 ± 0.00017	-0.3 ± 0.5	0.01478 ± 0.00024	0.01423 ± 0.00025	-3.6 ± 1.6
	4252	0.01332 ± 0.00025	0.01324 ± 0.00027	-0.6 ± 0.5	0.01386 ± 0.00028	0.01323 ± 0.00025	-4.4 ± 0.9
	5315	0.01332 ± 0.00013	0.01318 ± 0.00015	-1.1 ± 0.3	0.01397 ± 0.00017	0.01328 ± 0.00020	-5.0 ± 0.6

M. Compute Budget and Carbon Footprint

- **Hardware.** AWS `c7i.24xlarge` (96 vCPU, 192 GB RAM, Xeon Platinum 8480C, ≈0.59 kW active draw).[6]
- **Runtime.** 13.562 h total (9.103 h MLP baseline, 4.459 h XGB baseline).
- **Energy.** $13.562 \times 0.59 \approx 8.0$ kWh.
- **CO_2-eq.** Local grid intensity 34.5 g CO_2/kWh $\Rightarrow 8.0 \times 0.0345 \approx$ 0.28 kg CO_2.

[5] Per-seed results are available in the code repository at `experiments/{dataset}/{model}/interim_results`

[6] Power estimate from Intel C7i workload proof sheet.