Counterfactual Residual Data Augmentation for Regression

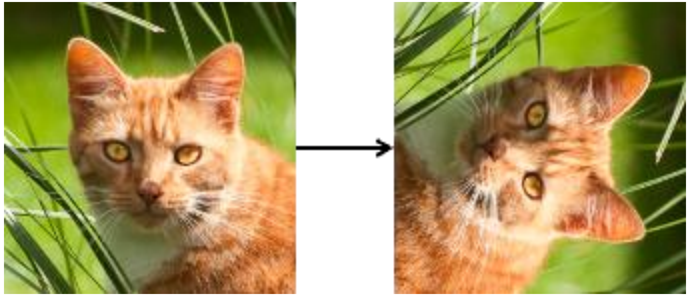
ICML 2026

crda-project.github.io

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

[1]University of Waterloo, [2]Vector Institute, [3]Simon Fraser University

Why “Standard” Augmentation Fails



Invariant (still a cat)

Original	Perturbed
[Age: 30, Blood Pressure: 120, Cancer Risk: 0.05]	[Age: 60, Blood Pressure: 240, Cancer Risk: 0.05]

Manifold Intrusion (is the patient still alive?)

Vision/NLP:

- Domain-specific symmetries exist (Rotation, Translation)
- Labels typically do not change

Tabular Data:

- No spatial/temporal structure
- Perturbing features can arbitrarily destroy the relationship with the target

Our Intuition

The House Price Problem:

- *Task is to predict the price of a house given features about it*

Features (systematic signal):

- Location, Square Footage, Build Year, Garage Finish

Unobserved Factors (noise):

- Bidding War, The Buyer’s Urgency

Our Intuition

The House Price Problem:

- *Task is to predict the price of a house given features about it*

Decomposition:

- Price = Signal + Noise

The Insight:

- Changing a secondary feature (e.g., garage finish) alters the systematic price, but not the specific noise (the bidding war)

Residual Invariance:

- We can synthesize new samples by perturbing certain features while preserving the observed residual (noise)

Formalizing the Insight

Additive Noise Model: $Y = g(X) + Z$

- **Y** is the target, **g** is the learnable signal, **Z** is the unobserved noise

Features are split into:

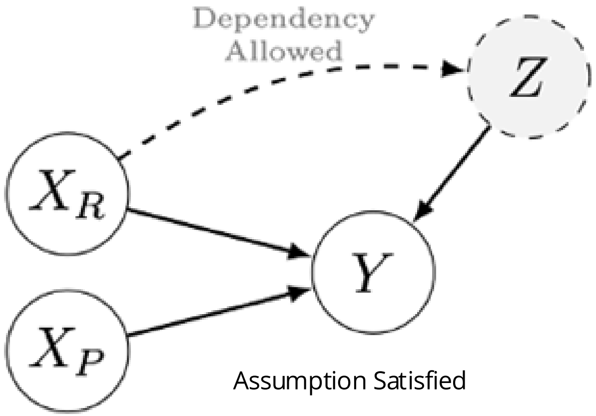
- X_P the perturbable (those we intend to modify)
- X_R the remaining (those we hold fixed)

Assumption (Residual Invariance):

- The noise is conditionally independent of the perturbed features

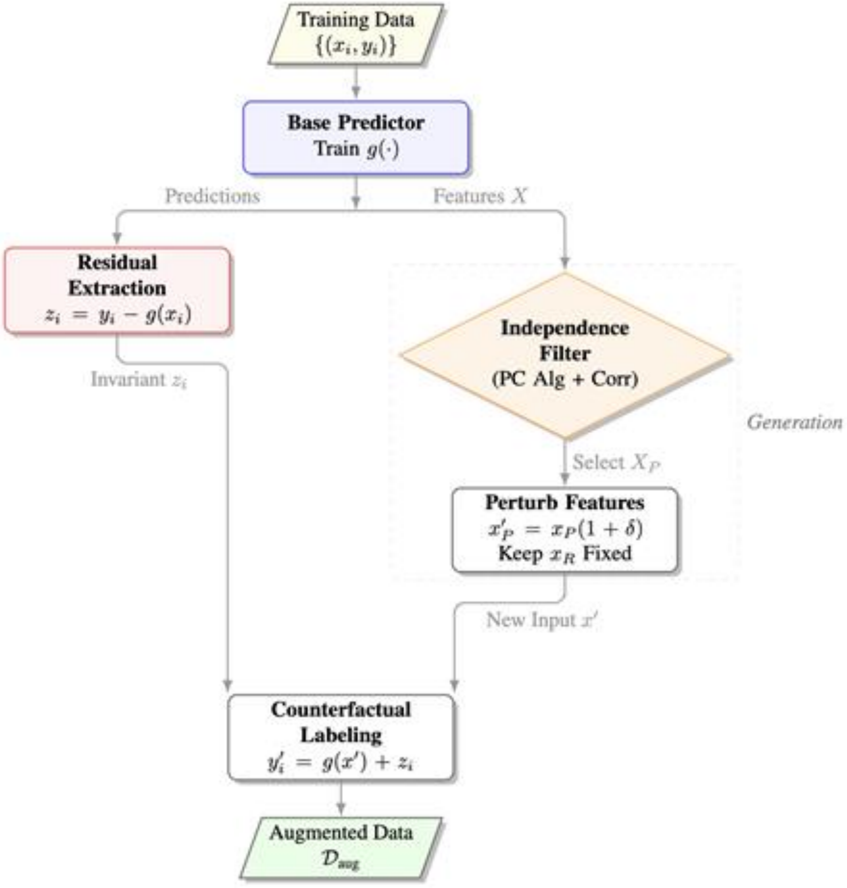
Counterfactual Reasoning:

- "Given that we observed outcome $Y = y$ for individual i, what would Y have been if X had been different?"



Assumption Satisfied

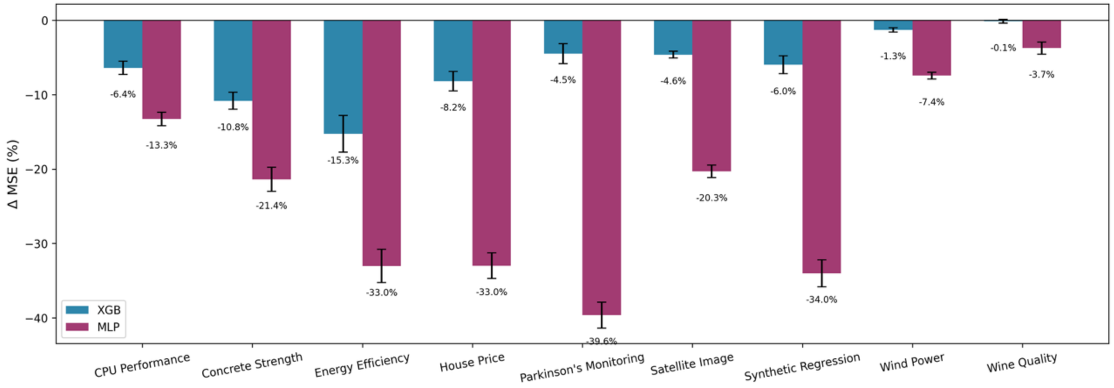
Counterfactual Residual Data Augmentation (CRDA)



Results

Across 9 real-world benchmarks (UCI, PMLB, Kaggle):

- **MLP Error Reduction: -22.9% Average MSE**
- **XGBoost Error Reduction: -6.4% Average MSE**



Beat state-of-the-art data augmentation/generation techniques:

- C-Mixup, ADA, TabDDPM, TVAE, CTGAN

Contribution

Released the fully autonomous CRDA pipeline as a pip package:

- https://pypi.org/project/crda/

```
pip install crda
```

Project Page:

- https://crda-project.github.io/

```python
from crda import CRDA, Config
from xgboost import XGBRegressor

# Configure the experiment
config = Config(
    dataset="path/to/your/data.csv",
    dataset_name="my_dataset",
    random_seed=42,
    verbose=True
)

# Create CRDA instance
crda = CRDA(config)

# Run the augmentation pipeline with your model
results = crda.run(XGBRegressor())
```