Why You Should Split Your Data Before Correcting Skewness

Executive Summary

Addressing skewness in your features is a critical preprocessing step for many machine learning models. However, fitting skew-correction transforms on the entire dataset *before* splitting into training and test sets introduces data leakage and inflates performance estimates. This report explains why you should always split your data first, then fit skewness-correction (and any other parameterized preprocessing) on the training set only, and finally apply the learned transformation to the test set.

---

1. Introduction

Proper validation of model performance relies on keeping the test data truly unseen during all aspects of model building, including preprocessing. Skewness—non-normal distributions with long tails—can impair many models' assumptions and convergence. Common corrections include logarithmic or square-root transforms, as well as parameterized methods like Box–Cox or Yeo–Johnson. When these transforms estimate parameters from the data, fitting them on the full dataset risks contaminating your evaluation.

2. Understanding Skewness and Its Correction

- **Skewness** measures asymmetry in a variable's distribution. Positive skew (right tail) often appears in financial or count data; negative skew (left tail) appears in bounded scales.
- **Data-driven transforms** (e.g., Box–Cox, Yeo–Johnson) estimate one or more parameters (λ) that best normalize the distribution.
- **Fixed transforms** (e.g., $\log(x + 1)$, square-root) do not estimate parameters and therefore do not introduce leakage—but including them in a train-fitted pipeline promotes consistency.

3. The Risk of Data Leakage

Data leakage occurs when information from the test set is inadvertently used during training. Fitting skew-correction on the entire dataset before splitting leaks statistics (such as the optimal λ) from the test set into the model. Consequences include:

- **Biased parameter estimates:** The skew-correction becomes tuned to all available data, making training artificially easier.
- **Overoptimistic performance:** Evaluation on “leaked” test data underestimates real-world error.
- **Reduced reproducibility:** Future data may exhibit different skew patterns; relying on test-derived statistics hides this variability.

4. Best Practice: Split First, Then Fit

1. **Split the dataset** into training and test subsets (commonly an 80/20 or 70/30 ratio).
2. **Fit parameterized preprocessing** (e.g., estimating λ for Box–Cox) *only* on the training subset.
3. **Apply the learned transformation** to the test subset using the parameters derived from training.
4. **Train and evaluate** your model on the transformed training and test sets, respectively.

This workflow ensures a fair, unbiased evaluation and preserves the integrity of the test set as a proxy for unseen data.

5. Practical Example (Conceptual)

- **Data split:** Reserve 20 % of the data as a hold-out test set.
- **Pipeline construction:** Create a preprocessing sequence that includes skew-correction followed by any additional steps (e.g., scaling, encoding).
- **Pipeline fitting:** Estimate skew-correction parameters using only the training data.
- **Transformation application:** Use the fitted pipeline to transform both training and test data.
- **Model training and evaluation:** Train the model on the transformed training set, then evaluate on the transformed test set.

This approach keeps the test data unseen until the very last step, yielding realistic performance metrics.

6. Conclusion

Splitting your dataset *before* fitting skew-correction or any other parameterized preprocessing prevents data leakage, yields reliable performance estimates, and ensures your model generalizes effectively to new, unseen data. Adopting this discipline in every machine learning workflow promotes scientific rigor and robust model validation.