International Journal of Web of Multidisciplinary Studies
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
website: http://ijwos.com
Vol.02 No.12.
OPEN ACCESS
E-ISSN : 3049-2424
DOI: https://doi.org/10.71366/ijwos

A Study on the impact of Data Preprocessing in Traditional Machine Learning Pipeline using Column Transformer

Anu Priya J[1], Priti Mondal [2]
[1,2] *Associate Faculty, Airtics, Exeed College, Sharjah, UAE*

Article Info

Article History:

Published:10 Dec 2025

Publication Issue:
Volume 2, Issue 12
December-2025

Page Number:
177-185

Corresponding Author:
Anu Priya J

Abstract:

Data Preprocessing is a crucial step that has a huge impact on the overall result of the model irrespective of the algorithm. There is limited exploration of how preprocessing techniques specifically impact the performance of models across a variety of preprocessing approaches. The challenge is in identifying which preprocessing methods best improve the models' performance on real-world datasets, such as the Adult Income dataset. This study aims to explore the volume of impact each preprocessing model has when it is combined with one another. StandardScaler, MinMaxScaler, and OneHotEncoder are associated with Logistic Regression and Support Vector Machine to determine its effectiveness. Each Preprocessing method has its own influence on the outcome of the results. The goal is to compare the methods and observe which preprocessing techniques enhance the performance of machine learning models and under which circumstances these techniques are most effective.

Keywords: Data Preprocessing, Column Transformer, MinMaxScaler, OneHotEncoder, StandardScalar, Logistic Regression, Support Vector Machine.

1. Introduction

Machine Learning [ML] is a subset of artificial intelligence that focuses on enabling the system to learn from the data without explicit programming. The quality of a Machine Learning model is based on two major factors: the dataset quality features, and the pipeline of the model building. A crucial part of building an ML model involves well laid out preprocessing work. Building a model starts with collecting the data, cleaning and preparing the data [preprocessing], training the model, testing the model, and iterative improvements until the results are satisfactory. Here, the iterative analysis focusses on data preprocessing techniques on most scenarios to improve the outcome because of its magnitude. Traditional ML algorithms like Support Vector Machine and Logistic Regression have a dependency on well-processed data. This study is to use that dependency and discover the importance of each preprocessing step and its impact on the overall model performance.

1.1 Problem Statement

In traditional machine learning pipelines, data preprocessing remains critical yet inconsistent step especially when dealing with a dataset of diverse datatypes [numerical, categorical, timeseries, textual,].

Manual preprocessing may lead to

- Data Leakage if scaling is done before train-test split

Copyright © 2025 The Author(s) : This is an open access article under the CC BY license (http://creativecommons.org/licenses/by-nc/4.0/)

- Redundancy of the code decreases the modularity of the performing model
- Varying accuracy up to ± 5% due to improper feature normalization

The purpose of this study is to investigate how scikit's column transformer can optimize the preprocessing steps by encapsulating transformations into a single reusable compound, preventing data leakage, and enabling fair comparisons between preprocessing strategies. By evaluating Logistic regression and SVM we can quantify the benefits of ColumnTransformer using evaluation metrics, Code Maintenance, and reproducibility. How does the incorporation of ColumnTransformer into ML pipeline affect the performance of traditional models while dealing with mixed type datasets and which preprocessing strategy yields optimal results?

After a comprehensive literature review it is evident that most studies focus on either numerical or categorical preprocessing and only a few systematically compare the features together. This study focusses on addressing this gap and testing the combination of scaling methods.

1.2 Research Objectives

- To compare the impact of StandardScaler, MinMaxScaler, and no scaling on traditional ML models using ColumnTransformer.
- To evaluate how ColumnTransformer improves reproducibility and reduces manual preprocessing effort.
- Analyze model-specific sensitivity (Logistic Regression vs. SVM) to different preprocessing strategies.

1.3 Research Questions

- How does preprocessing (StandardScaler vs. MinMaxScaler vs. raw) affect model accuracy in pipelines using ColumnTransformer?
- Does ColumnTransformer significantly reduce code complexity and improve reproducibility compared to manual preprocessing?
- Why do Logistic Regression and SVM respond differently to the same preprocessing techniques?

2. Literature Review

2.1 The Critical Role of Data Preprocessing in Machine Learning

Data preprocessing is widely acknowledged as a foundational stage in the machine learning (ML) workflow, often consuming a substantial share of the overall project timeline. In practical applications, raw data is rarely suitable for immediate modeling; it typically contains missing values, noise, inconsistencies, and heterogeneous feature types. As Maharana et al. (2022) emphasize, preprocessing and data augmentation are essential for improving data quality, which in turn has a direct influence on model performance. Despite this, preprocessing is frequently treated as a routine preliminary step rather than a strategic component of model development. Chahid et al. (2023) argue that in high-stakes domains such as healthcare, preprocessing extends far beyond data formatting and plays a critical role in ensuring the reliability of predictive systems. Nonetheless, a key challenge persists: determining which specific combinations of preprocessing techniques yield the most substantial performance gains for standard algorithms, particularly when working with mixed-type datasets.

2.2 **Feature Scaling and Algorithm Sensitivity**

Existing literature clearly demonstrates that traditional ML algorithms vary significantly in their sensitivity to feature scale. For instance, Support Vector Machines (SVMs) depend heavily on distance-based calculations, such as Euclidean distance, to identify optimal hyperplanes. Without appropriate scaling, features with larger numeric ranges disproportionately influence these distance metrics, often resulting in suboptimal classification boundaries. This sensitivity makes proper normalization essential, as neglecting it can lead to notable fluctuations in predictive accuracy.

Logistic Regression, although fundamentally a probabilistic model, typically relies on gradient-based optimization techniques. When features are left unscaled, the objective function can adopt an elongated or skewed geometry, which may slow down convergence or increase the likelihood of the solver becoming trapped in local minima. However, unlike SVMs, Logistic Regression tends to be comparatively stable in the absence of scaling, although performance and convergence efficiency still benefit from it.

While many studies have analyzed scaling and encoding separately, there remains a shortage of systematic comparisons that evaluate their combined influence—particularly the interaction of methods such as StandardScaler, MinMaxScaler, and OneHotEncoder—within a unified preprocessing pipeline for mixed-type datasets like the Adult Income dataset.

2.3 The Shift Toward Modular Pipelines (ColumnTransformer)

Traditional approaches to preprocessing often rely on ad-hoc scripts executed prior to train–test splitting. Such methods introduce risks related to reproducibility and increase the likelihood of data leakage, wherein information from the test set inadvertently influences the training process—for example, when scaling parameters such as means or variance are computed over the entire dataset.

Modern ML frameworks have responded to these challenges through the introduction of modular tools such as the ColumnTransformer and Pipeline. These constructs allow preprocessing steps to be encapsulated into reusable, transparent components that ensure transformations are fitted exclusively on the training data and applied consistently thereafter. Although technical documentation for these tools is extensive, empirical studies that quantify their contributions to code maintainability, pipeline robustness, and reproducibility remain relatively scarce.

2.4 **Evaluation Metrics and Fairness in Preprocessing**

Meaningfully assessing the impact of preprocessing requires moving beyond accuracy alone, particularly when working with imbalanced datasets such as Adult Income. Varoquaux and Colliot (2023) highlight the value of incorporating metrics such as precision, recall, and F1-score to obtain a more nuanced understanding of model behavior. Emerging research further underscores that preprocessing choices may have ethical implications. Wu (2024), for example, shows that certain preprocessing strategies can unintentionally influence fairness outcomes in probability-of-default and related risk models, underscoring the need for careful methodological selection.

2.5 Research Gap

A review of the current literature reveals several key gaps:

Limited Comparative Analysis on Mixed Data: Existing research frequently isolates numerical scaling or categorical encoding rather than evaluating their combined impact within a unified preprocessing framework.

Lack of Quantification of Pipeline Benefits: Although the theoretical advantages of tools such as Scikit-Learn's ColumnTransformer—particularly in preventing data leakage—are widely recognized, empirical evidence demonstrating improvements in performance consistency and code modularity remains limited.

Need for Algorithm-Specific Insights: There is an opportunity to refine our understanding of how established models such as SVMs and Logistic Regression respond to contemporary, modular pipeline architectures. Notably, there is insufficient empirical evidence distinguishing why SVM performance may degrade sharply in the absence of appropriate scaling, whereas Logistic Regression typically exhibits greater robustness.

Addressing these gaps, the present study seeks to empirically evaluate the influence of ColumnTransformer-based pipelines on model performance, reproducibility, and sensitivity for both Logistic Regression and SVM classifiers. The study aims to clarify how preprocessing choices shape algorithmic outcomes and contribute to more reliable modeling practices for mixed-type datasets.

3. Case and Methodology

3.1 Dataset

The dataset preferred for this study is the most popular "Adult Income" dataset with 14 features of both numerical and categorical types. There are 48842 instances before handling the null values and 45222 after removing the null values. This is a suitable dataset for this study because it contains both numerical and categorical features. Target variable of this dataset is to determine whether a person's income is over $50,000 a year.

Numerical Features	Categorical Features	Binary Feature
age, fnlwgt, education-num, capital-gain, capital-loss, hours-per-week	workclass, education, marital-status, occupation, relationship, race, native-country	income[target], sex

3.2 Handling Missing Values

Before implementing the ColumnTransformer, the dataset has to be understood by doing Exploratory Data Analysis [EDA] and finding out there were some null values. Since these values were less than 10% in the columns workclass, occupation, and native country they were removed.

3.3 Data Preprocessing:

The dataset has been split into X and Y. X containing all the columns both numerical and categorical except the target variable, y contains only the target variable [Income]

Two variables were created to split X further into numerical columns and categorical columns for applying different preprocessing methods.

3.4 Column Transformer:

Each of these combinations would loop through SVM and Logistic Regression for evaluation and the obtained results would be compared.

Preprocessing Combination	Numerical Columns	Categorical Columns
1	StandardScaler	OneHotEncoding
2	MinMaxScaler	OneHotEncoding
3	Passthrough	OneHotEncoding

Table 1: Preprocessing plan for Numerical and Categorical Columns in the Adult Income dataset

3.5 Pipeline

Since the impact of the methods are more visible on preprocessing dependent on ML algorithms. Support Vector Machine and Logistic Regression were chosen for the pipeline. A list is initialized to store the metrics data to store the outcome of the results.

A loop is created to go through each preprocessing combination and evaluate it with the selected algorithms and the results are appended to the initialized list for comparison and conclusion.

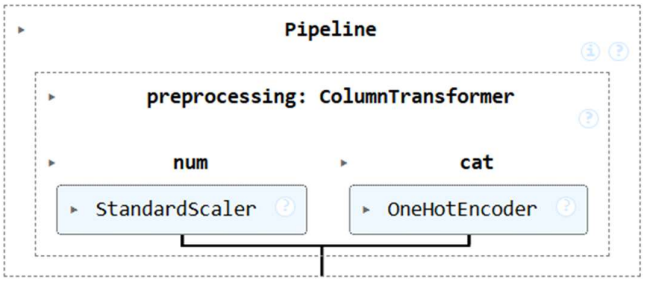


Fig.1. Pipeline Structure for StandardScaler +OneHotEncoder

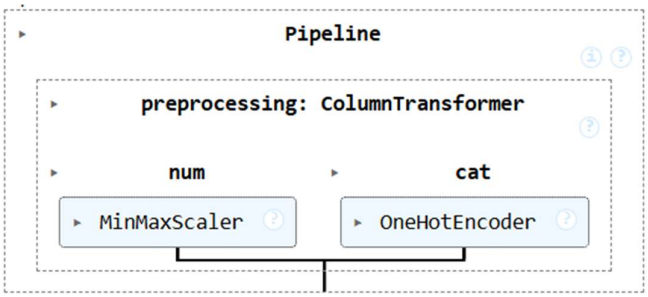


Fig.2. Pipeline Structure for MinMaxScaler+OneHotEncoder

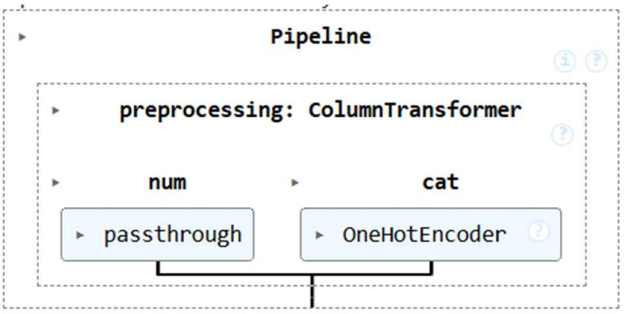


Fig.3. Pipeline Structure for only OneHotEncoder

4. Results & Analysis

For Binary Classification, performance of the pipeline is better known by using accuracy score, Precision, Recall and F1-Score.

4.1 Confusion Matrix:

For binary classification, the confusion matrix divides the test samples into four categories, depending on their true and predicted labels: True Positive, True Negative, False Positive, False Negative. [3]

True Positive: The person actually earns less than 50K per year (0), and the model classifies the same (0).

True Negative: The person actually earns more than 50k per year (1), and the model classifies the same (1).

False Positive: The person actually earns more than 50k per year (1), and the model classifies different (0).

False Negative: The person actually earns less than 50k per year (0), and the model classifies different (1).

4.2 Support Vector Machine

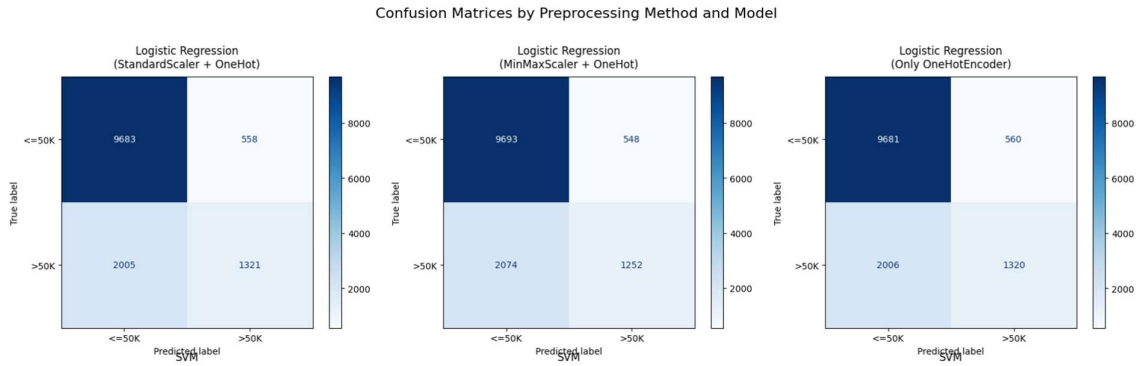


Figure 1; The confusion matrix for Logistic regression over various preprocessing steps

4.3 Logistic Regression

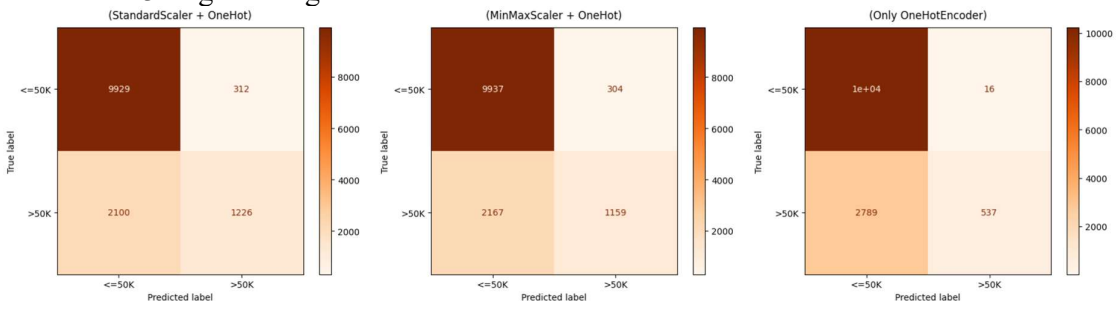


Figure 2: The confusion matrix for Support Vector Machine over various preprocessing steps

4.3 Accuracy

Accuracy score is calculating the ratio of correctly classified instances to the total number of instances.

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Predictions}$$

4.4 Precision

Precision is the ratio of the true positive over all the positives observed. All the positives here include the true positives observed, plus some true negatives wrongly observed as positives (FP). [3]

$$Precision = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Positive(FP)}$$

4.5 Recall

Recall, also known as sensitivity or the true positive rate, is the ability of a model to correctly identify the true positive.

$$Recall = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Negative(FN)}$$

4.6 F1-Score

The F1-Score is the harmonic mean or the weighted average of precision and recall. By definition, it is a measure of a test's accuracy and calculated from the precision and recall of the test. [4]

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

4.7Metrics Summary

4.7.1 Logistic Regression

Preprocessing Method	Model	Precision	Recall	F1-Score	Accuracy
StandardScalar+OneHot Encoding	Logistic Regression	0.701042	0.393567	0.504120	0.819286
MinMaxScaler+OneHot Encoding	Logistic Regression	0.697662	0.375146	0.487925	0.816215
Only OneHotEncoding	Logistic Regression	0.701407	0.393567	0.504214	0.819354

Table 2: Precision, Recall, F1-Score and accuracy for Logistic Regression over various preprocessing Methods

4.7.2 Support Vector Machine

Preprocessing Method	Model	Precision	Recall	F1-Score	Accuracy
StandardScalar+OneHot Encoding	SVM	0.772808	0.383918	0.512991	0.829864

MinMaxScaler+OneHot Encoding	SVM	0.781529	0.358772	0.491784	0.826930
Only OneHotEncoding	SVM	0.955519	0.169591	0.288056	0.804340

Table 3: Precision, Recall, F1-Score and accuracy for Support Vector Machine over various preprocessing Methods

5. Conclusion

This experimental outcome demonstrates extreme variations in model performance depending on preprocessing combinations. Logistic Regression obtained its best accuracy (81.93%) and F1-score (0.504) with StandardScaler + OneHot Encoding, although recall was still low (0.394), showing continued difficulty in detecting high-income individuals. While MinMaxScaler decreased performance slightly (F1: 0.488), excluding scaling altogether (Only OneHotEncoding) had little effect on Logistic Regression, indicating natural robustness against unscaled features. SVM showed abrupt sensitivity to preprocessing, however. While StandardScaler + OneHot produced the best results (accuracy: 82.99%, F1: 0.513), OneHot-only preprocessing led to the disaster of catastrophic failure (recall: 0.170, F1: 0.288), highlighting SVM's feature scaling dependency. Both models prefered precision over recall (SVM precision: 0.773–0.782; Logistic Regression: 0.697–0.701), which proves to conservative predictions sacrificing false positives for missing true high-income cases.

5.1 Key Takeaway

For High precision Tasks SVM+StandardScalar is the optimal choice

For Balanced Performance Logistic Regression+ Standard Scalar is preferred

Scaling plays a critical role on both the algorithms, especially for SVM

Recommendations

- **Feature Engineering:**
  - Creation of interaction terms (e.g., age × education).
  - Bin numerical features to capture income thresholds better.
- **Model Optimization**:
  - Tune hyperparameters (e.g., regularization for LR, kernel selection for SVM).
  - Test ensemble methods (Random Forests, Gradient Boosting) for mixed data.
- **Preprocessing Improvements**:
  - Use target encoding for high-cardinality categorical features.
  - Apply robust scaling for outlier-prone numerical features.
- **Domain-Specific Adjustments**:
  - Implement cost-sensitive learning to reduce false negatives (>50K cases).
  - Explore ethical constraints (e.g., minimizing bias in income prediction).

References

[1] K. Maharana, S. Mondal and B. Nemade, "Data pre-processing and data augmentation techniques", Global Transitions Proceedings, Vol. 3, No. 1, pp. 91-99, 2022.

[2] UCI Machine Learning Repository. (1996). *Adult Dataset* [Dataset]. Available at: https://doi.org/10.24432/C5XW20. Accessed on [07][05] [2025]

[3] Varoquaux, G., Colliot, O. (2023). Evaluating Machine Learning Models and Their Diagnostic Value. In: Colliot, O. (eds) Machine Learning for Brain Disorders. Neuromethods, vol 197. Humana, New York, NY. https://doi.org/10.1007/978-1-0716-3195-9_20

[4] Obi, J.C., 2023. A comparative study of several classification metrics and their performances on data. *World Journal of Advanced Engineering Technology and Sciences*, *8*(1), pp.308-314.

[5] Werner de Vargas, V., Schneider Aranda, J.A., dos Santos Costa, R., da Silva Pereira, P.R. and Victória Barbosa, J.L., 2023. Imbalanced data preprocessing techniques for machine learning: a systematic mapping study. *Knowledge and information systems*, *65*(1), pp.31-57.

[6] Chahid, I., Elmiad, A.K. and Badaoui, M., 2023, November. Data preprocessing for machine learning applications in healthcare: a review. In *2023 14th International Conference on Intelligent Systems: Theories and Applications (SITA)* (pp. 1-6). IEEE.

[7] Wu, D., 2024. The effects of data preprocessing on probability of default model fairness. arXiv preprint arXiv:2408.15452.

[8] Artificial intelligence (AI) vs. machine learning (ML) https://cloud.google.com/learn/artificial-intelligence-vs-machine-learning