Grenze International Journal of Engineering and Technology, June Issue

Intuitive Model Development and Data Preprocessing with Web and Command-Line Interfaces

Sri Charani P[1], Sri Krishna Adusumalli[2], Kalyani Kaligotla[3], Rishitha Ramadurgam[4], Ramyasri Sydu[5], Yamini Devi Balam[6] and Bhagya Shanmukhi Tavvala[7]
[1-7]Department of Artificial Intelligence, Shri Vishnu Engineering College for Women, Bhimavaram, Andhra Pradesh, India.
Email: charani.yashu@svecw.edu.in, {http://www.orcid.org/0000-0002-9134-011X, kalyanikaligotla22@gmail.com, rishitharamadurgam.0403@gmail.com, ramyasrisydu@gmail.com, yaminibalam182003@gmail.com, holytavvala@gmail.com}

***Abstract*— Effective machine learning models hinge upon meticulous data preprocessing for optimal performance. This study introduces a versatile data preprocessing tool developed within the Flask framework, a Python-based web framework. The application not only streamlines conventional preprocessing tasks for structured datasets but also addresses a significant gap in the preprocessing landscape by accommodating qualitative data. Users can seamlessly interact with the program through a web-based interface or command-line input to upload datasets, perform operations such as column deletion, handle missing values, compress categories, and standardize or normalize facts. The versatility of the tool, coupled with its accessibility and comprehensive functionality, positions it as a valuable asset for researchers and practitioners in the machine learning community.**

Moreover, the tool integrates popular machine learning algorithms such as linear regression, decision trees, random forest, k-nearest neighbours (KNN), logistic regression, and support vector machines (SVM). This inclusive approach, coupled with a no-code environment, empowers users of varying technical backgrounds to effortlessly prepare datasets for machine learning endeavours. The tool's adaptability extends to incorporating these algorithms into the preprocessing workflow, allowing users to perform model training and evaluation seamlessly within the same environment.

***Index Terms*— Machine Learning, Web Interface, Command-Line Interface, Data-Preprocessing, No-Code Environment, Evaluation Metrics, Machine Learning Pipeline.**

I. INTRODUCTION

Introducing a comprehensive data preprocessing tool built on Flask, a flexible Python web framework. This tool tackles nuanced challenges in qualitative data processing, offering a user-friendly web interface for CSV uploads and a command line interface for technical users. Furthermore, our tool emphasizes a no-code environment, ensuring that users can perform complex preprocessing tasks without the need for extensive coding knowledge. This no-code approach democratizes the data preparation process, making machine learning more accessible to a broader audience.

Our Flask-based data preprocessing tool goes beyond traditional approaches by seamlessly integrating qualitative data handling, providing a command line interface, embracing a no-code environment, and incorporating popular machine learning algorithms. By addressing these multifaceted aspects of data preprocessing, this tool aims to

Grenze ID: 01.GIJET.10.2.89

make machine learning more inclusive, empowering users to efficiently prepare datasets for diverse machine learning task

Data preprocessing encompasses various tasks, including cleaning, encoding, scaling, and dimensionality reduction. The resolution of data issues, commonly known as data cleaning, targets problems such as missing values, inaccurate data types, and outliers. The primary goal of data cleaning is to rectify issues related to missing values, inconsistencies, and noisy data. Additionally, data preprocessing involves operations like feature encoding, scaling, and dimensionality reduction.

The paper references works by Ahsan et al. (2021), Brownlee (2020), LeDell and Poirier (2020), and Escalante (2020), providing insights into data preprocessing and automated machine learning.By integrating methodologies from these works, the paper contributes to advancing automated data preprocessing techniques and fostering the adoption of machine learning in diverse applications.

II. METHODOLOGY

The research begins by identifying common preprocessing tasks and breaking them down into practical components. Modular scripts are then developed using tools like scikit-learn, Pandas, and shell scripting for versatility across datasets. Both a web interface and a command-line interface (CLI) are created to cater to user preferences.

The research employs an iterative methodology focused on user needs and continual improvement. The web interface prioritizes usability and security with robust authentication and authorization mechanisms. Asynchronous processing handles large datasets efficiently, providing timely task updates. Logging and monitoring functionalities offer insights into preprocessing pipeline efficiency. Rigorous testing validates reliability across various scenarios.

A significant challenge encountered during development was devising a method to align each data preprocessing step with its corresponding transformer method to achieve our objectives. While one approach might involve exhaustively evaluating all permutations of techniques on datasets to identify the optimal combination, such a method proves computationally intensive and time-consuming.

A. Datatype Recognition Engine

The Datatype Recognition Engine serves a crucial role in enabling effective data-driven analysis and computation by accurately identifying and categorizing data types, including both basic and statistical types. While leveraging the Pandas library's capabilities for automatic detection of general datatypes, it's essential to address its limitations in inferring complex data types beyond integers, floats, and objects.

To address these limitations, a two-step approach is proposed. Initially, Pandas' simple inference method identifies basic data types like integer, float, and object. Then, a more comprehensive reassessment of object-type features refines the datatype inference process, accurately categorizing Boolean, categorical, and string data types. Boolean data represents binary states crucial for logical operations. Categorical data lacks natural numerical order and is measured in categories or classes. String data represents sequences of characters, fundamental for textual information storage. The Datatype Recognition Engine combines logical principles with Pandas' functionality for automated and efficient preprocessing, enhancing data analysis pipeline reliability and usability.

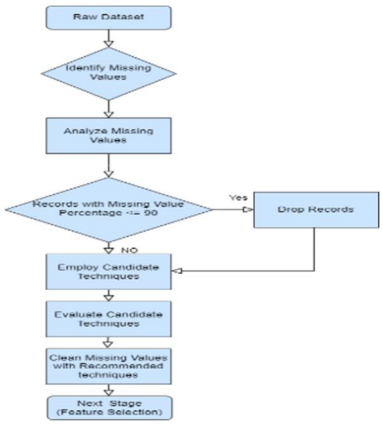

FIG. 1 Workflow of Handling Missing Values

B. Intutive Missing Value Handling

Handling missing values in a dataset involves a multi-step process that begins with their identification and culminates in the selection of an applicable imputation method. Extensive literature indicates that no single algorithm widely outperforms others, as the effectiveness of an approach is heavily influenced by the characteristics of the missing data.

Accordingly, we break down this task into distinct subtasks, including the discovery of missing values, elimination of features with a high incidence of missing data, imputation of missing values using various techniques, and the ultimate selection of the most suitable imputation method.

The workflow, illustrated in the accompanying figure fig-1, commences with the identification of missing values. Subsequently, we evaluate a range of candidate imputation techniques to determine their efficacy, ultimately recommending the best-performing method. This structured approach ensures a systematic and user-friendly process for handling missing values in the dataset.

1. Identification of Missing Values

Detecting missing data presents an initial challenge in automating data handling processes. These missing values can manifest in diverse formats within datasets, complicating the detection process. To provide a comprehensive approach, we have categorized these representations into three distinct types: (1) Standard Missing Values, which include null as 'NAN,' 'n/a,' and empty cells; (2) Non-Standard Representations, encompassing symbols like ',', '-', 'na,' or '?'.

2. Missing Mechanism

In the realm of missing data, three main mechanisms are recognized: Missing at Random (MAR), Missing Completely at Random (MCAR), and Missing Not at Random (MNAR). Testing for MNAR is often unfeasible due to data limitations. While it's speculated that missingness may be influenced by certain values, such as higher salaries leading to less reported incomes, this remains unverified. However, exploring associations between missing values and other features is feasible for MAR and MCAR. MAR implies randomness in missingness given observed data, allowing for unbiased imputation. MCAR denotes randomness in missingness unrelated to any data, enabling unbiased statistical methods and imputation techniques, pending confirmation through appropriate tests.

3. Maximum Allowed Percentage of Missing Values

We may notice that the values of particular records or features are mainly absent after detecting missing values in the dataset.

4. Missing Value Imputation

Missing Values are a common challenge in data analysis, occurring when information for a specific variable is absent for some observations in your dataset. This can happen due to various reasons, like incomplete data entry, technical errors, or participant dropout in surveys. To overcome this challenge and utilize the full potential of your data, a technique called missing value imputation comes into play. Imputation essentially involves replacing missing values with plausible estimates, allowing you to analyse the complete dataset effectively. Different techniques can be used, depending on the nature of data and the missing values themselves. We have considered MCAR for mean, median and mode

C. Automatic Qualitative Features Encoding

After addressing missing values, the subsequent step involves encoding qualitative features within the dataset. In machine learning models, all input and output variables must be represented numerically. Therefore, any qualitative data present in the dataset necessitates conversion into numeric form. Qualitative data is typically categorized as Boolean, string, or categorical. Moreover, within categorical data, distinctions are made between nominal and ordinal attributes.

Nominal attributes pertain to features whose values denote categories or labels without inherent order or quantitative significance. On the other hand, ordinal attributes represent categories with a natural order or ranking, but the distances between these categories are either unknown or lack meaningful interpretation.

To encode categorical data, two primary techniques are commonly employed: label encoding and one-hot encoding. These methods facilitate the transformation of categorical variables into a numerical format suitable for machine learning algorithms. Label encoding assigns a unique numeric label to each category within a feature, while one-hot encoding creates binary columns to represent the presence or absence of each category, effectively avoiding ordinal assumptions associated with label encoding.

FIG. 2 Workflow of Feature Encoding

III. Experimental Results

A. Datasets

Our dataset selection process aims to accommodate diverse data preparation needs by intentionally selecting datasets from multiple platforms, including UCI, OpenML, and Kaggle. We prioritize commonly used datasets, considering factors like variety, size. Specifically, we curate four datasets for regression tasks and an additional four for classification activities, allowing for a thorough examination across a range of predictive modelling situations

TABLE I. DETAILS OF THE DATASETS

NAME OF THE DATASET	NUMBER OF ATTRIBUTES	DATASET SIZE	TYPE OF ML TASK (REGRESSION/CLASSIFICATION)
BOOKS DATASET	15	1070	REGRESSION
FLIGHT PRICE PREDICTION	12	300153	REGRESSION
INSURANCE	7	1338	REGRESSION
RESTAURANT REVENUE PREDICTION	8	1000	REGRESSION
BREAST CANCER	16	4024	BINARY CLASSIFICATION
CREDIT CARD DEFAULTER PREDICTION	25	8000	BINARY CLASSIFICATION
TITANIC	12	891	BINARY CLASSIFICATION

B. Machine Learning Algorithms

Our major goal is to train a broad selection of machine learning models on the datasets and then rigorously evaluate their performance using a wide range of metrics suited to each dataset, such as accuracy, precision, and mean square error. Given our emphasis on practical outcomes rather than digging into the specifics of the models, we purposely utilize simple and well-established machine learning techniques for both classification and regression tasks.

C. Results of Preprocessing and Model Building tasks in web interface

The Web Interface Layout using Flask is showed in the FIG.3.

D. Classification Performance Evaluation

Confusion Matrix: A confusion matrix summarises a classification algorithm's performance. It displays the number of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) forecasts.

TABLE II. CONFUSION MATRIX

Predicted /Actual Class	1	0
1	True Positive	False Negative
0	False Positive	True Negative

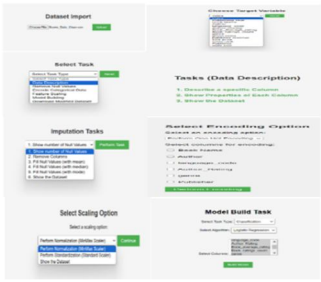

FIG. 3 Web Interface Results

E. Classification Outcomes

This investigation pre-processes four categorization datasets (Breast Cancer Detection, Titanic Survival Prediction, Credit Card Default Prediction, and Pizza Brand Classification), each presenting different challenges. Various machine learning algorithms, including Logistic Regression, Decision Tree, Random Forest Classifier, K-Nearest Neighbour Classifier, and Support Vector Machine, are evaluated on auto-preprocessed data. The performance metrics, such as accuracy, precision, and recall, are analysed via confusion matrices, aiding in determining the most effective technique for accurate predictions. These findings support the study's objectives. The results are shown in TABLE III – V, Fig 4 – 23

TABLE III. LOGISTIC REGRESSION AND DECISION TREE RESULTS OF EACH DATASET

Dataset	ML ALGORITHM							
	LOGISTIC REGRESSION				DECISION TREE			
	ACCURACY	PRECISION	RECALL	F1 SCORE	ACCURACY	PRECISION	RECALL	F1 SCORE
Breast Cancer Detection	90.31	89.69	90.31	89.09	86.83	87.11	86.83	86.96
Titanic	78.77	78.65	78.77	78.62	79.89	79.89	79.89	79.89
Credit Card Defaulter Prediction	78.12	61.02	78.12	68.52	72.07	72.92	72.07	72.47
Pizza	83.33	89.05	83.33	85.13	90.00	92.13	90.00	90.59

TABLE IV. RANDOM FOREST AND KNN RESULTS OF EACH DATASET

Dataset	ML ALGORITHM							
	RANDOM FOREST				KNN			
	ACCURACY	PRECISION	RECALL	F1 SCORE	ACCURACY	PRECISION	RECALL	F1 SCORE
BREAST CANCER DETECTION	91.43	90.94	91.43	90.62	89.19	88.16	89.19	88.24
Titanic	82.68	82.82	82.68	81.42	66.48	66.02	66.48	64.96
Credit Card Defaulter Prediction	81.77	79.91	81.77	79.64	75.08	70.26	75.08	71.71
Pizza	96.67	96.99	96.67	96.41	91.67	93.00	91.67	92.30

TABLE V. SVC RESULTS OF EACH DATASET

Dataset	ML ALGORITHM			
	SVC			
	ACCURACY	PRECISION	RECALL	F1 SCORE
BREAST CANCER DETECTION	89.57	89.02	89.57	87.83
Titanic	67.04	73.30	67.04	61.17
Credit Card Defaulter Prediction	75.08	70.26	75.08	71.71
Pizza	65.00	65.00	65.00	59.45

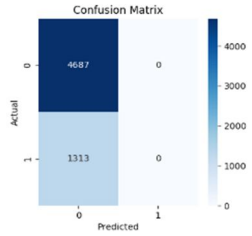

Fig. 4 confusion matrix of credit card defaulter prediction using logistic

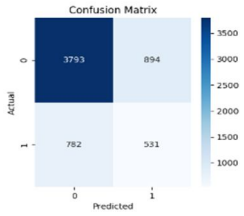

Fig. 5 confusion matrix of credit card defaulter prediction using decision tree

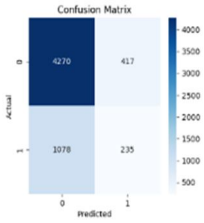

Fig. 6 confusion matrix of credit card defaulter prediction using knn

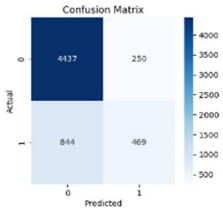

Fig. 7 confusion matrix of credit card defaulter prediction using random forest

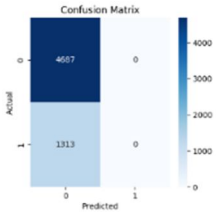

Fig. 8 confusion matrix of credit card defaulter prediction using svc

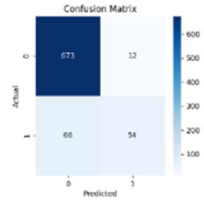

Fig. 9 confusion matrix of breast cancer detection using logistic regression

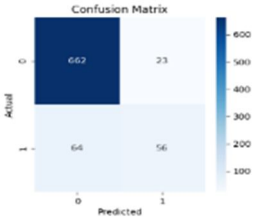

Fig. 10 confusion matrix of breast cancer detection using decision tree

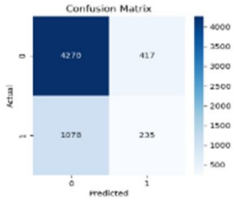

Fig. 11 confusion matrix of breast cancer detection using knn

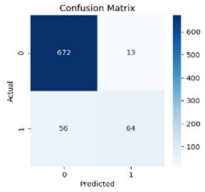

Fig. 12 confusion matrix of breast cancer detection using random forest

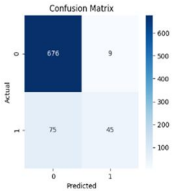

Fig. 13 confusion matrix of breast cancer detection using svc

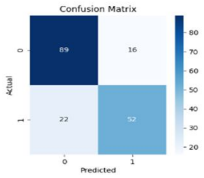

Fig.14 confusion matrix of titanic using logistic regression

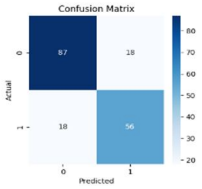

Fig. 15 confusion matrix of titanic using decision tree

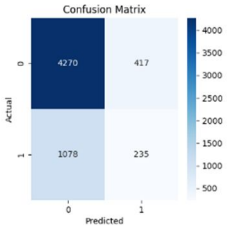

Fig. 16 confusion matrix of titanic using knn

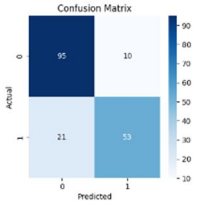

Fig. 17 confusion matrix of titanic using random forest

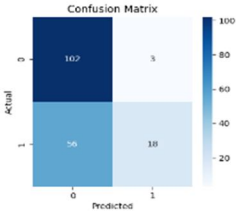

Fig.18 confusion matrix of titanic using svc

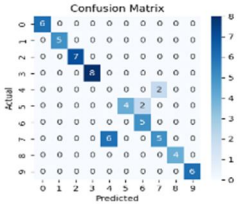

Fig. 19 confusion matrix of pizza dataset using logistic

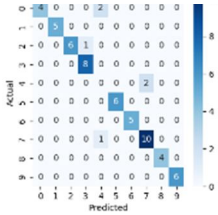

Fig. 20 confusion matrix of pizza dataset using decision tree

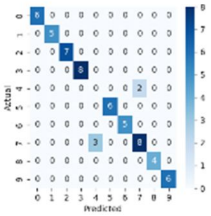

Fig. 21 confusion matrix of pizza dataset using knn

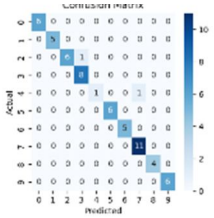

Fig. 22 confusion matrix of pizza dataset using random

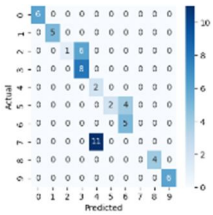

Fig. 23 confusion matrix of pizza dataset using svc

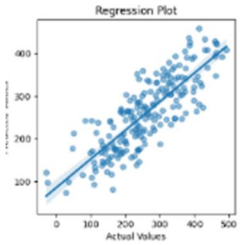

Fig. 24 scatter plot of restaurant revenue

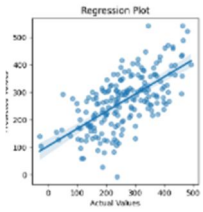

Fig. 25 scatter plot of restaurant

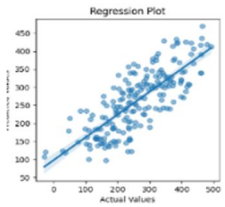

Fig. 26 scatter plot of restaurant

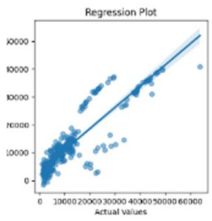

Fig. 27 scatter plot of insurance dataset

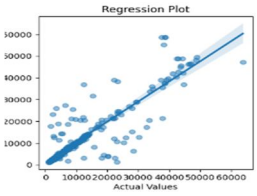

Fig. 28 scatter plot of insurance dataset using decision tree

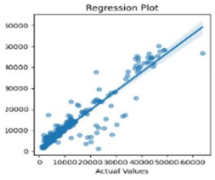

Fig. 29 scatter plot of insurance dataset using random forest

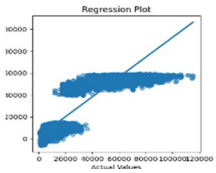

Fig. 30 scatter plot of flight price prediction using

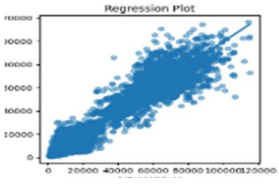

Fig. 31 scatter plot of flight price prediction using decision tree

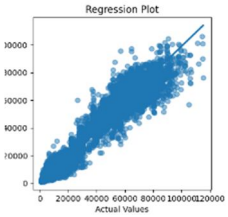

Fig. 32 scatter plot of flight price prediction using random forest

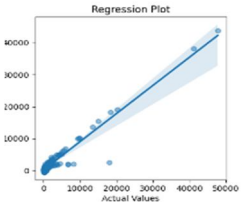

Fig. 33 scatter plot of books dataset using linear regression

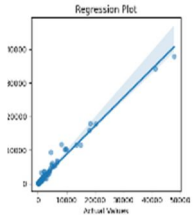

Fig. 34 scatter plot of books dataset using decision tree

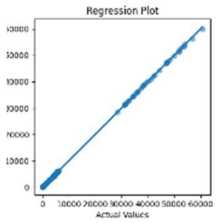

Fig. 35 scatter plot of books dataset using random forest

F. Regression Performance Evaluation

$$\textit{Mean squared error (MSE):}\ \mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \qquad (1)$$

$$\textit{Mean Absolute Error (MAE):}\ \mathrm{MAE} = \frac{\sum_{i=1}^{n}|Y_i - \hat{Y}_i|}{n} \qquad (2)$$

$$\textit{Root Mean Squared Error (RMSE):}\ \mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{n}} \qquad (3)$$

$$\textit{The R-squared (R2) Score:}\ R^2 = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}} \qquad (4)$$

R^2 = Coefficient of determination
RSS = sum of squares of residuals, TSS = Total sum of squares

G. Regression Outcomes

Similarly, to automate preprocessing steps, both the web interface and CLI are applied to four distinct Regression datasets with varying attributes, data quality, and preprocessing requirements. These datasets undergo analysis using linear regression, decision tree, and random forest regressor models via both interfaces. Scatter plots with regression lines summarize dataset trends, demonstrating the approach's success. Random Forest proves accurate for flight price prediction and adept at handling complexity in restaurant revenue prediction, while MLP excels in

restaurant prediction by MAE and MSE. Despite Random Forest's general effectiveness, MLP's success on less complex tasks suggests a case-specific approach may be optimal.The proposed approach streamlines preprocessing, enhancing model performance, though challenges persist with outliers and imbalanced data, guiding potential future improvements. The results are shown in Table VI, Fig 24-35.

TABLE VI. LINEAR REGRESSION, DECISION TREE AND RANDOM FOREST RESULTS FOR EACH DATASET

Dataset	ML Algorithm											
	Linear Regression				Decision Tree				Random Forest			
	MAE	MSE	R2 SCORE	RMSE	MAE	MSE	R2 SCORE	RMSE	MAE	MSE	R2 SCORE	RMSE
Flight Price Prediction	4622.19	7013.56	0.9	83.75	890.24	2974.21	0.98	54.54	862.33	2319.59	0.99	48.16
Books Dataset	639.36	1414.05	0.92	37.6	256.12	1025	0.96	32.02	256.12	1025	0.96	32.02
Restaurant Revenue Prediction	47.15	59.66	0.67	7.72	70.91	87.22	0.30	9.34	52.75	64.77	0.62	8.05
Insurance	4186.5	5799.59	0.78	76.16	2969.54	6495.18	0.73	80.59	2466.04	4548.41	0.87	67.44

IV. CONCLUSIONS

We have successfully implemented an automated solution for data preprocessing tasks, accessible through both a user-friendly web interface and a Command Line Interface (CLI). The emphasis on a no-code environment is designed to significantly reduce the time required for crucial preprocessing steps in machine learning pipelines. The web interface, in particular, enhances user experience, allowing users to effortlessly perform tasks such as data type detection, null value imputation, categorical feature encoding, and feature scaling.

Our innovative system is not only accessible through a straight forward Command Line Interface but also features a powerful Web Interface, making it exceptionally user-friendly. The latter provides an intuitive platform for users to navigate and execute essential preprocessing tasks with ease. It simplifies complex operations such as datatype detection, null value imputation, categorical feature encoding, and feature scaling, all of which contribute to a streamlined and efficient data preprocessing experience.

We are presently engaged in the development of an automated and interactive system designed to proficiently identify data errors. Our methodology has undergone rigorous testing across four regression and four classification datasets, each characterized by distinctive features. We have utilized a diverse range of effective machine learning models tailored to each dataset to evaluate the efficacy of our approach. The evaluation process entails a meticulous comparison of metrics obtained from models trained on meticulously pre-processed data.

Our compelling findings underscore the significance of our web interface, not only in providing users with a superior experience but also in simplifying the process of training multiple models and rapidly pre-processing data when compared to traditional manual methods. The web interface emerges as a key component in ensuring accessibility, efficiency, and user satisfaction throughout the data preprocessing and machine learning model training phases.

REFERENCES

[1] Mehwish Bilal, Ghulam Ali, Muhammad Waseem Iqbal, Muhammad Anwar, Muhammad Sheraz Arshad Malik, Rabiah Abdul Kadir, “Auto-Prep: Efficient and Automated Data Preprocessing Pipeline” vol. 10, pp. 107764 – 107784, Aug. 2022.

[2] M. Ahsan, M. Mahmud, P. Saha, K. Gupta, and Z. Siddique, ‘‘Effect of data scaling methods on machine learning algorithms and model performance,’’ Technologies, vol. 9, no. 3, p. 52, Jul. 2021.

[3] J. Brownlee, Data Preparation for Machine Learning: Data Cleaning, Feature Selection, and Data Transforms in Python. Melbourn, VIC, Australia:Jason Brownlee, 2020.

[4] H. J. Escalante, "Automated machine learning—A brief review at the end of the early years", arXiv:2008.08516, 2020.

[5] E. LeDell and S. Poirier, H2O AutoML: Scalable Automatic Machine Learning. 2020, pp. 1–16.

[6] P.Sricharani and B. SrinivasaRao. “Estimation of Reliability in Multi Component Stress Strength Based on Dagum Distribution”.AIP Confrencepreceedings, Volume 2707, Issue 19 May 2023
[7] P.Sricharani and B. SrinivasaRao.”Variable control charts based on Dagum distribution”,Research Journal of Mathematical and Statistical Sciences 2019, Vol. 7(3),pp: 43-50
[8] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed,‘‘A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction,’’ J. Appl. Sci. Technol. Trends, vol. 1, no. 2, pp. 56–70, May 2020
[9] X. He, K. Zhao, X. Chu ‘’AutoML: A survey of the state-of-the-art’’ Knowledge-Based Systems, 212 (2021), p. 106622
[10] Rashmi Gandhi , Sparsh Khurana , Harsh Manchanda , " ETL Data Pipeline to Analyze Scraped Data ", Decision Intelligence , vol. 1079 , pp. 379 , 2023
[11] J. T. Hancock and T. M. Khoshgoftaar, ‘‘Survey on categorical data for neural networks,’’ J. Big Data, vol. 7, no. 1, pp. 1–41, Dec. 2020