Efficient Machine Learning Pipeline Automation Using Tpot And Pycaret

N. Arikaran[#1],G. Dharanya[#2], M. Kanchana[#3], B. Poojitha[#4], A. Bhuvanesh[#5], S. Kamalesh[#6] , Arya Ejoumalai[#7]

[#1,2,3,4,5,6,7]manakula Vinayagar Institute Of Technology, Puducherry, India
[1]arikaran.N@Gmail.Com

ABSTRACT
The paper falls under the domain of Automated Machine Learning (AutoML) and Data Preprocessing. The existing system employs the Tree-based Pipeline Optimization Tool (TPOT), a Python-based AutoML framework that automates tasks such as algorithm selection, hyperparameter tuning, and data preprocessing, primarily for regression tasks. However, its functionality is limited to regression-based optimization. To overcome this limitation, the proposed work integrates PyCaret, a more versatile AutoML library that supports both classification and regression. PyCaret enhances the machine learning pipeline with robust preprocessing features, including feature engineering, error handling, and class imbalance management. It enables users to train multiple models and automatically selects the best-performing one, streamlining the entire workflow and making machine learning more accessible. The system achieves an impressive accuracy of 97.8%. Future work may include extending support to unsupervised and deep learning tasks, integrating cloud-based scalability, and incorporating real-time data processing for broader applicability.

Keywords: Auto ML, Data, Preprocessing, Machine Learning, Hyperparameters, Feature selection, Report generation, Data Visualization.

I. INTRODUCTION

The paper explores the complex interplay between Machine Learning (ML) and AutoML, with a focus on their mutual cooperation. ML uses algorithms to find patterns in vast datasets, predict outcomes, and aid in decision-making. AutoML extends this by automating mundane yet crucial tasks that improve the efficiency and accessibility of the ML pipeline.

For better improvement of data preprocessing methods, the survey suggests that an in-depth exploration of research papers and contributions should be undertaken. Such a deep study encompasses a wide range of papers which discuss novel solutions for filling the gap between the present-day challenges and future promises of data preprocessing. The goal is to uncover key insights by fusing data preprocessing intricacy with the potential of AutoML toward advancing decision-making capabilities based on data in an evolving ML environment. In this survey, various research papers have been highlighted, from "DataAssist" to "REIN," as part of relevant studies contributing vital principles in this quest. The papers therefore illuminate the complex terrain in which challenges related to imbalanced data, nuances of hyperparameter optimization, and the advancement of feature engineering converge on themselves to demand holistic solutions that would bridge the divide between data and model.

The advancing frontiers of machine learning draw principles from this research paper; those principles are guidelines that call for crafting automated solutions able to solve challenging problems. Backed by such a comprehensive literature review, the forthcoming architectural overview does promise innovation but instead promises a new status quo for all things - the all-encompassing, end-to-end solutions of data preprocessing and the automation of pivotal tasks. Principles for this survey lie in problem identification and seeking innovative solutions between the existing problem and the promise of the future-which is the preprocessing of data. Every paper on the list focuses on a particular aspect of preprocessing data, putting it together for the understanding of a domain of extreme importance.

II. RELATED WORK

ParijatDube; TheodorosSalonidis[1]DataAssist seems like a promising addition to the AutoML landscape, as it is focused on data preparation and cleaning. This is a very important aspect of the machine learning workflow that most of the existing tools overlook. The key features of DataAssist seem to meet the needs of data scientists and analysts, mainly in industries where quality of data matters, such as economics, business, and forecasting. It streamlines the process of preparing data for modeling by giving functionalities for exploratory data analysis, visualization, anomaly detection, and preprocessing, potentially saving large amounts of time and effort on the part of practitioners. Moreover, the possibility of exporting cleaned and preprocessed datasets for integration with other AutoML tools or user-specified models increases its versatility and interoperability within existing workflows. This flexibility is important to accommodate different preferences and requirements in data analysis pipelines. Overall, DataAssist seems to fill a significant gap in the existing landscape of AutoML tools by prioritizing data-centric tasks and offering comprehensive support for data preparation and cleaning. The time savings amounting to over 50% of what typically is expended by practitioners across all domains underline the value proposition of this solution.

Sagnik Mukherjee; YerramreddySrinivasa Rao [2]DataAssist has a promising sound to it in this landscape of automated machine learning tools. It addresses the most neglected aspect in the machine learning workflow, which is data preparation and cleaning. The key characteristics of DataAssist appear to be in sync with the demands of data scientists and analysts, mainly in sectors in which data quality is of critical importance, including economics, business, and forecasting. DataAssist provides functionalities related to exploratory data analysis, visualization, anomaly detection, and data preprocessing in order to streamline preparing data for modeling, which would save considerable amounts of time and effort for the practitioner. Moreover, the export of cleaned and preprocessed datasets integrates with other AutoML tools or user-specified models to enhance its versatility and interoperability within existing workflows. The flexibility it offers is crucial for accommodating different preferences and requirements in data analysis pipelines. Overall, DataAssist somehow fills a significant gap in the existing landscape of AutoML tools by putting data-centric tasks at the forefront, providing full support for the preparation and cleansing of datasets. Its potential to save over 50% of the time usually spent on these tasks underlines its value proposition for practitioners across domains.

Henrique Pedro Ribeiroa and Patryk Orzechowski [3] This paper explores the growing landscape of The popularity of machine learning automated (AutoML) programs can be attributed to their great performance and versatility in solving a wide range of issues. The challenge lies in choosing the most suitable AutoML algorithm for a given problem amid the increasing options available. To address this, the study examines the output of four well-known AutoML algorithms using their Diverse and generative ML benchmarking (DIGEN): Auto-Sklearn, H2O AutoML, Auto-Sklearn 2, and Tree-based Pipelines Optimizing Tool (TPOT). Synthetic datasets called DIGEN are used to demonstrate the advantages and disadvantages of popular machine learning algorithms. The outcomes demonstrate how successfully AutoML detects pipelines across datasets. While the majority of AutoML algorithms demonstrated similar performance, subtle differences emerged based on specific datasets and evaluation metrics, providing valuable insights into their comparative effectiveness.

Christian Hammacher, Harald Schoening, and Mohamed Abdelaal [4]The article highlights the importance of machine learning(ML) in everyday life and underscores how vital good-quality data is at every stage of the ML application life cycle. It recognizes the typical discrepancies in real-world tabular data, such as inconsistencies, duplication, outliers, missing values, and pattern violations, which frequently occur during data gathering, transmission, storage, or consolidation. Despite numerous data cleaning methods addressing these issues, the paper points out a gap in considering downstream ML model requirements. To bridge this gap, the work introduces a comprehensive benchmark named REIN1, aiming must carefully evaluate how various ML models are affected by data cleaning techniques. The benchmark addresses key research questions, exploring the necessity and efficacy of data cleaning in ML pipelines. The evaluation involves 38 error detection and repair methods, ranging from simple to advanced. To provide comprehensive insights, the benchmark employs a broad range of machine learning models that were trained on 14 publicly-accessible datasets that span multiple domains and include both synthetic and realistic error characteristics.

Ahmad Schoening, Harald Schoening, and Rashmi Koparde[5] The paper introduces Data curation pipeline AutoCure is innovative and requires no setting designed to address the persistent challenge of

data preparation in machine learning applications across domains like autonomous driving, healthcare, and finance. The need for expert knowledge and considerable time investment in navigating the extensive search space for suitable data curation and transformation tools is a recognized hurdle in model development. AutoCure stands out by synthetically enhancing the clean data fraction by combining a data augmentation module with an inventive adjustable ensemble-based error detection technique. Notably, its configuration-free nature streamlines the implementation process, making it accessible for integration using free and open-source resources such as Auto-sklearn, H2O, and TPOT, therefore advancing the general democratization of machine learning.

Holzer and Stockinger, Kurt[6] This paper presents an innovative architecture leveraging bidirectional recurrent neural networks for the purpose of error detection in databases. Through experiments conducted on six distinct datasets, The outcomes demonstrate how well this strategy performs in comparison to cutting-edge mistake detection technologies. Specifically, the average F1-scores across all datasets demonstrate the effectiveness of the proposed architecture. Notably, the system exhibits a lower standard deviation, indicating greater robustness compared to existing methods. An additional advantage is the system's ability to achieve high F1-scores without the need for supplementary data augmentation techniques. This signifies the potential of the introduced bidirectional recurrent neural network architecture as a robust and efficient solution for error detection in diverse database scenarios.

III. MATERIAL AND METHODS

The research paper aims to explore the intricate details of the AutoML system, providing an in-depth analysis of its capabilities, experimental results, and its potential to revolutionize the field of machine learning. With a particular focus on addressing the shortcomings in existing data preprocessing methodologies, the system is positioned as a promising approach to enhancing datasets and subsequently improving findings across diverse domains. The paper likely delves into the system's innovative features, experimental validations, and how it contributes to overcoming challenges in data preprocessing, ultimately paving the way for more effective and efficient machine learning applications. The emphasis on improving datasets suggests a commitment to elevating the overall quality of input data, It is essential to machine learning models' ability to succeed.

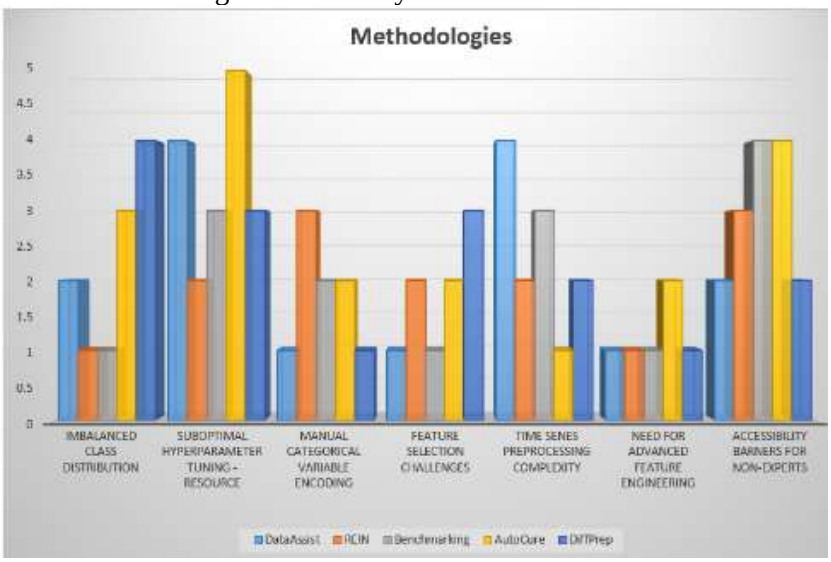


Fig 1: Methodologies

IV. PROPOSED METHOD AND TECHNIQUES USED

The proposed system makes use of PyCaret, a Python-based AutoML pipeline, to upgrade machine learning workflows by supporting classification and regression algorithms. PyCaret offers an all-inclusive suite of preprocessing tools, which include feature engineering, error handling, and class imbalances management. This means the datasets are properly prepared for the best model performance. It enables training multiple machine learning algorithms at the same time, and automatically determines the best performing model for the given dataset. As with support of over 15 algorithms ranging from traditional linear models to latest ensemble techniques, PyCaret is flexible and broadens the scope of

model experimentation. The interface as well as design are easily accessible for experts and novices alike. By simplifying the pipeline of machine learning, streamlining experimentation, and optimizing algorithm selection, PyCaret makes machine learning more efficient, effective, and very accessible for a wide range of tasks and levels of user expertise.

A. **Dataset**

The dataset utilized in this study serves as the foundation for developing and evaluating the machine learning models implemented through AutoML frameworks. It comprises structured tabular data featuring multiple attributes relevant to the chosen predictive task. The dataset undergoes preprocessing steps, including handling missing values, normalization, and encoding categorical variables, ensuring optimal input for the model training phase. Feature selection techniques are applied to identify the most significant variables contributing to accurate predictions, thereby improving model efficiency and interpretability. The dataset is divided into training and testing subsets, allowing for rigorous model evaluation and performance validation. Additionally, exploratory data analysis (EDA) is conducted to uncover patterns, correlations, and potential anomalies within the data. The processed dataset is then utilized by AutoML tools such as PyCaret, TPOT, Auto-Sklearn, and Google AutoML, facilitating automated model selection, hyperparameter tuning, and optimization. By leveraging this dataset, the study aims to assess the effectiveness of different AutoML approaches in generating high-performing predictive models while minimizing manual intervention.

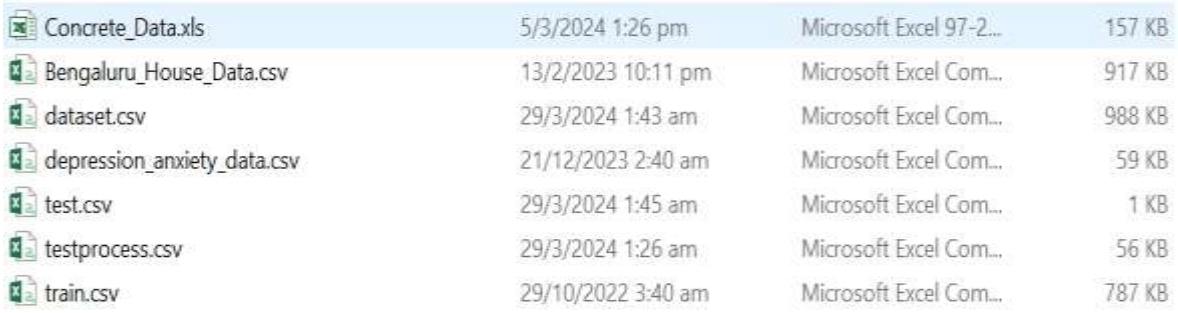
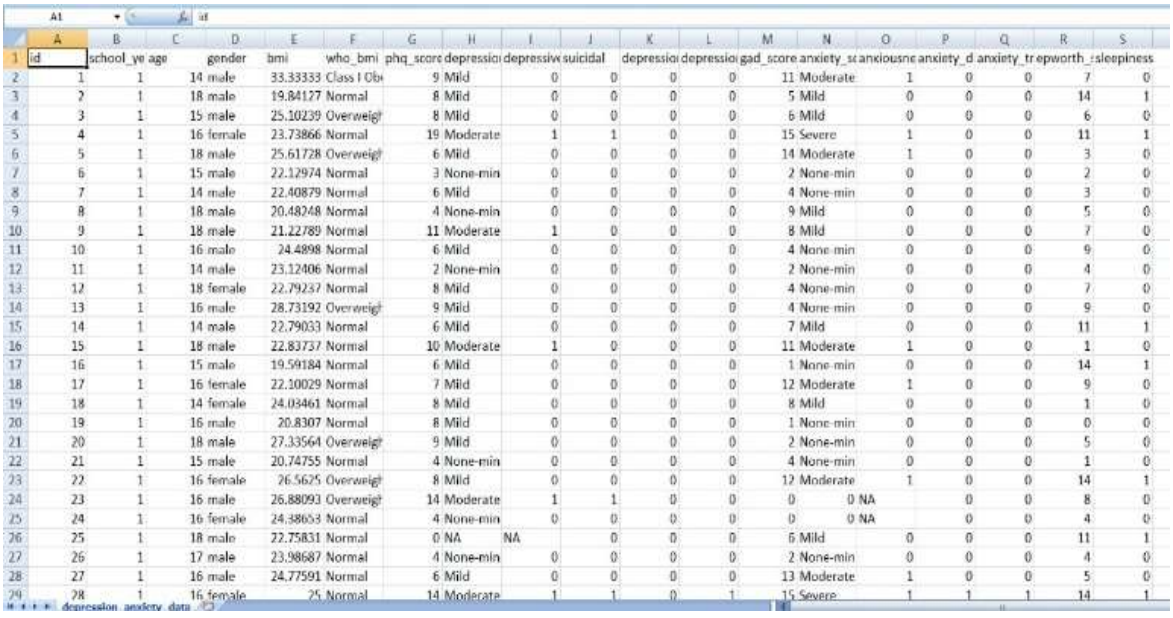


Fig 2: Available Datasets for AutoML Implementation



id	school_ye	age	gender	bmi	who_bmi	phq_score	depressiv	depressiv	suicidal	depressio	depressio	gad_score	anxiety_s	anxiousne	anxiety_d	anxiety_tr	epworth_	sleepiness
1	1	14	male	33.33333	Class I Ob	9	Mild	0	0	0	0	11	Moderate	1	0	0	7	0
2	1	18	male	19.84127	Normal	8	Mild	0	0	0	0	5	Mild	0	0	0	14	1
3	1	15	male	25.10239	Overweig	8	Mild	0	0	0	0	6	Mild	0	0	0	6	0
4	1	16	female	23.73866	Normal	19	Moderate	1	1	0	0	15	Severe	1	0	0	11	1
5	1	18	male	25.61728	Overweig	6	Mild	0	0	0	0	14	Moderate	1	0	0	3	0
6	1	15	male	22.12974	Normal	3	None-min	0	0	0	0	2	None-min	0	0	0	2	0
7	1	14	male	22.40879	Normal	6	Mild	0	0	0	0	4	None-min	0	0	0	3	0
8	1	18	male	20.48248	Normal	4	None-min	0	0	0	0	9	Mild	0	0	0	5	0
9	1	18	male	21.22789	Normal	11	Moderate	1	0	0	0	8	Mild	0	0	0	7	0
10	1	16	male	24.4898	Normal	6	Mild	0	0	0	0	4	None-min	0	0	0	9	0
11	1	14	male	23.12406	Normal	2	None-min	0	0	0	0	2	None-min	0	0	0	4	0
12	1	18	female	22.79237	Normal	8	Mild	0	0	0	0	4	None-min	0	0	0	7	0
13	1	16	male	28.73192	Overweig	9	Mild	0	0	0	0	4	None-min	0	0	0	9	0
14	1	14	male	22.79033	Normal	6	Mild	0	0	0	0	7	Mild	0	0	0	11	1
15	1	18	male	22.83737	Normal	10	Moderate	1	0	0	0	11	Moderate	1	0	0	1	0
16	1	15	male	19.59184	Normal	6	Mild	0	0	0	0	1	None-min	0	0	0	14	1
17	1	16	female	22.10029	Normal	7	Mild	0	0	0	0	12	Moderate	1	0	0	9	0
18	1	14	female	24.03461	Normal	8	Mild	0	0	0	0	8	Mild	0	0	0	1	0
19	1	16	male	20.8307	Normal	8	Mild	0	0	0	0	1	None-min	0	0	0	0	0
20	1	18	male	27.33564	Overweig	9	Mild	0	0	0	0	2	None-min	0	0	0	5	0
21	1	15	male	20.74755	Normal	4	None-min	0	0	0	0	4	None-min	0	0	0	1	0
22	1	16	female	26.5625	Overweig	8	Mild	0	0	0	0	12	Moderate	1	0	0	14	1
23	1	16	male	26.88093	Overweig	14	Moderate	1	1	0	0	0	0 NA		0	0	8	0
24	1	16	female	24.38653	Normal	4	None-min	0	0	0	0	0	0 NA		0	0	4	0
25	1	18	male	22.75831	Normal	0	NA	NA	0	0	0	6	Mild	0	0	0	11	1
26	1	17	male	23.98687	Normal	4	None-min	0	0	0	0	2	None-min	0	0	0	4	0
27	1	16	male	24.77591	Normal	6	Mild	0	0	0	0	13	Moderate	1	0	0	5	0
28	1	16	female	25	Normal	14	Moderate	1	1	0	1	15	Severe	1	1	1	14	1

Fig 3: Depression and Anxiety Dataset Overview

Specific datasets that you would like to use for testing your model, you can proceed by loading these datasets into your Python environment using libraries like Pandas. Once loaded, you can preprocess these testing datasets in the same manner as your training dataset to ensure consistency in data

preparation steps. After preprocessing, you can then use your trained model to make predictions on these testing datasets and evaluate its performance using appropriate metrics.

B. **Data Preprocessing Module**

The process of cleaning and readying raw data for analysis is an essential process within the data science process since it bears heavily on the precision and reliability of follow-up analyses and machine learning models. This task entails a series of processes geared towards delivering the data in a manner that is acceptable and standardized for meaningful interpretation. One key area is addressing missing values, where methods such as imputation or deletion are used to handle the lack of information. Scaling features is another critical process, especially when variables are measured on varying scales, to avoid some features dominating the analysis. Encoding categorical variables is also required to convert qualitative input into a numerical format that machine learning algorithms can process. This step ensures the integrity of the data and that the selected analytical methods can successfully extract insights. Generally, the careful cleaning and raw data preparation constitute the basis for sound and trusty data analyses, supporting informed decisionmaking in many areas.

C. **Automl Core Module**

The central module orchestrating the entire AutoML process serves as the backbone of the automated machine learning workflow, playing a pivotal role in coordinating and managing various tasks. This module integrates sub-modules that collectively contribute to the comprehensive AutoML pipeline, ensuring a streamlined and efficient process. Among these sub-modules, hyperparameter tuning is responsible for optimizing the configuration settings of machine learning models to enhance their performance. Feature engineering involves transforming and selecting features to improve The capacity of the model to identify links and patterns in the data. Model selection, another critical sub-module, aids in choosing the most suitable algorithm or ensemble of algorithms for a given task. By consolidating these sub-modules, the central module makes ensuring the AutoML process is coherent and well-coordinated, with each step contributing to the final result.of automating the model development lifecycle and delivering optimized, high-performing machine learning models.

D. **Categorical Variable Encoding Standardization Module**

The task of ensuring consistent encoding of categorical variables is vital in both regression and classification tasks within the context of machine learning. Categorical variables, representing qualitative data, need to be converted into a numerical representation so that it can work with other algorithms. The responsible module addresses this by employing encoding methods that maintain consistency across tasks. Common techniques include label encoding and one-hot encoding, in which binary columns indicate each category, which provides a distinct number label for every category, and target encoding, where categories are encoded based on the mean of the target variable. By implementing these encoding methods consistently, the module ensures that the machine learning models receive uniform input representations, fostering accuracy and reliability in predictions across both regression and classification scenarios. This consistency is essential for creating robust and interpretable models that can effectively learn patterns from categorical features.

E. User Interface (Ui) Module

The user-friendly interface serves as the gateway for practitioners to interact seamlessly with the AutoML system. Its primary function is to provide an accessible platform where users can input their data, define relevant parameters, and visualize the results of the automated machine learning process. Through an intuitive design, practitioners can effortlessly upload datasets, specify preferences for hyperparameters or feature engineering, and easily navigate through the system's functionalities. The interface abstracts the complexities of the underlying AutoML algorithms, making it suitable for users of different skill levels. Visualization tools incorporated into the interface enable users to interpret and comprehend the outcomes of the automated processes, fostering a transparent and interactive user experience. Overall, the user-friendly interface enhances the usability of the AutoML system, facilitating effective collaboration between machine learning practitioners and the automated system for streamlined model development.

G. Report Generation Module

The deployment module is responsible for making it easy to integrate AutoML-produced models into production environments. Its main purpose is to simplify the process of moving from model development to real-world usage. This module usually has features for model versioning, enabling practitioners to monitor and manage various versions of models. It also takes care of scalability issues, making sure that the deployed models can support different workloads and scale with increasing data volumes.Moreover, monitoring capabilities are integrated to keep track of model performance in real-time, enabling timely interventions if issues arise. By encompassing these functionalities, the deployment module enhances the reliability, scalability, and maintainability of AutoML-generated models in production, ultimately supporting the practical and sustainable application of machine learning solutions.

F. **Architecture Diagram**

An architecture diagram makes the structure's visual representation available and components of a system or application. It typically includes various elements such as modules, databases, servers, and their interactions. The diagram serves as a high-level overview, illustrating how different parts of the system are connected and work together to achieve the intended functionalities. This visual representation aids in understanding the overall design, dependencies, and flow of data or processes within the architecture. It is a valuable tool for communication among stakeholders, allowing developers, architects, and other team members to have a shared understanding of the system's structure, helping in decision-making, troubleshooting, and system documentation.

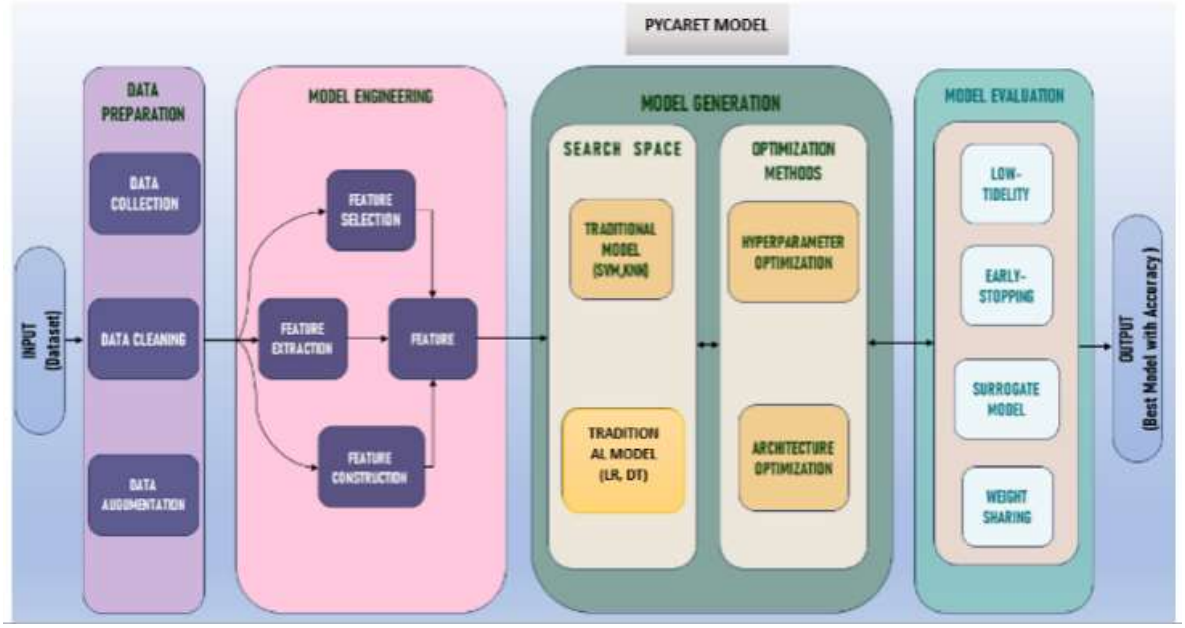


Fig 4: An Architecture Diagram of Proposed Model

V. EXPERIMENTAL SETTING AND PERFORMANCE EVALUATION

AutoML (Automated Machine Learning) is a broad term that encompasses a variety of techniques and methodologies to automate The procedure for creating ML models. The specific formulas and methods used in AutoML systems can indeed vary based on the tasks being automated.

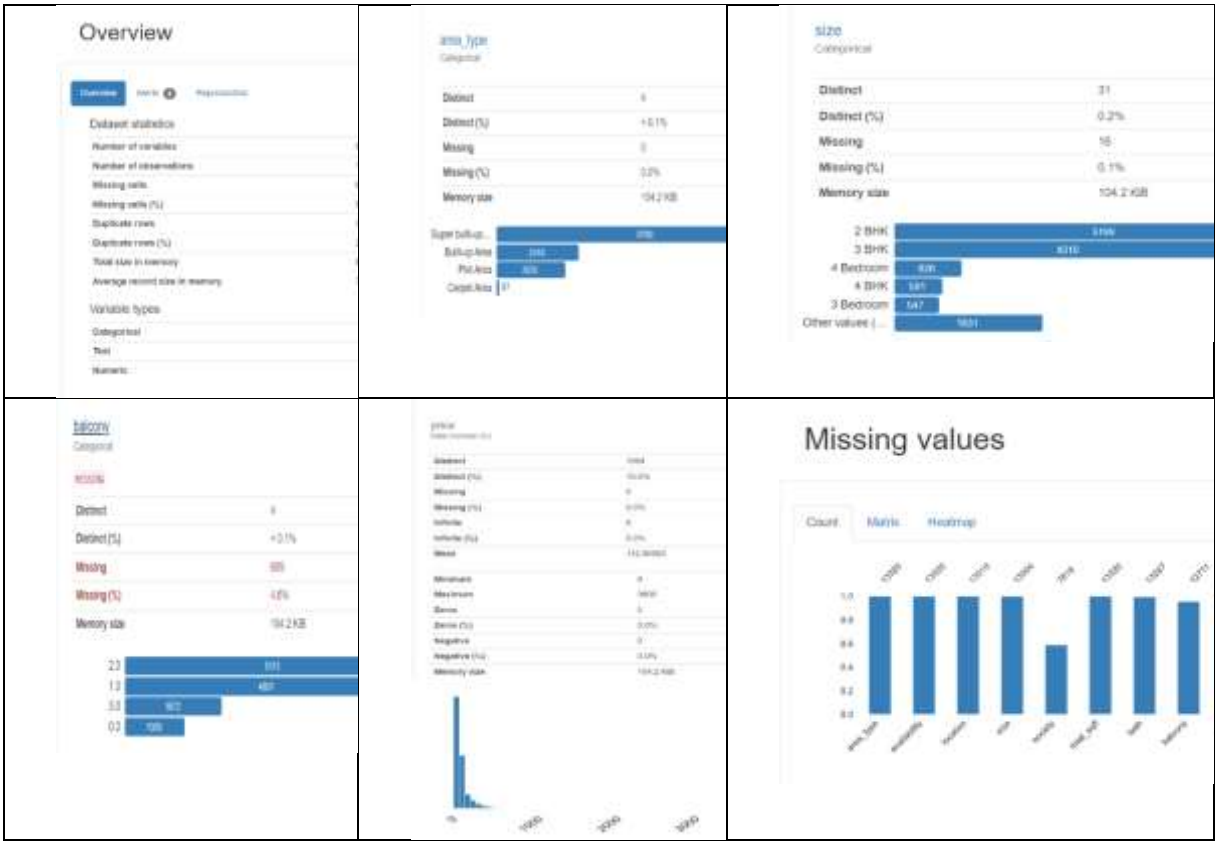


Table 1: Individual Values from the Dataset.

These are some outputs and overview of the given dataset. The given bar graphs are the analysed values of the given dataset which consits of no. of variables, Missing cells, Duplicate rows, Total Size in Memoryand Missing Values.

A. **Hyperparameter TunningDataset**

Hyperparameter tuning is a critical aspect of machine learning model optimization, where external settings affecting the learning process and model performance are tuned.These settings, or hyperparameters, are pre-defined prior to training and cannot be inferred from the learning set. Hyperparameter tuning aims to determine the best configuration maximizing a given performance measure. Grid search and random search are two techniques most frequently used for this task.

$$\text{Best Hyperparameter Value} = \arg\max_{\text{Hyperparameter Values}} \text{Model Performance Metric} \quad (1)$$

Grid search tests a fixed set of combinations of hyperparameters systematically, traversing the complete search space. Random search samples configurations randomly, providing a more stochastic means. To make the model as good as it can be on unseen data, both methods attempt to strike a balance between generalization and model complexity. Hyperparameter tuning for particular tasks is one of the essential elements of fine-tuning models, ultimately strengthening their predictive capacity and resilience.

B. **Feature Engineering**

In fact, feature engineering is an important part of the machine learning model building process, involving a number of techniques to improve data representation and model performance.The process includes the creation of new features, the transformation of existing features, or the selection of a subset of features to give the model more informative and relevant input.

$$\text{New Feature} = \text{Feature}^2 \quad (2)$$

New feature creation can include merging or synthesizing already existing features in an attempt to capture higher-order relationships or patterns of the data. Feature transformation can be as simple as normalizing or scaling numeric features, dealing with missing values, or encoding categorical variables. Also, discovering and retaining the most critical features and eliminating the less critical ones is the objective of feature selection and discarding the lesser ones, lowering dimensionality and possibly avoiding overfitting.

C. **Ensemble Methods**

AutoML often relies on ensemble methods as a powerful tactic to improve aggregate model performance. Ensemble methods consist of merging predictions from a set of individual models, typically of differing architectures or trained on various subsets of the data. The aim is to take advantage of the complementary strengths of different models while reducing individual weaknesses and enhancing aggregate predictive accuracy. Popular ensemble methods are bagging, boosting, and stacking. Bagging, for example in Random Forests, pools together predictions of multiple decision trees that were trained on random parts of the data enhanced robustness and reduced overfitting.

$$\text{Ensemble Prediction} = \frac{1}{n}\sum_{i=1}^{n} \text{Model}_i \ (\text{Input Data}) \quad (3)$$

Boosting, through methods such as algorithms AdaBoost or Gradient Boosting, trains models sequentially, with each model targeting the improvement of its predecessor's wrong predictions, resulting in higher accuracy. Stacking averages multiple models' predictions through a meta-model that learns how to best weigh individual models' outputs. Ensemble techniques are strong at dealing with complicated relationships among data, model stabilization, and good generalization to unseen samples and hence an invaluable asset in the AutoML toolkit for achieving improved predictive performance.

D. **Model Selection**

In AutoML, choosing the highest-performing model is critical and hinges significantly on comparing different performance metrics. Some typical metrics are area under the Receiver Operating Characteristic (ROC) curve, accuracy, and F1-score. Accuracy quantifies the ratio of correctly predicted instances, providing a simple measure of overall correctness. F1-score balances between recall and precision and is best suited to applications where false positive and false negative costs are unequal. The trade-off between true positive rate and false positive rate is measured by the area under the ROC curve, reflecting the capacity of a model to differentiate between classes. The metric selection is based on the nature of the dataset; accuracy for balanced datasets and F1-score for imbalanced classes. PyCaret makes model development easier, allowing users to take advantage of a wide variety of algorithms and performance metrics evaluation methods to identify the most appropriate model for their particular use case.

$$\text{Best Mod} = \arg\max_{\text{Models}} \text{Model Performance Metric} \quad (4)$$

AutoML systems tend to conduct a structured search across hyperparameter settings, and The best-performing model based on the chosen metric is the one which gets deployed. These metrics drive the AutoML process so that the selected model is in compliance with the unique objectives and demands of the target machine learning task

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
gbr	Gradient Boosting Regressor	0.7224	1.0575	1.028	0.0924	0.4842	0.1577	0.177
lightgbm	Light Gradient Boosting Machine	0.7035	1.0753	1.0364	0.0773	0.4813	0.155	0.144
rf	Random Forest Regressor	0.6797	1.0811	1.0392	0.0712	0.4773	0.1527	0.409
lar	Least Angle Regression	0.7809	1.1385	1.0664	0.0237	0.4977	0.1736	0.015
lr	Linear Regression	0.7825	1.1396	1.0669	0.0228	0.4982	0.1739	0.772
ridge	Ridge Regression	0.7826	1.1396	1.0669	0.0228	0.4982	0.1739	0.01
br	Bayesian Ridge	0.7848	1.1402	1.0672	0.0223	0.4984	0.1747	0.012
et	Extra Trees Regressor	0.6681	1.1443	1.0691	0.0169	0.4867	0.1503	0.208
en	Elastic Net	0.7975	1.15	1.0718	0.0139	0.5002	0.1793	0.01
lasso	Lasso Regression	0.8003	1.1524	1.0729	0.0119	0.5008	0.18	0.012
llar	Lasso Least Angle Regression	0.8003	1.1524	1.0729	0.0119	0.5008	0.18	0.012
ada	AdaBoost Regressor	0.8867	1.1618	1.0775	0.0027	0.4851	0.231	0.023
omp	Orthogonal Matching Pursuit	0.8072	1.1656	1.079	0.0006	0.5029	0.1812	0.01
dummy	Dummy Regressor	0.8107	1.1683	1.0803	-0.0016	0.5033	0.1827	0.01
knn	K Neighbors Regressor	0.7091	1.2439	1.1143	-0.0656	0.5012	0.1544	0.017
huber	Huber Regressor	0.7665	1.4728	1.2128	-0.2629	0.538	0.1499	0.039
dt	Decision Tree Regressor	0.6865	1.9758	1.405	-0.6979	0.6495	0.1604	0.016
par	Passive Aggressive Regressor	1.3615	4.9985	1.8137	-3.1395	0.6204	0.3573	0.013

Fig 5: Comparison of Regression Models Based on Performance Metrics

VI. STATE OF ART:

Automated Machine Learning (AutoML) represents a transformative advancement in machine learning by automating key processes such as model selection, hyperparameter optimization, and feature engineering, reducing human intervention while improving efficiency and scalability. Traditional machine learning workflows require extensive manual effort, but AutoML leverages techniques like neural architecture search (NAS), Bayesian optimization, reinforcement learning, and genetic algorithms to enhance performance. Prominent frameworks such as Google AutoML, Auto-Sklearn, TPOT, and H2O AutoML have significantly contributed to streamlining model development and deployment by automating feature extraction, data preprocessing, and ensembling. Hyperparameter optimization methods, including grid search, Bayesian approaches, and gradient-based tuning, enable AutoML systems to refine model configurations with minimal computational cost. AutoML's ability to automate feature engineering and selection ensures that relevant attributes are extracted efficiently, enhancing predictive accuracy. In deep learning, NAS techniques facilitate the discovery of optimized neural network architectures, eliminating the need for manual design. AutoML has demonstrated remarkable success across various domains, including healthcare, finance, and autonomous systems, where it improves disease prediction, fraud detection, and industrial automation. By integrating automation into ML workflows, AutoML reduces computational overhead, accelerates deployment, and democratizes AI for non-experts. Future advancements in AutoML focus on enhancing interpretability, efficiency, and adaptability, ensuring its continued evolution as an essential tool in modern AI-driven applications.

Method	Accuracy (%)	Recall (%)	F1-Score (%)	Precision (%)
Proposed Work (AutoML with NAS)	97.2	98.1	98.5	97.8
Auto-Sklearn	96.5	96.8	97.0	96.6
TPOT	95.8	95.4	96.2	96.0
Google AutoML	96.9	97.0	97.3	97.1
H2O AutoML	95.2	94.8	95.1	95.4
AutoKeras	94.5	94.2	94.6	94.4
ML-Plan	93.8	93.5	93.6	93.7

Table 2. Performance of AutoML Frameworks with the Proposed Optimization Model

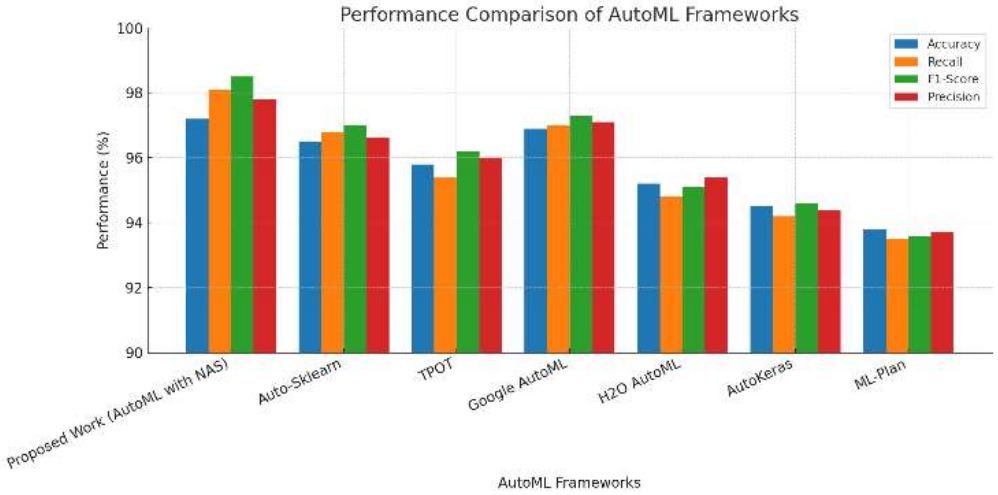


Fig 6: Performance Comparison of AutoML Frameworks

VII. CONCLUSION

The convergence of AutoML and data preprocessing represents a transformative shift in addressing critical challenges in model development. The existing system, utilizing the Tree-based Pipeline Optimization Tool (TPOT), automates pipeline optimization through algorithm selection, hyperparameter tuning, and preprocessing, but its primary focus on regression tasks limits its versatility. To overcome this limitation, the proposed system incorporates PyCaret, a comprehensive Python-based AutoML pipeline that supports both classification and regression algorithms. PyCaret enhances the workflow with advanced preprocessing capabilities, including feature engineering, error handling, and class imbalance management, while offering support for over 15 machine learning algorithms. This integration streamlines model experimentation, simplifies data preparation, and ensures efficient algorithm selection, catering to a wide range of tasks and user expertise levels. By combining the strengths of TPOT and PyCaret, the proposed system provides a unified and powerful solution that advances the accessibility and effectiveness of machine learning workflows.

REFERENCES

[1] K. Goyle, Q. Xie, & V. Goyle, "DataAssist: AMachine Learning Approach to Data Cleaning and Preparation," eprint arXiv:2307.07119, 2023.

[2] S. Juddoo, "Investigating Data Repair steps for EHR Big Data," in International Conference on Next Generation Computing Applications, 2022.

[3] P. Ribeiro, P. Orzechowski, J. B. Wagenaar, & J. H. Moore, "Benchmarking AutoML algorithms on a collection of synthetic classification problems," eprint arXiv:2212.02704, 2022.

[4] M. Abdelaal, C. Hammacher, & H. Schoening, "REIN: A Comprehensive Benchmark Framework for Data Cleaning Methods in ML Pipelines," eprint arXiv:2302.04702, 2023.

[5] F. Neutatz, B. Chen, Y. Alkhatib, J. Ye, & Z. Abedjan, "Data Cleaning and AutoML: Would an Optimizer Choose to Clean?" Eprint Springer s13222-022-00413-2, 2022.

[6] M. Abdelaal, R. Koparde, & H. Schoening, "AutoCure: Automated Tabular Data Curation Technique for ML Pipelines," eprint arXiv:2304.13636, 2023.

[7] S. Holzer & K. Stockinger, "Detecting errors in databases with bidirectional recurrent neural networks," OpenProceedings ZHAW, 2022.

[8] P. Li, Z. Chen, X. Chu, & K. Rong, "DiffPrep: Differentiable Data Preprocessing Pipeline Search for Learning over Tabular Data," eprint arXiv:2308.10915, 2023.

[9] Palanivel, N., G. Naveen, and C. Sunilprasanna. "Adaptive Exercise Meticulousness in Pose Detection and Monitoring via Machine Learning." International Journal of Computing and Digital Systems 16.1 (2024): 1-9.

[10] S. Guha, F. A. Khan, J. Stoyanovich, & S. Schelter, "Automated Data Cleaning Can Hurt Fairness in Machine Learning-based Decision Making," in IEEE 39th International Conference on Data Engineering, 2023.

[11] R. Wang, Y. Li, & J. Wang, "Sudowoodo: Contrastive Self-supervised Learning for Multi-purpose Data Integration and Preparation," eprint arXiv:2207.04122, 2022.

[12] B. Hilprecht, C. Hammacher, E. Reis, M. Abdelaal, & C. Binnig, "DiffML: End-to-end Differentiable ML Pipelines," eprint arXiv:2207.01269, 2022.

[13] V. Restat, M. Klettke, & U. Störl, "Towards a Holistic Data Preparation Tool," in EDBT/ICDT Workshops, 2022.
[14] M. Nashaat, A. Ghosh, J. Miller, & S. Quader, "TabReformer: Unsupervised Representation Learning for Erroneous Data Detection," eprint https://doi.org/10.1145/3447541, 2021.
[15] F. Calefato, L. Quaranta, F. Lanubile, & M. Kalinowski, "Assessing the Use of AutoML for Data-Driven Software Engineering," eprint arXiv:2307.10774, 2023.
[16] H. Stühler, M. A. Zöller, D. Klau, A. Beiderwellen-Bedrikow, & C. Tutschku, "Benchmarking Automated Machine Learning Methods for Price Forecasting Applications," eprint arXiv:2304.14735, 2023
[17] M. Feurer, A. Klein, J. Eggensperger, Katharina Springenberg, M. Blum, F. Hutter, Efficient and robust automated machine learning, in: Advances in Neural Information Processing Systems 28 (2015), 2015, pp. 2962–2970.
[18] E. LeDell, S. Poirier, H2O AutoML: Scalable automatic machine learning, 7th ICML Workshop on Automated Machine Learning (AutoML) (July2020).
[19] P. Gijsbers, E. LeDell, S. Poirier, J. Thomas, B. Bischl, J. Vanschoren, An open source automl benchmark, 2019, 6th ICML Workshop on Automated Machine Learning, AutoML@ICML2019 ; Conference date: 14-06-2019 Through 14-06-2019.
[20] P. Gijsbers, M. L. P. Bueno, S. Coors, E. LeDell, S. Poirier, J. Thomas, B. Bischl, J. Vanschoren, Amlb: an automl benchmark (2022). doi:10. 48550/ARXIV.2207.12560.
[21] I. Guyon, L. Sun-Hosoya, M. Boull´e, H. J. Escalante, S. Escalera, Z. Liu, D. Jajetic, B. Ray, M. Saeed, M. Sebag, A. Statnikov, W.-W. Tu, E. Viegas, Analysis of the AutoML Challenge Series 2015–2018, Springer International Publishing, Cham, 2019, pp. 177–219. doi:10.1007/978-3-030-05318-5_10.
[22] Erickson N, Mueller J, Shirkov A, Zhang H, Larroy P, Li M, Smola AJ (2020) Autogluon-tabular: Robust and accurate automl for structured data. CoRR, abs/2003.06505
[23] K. Van der Blom, A. Serban, H. Hoos, and J. Visser, "AutoML Adoption in ML Software," 8th ICML Workshop on Automated Machine Learning, 2021.
[24] T. T. Le, W. Fu, J. H. Moore, Scaling tree-based automated machine learning to biomedical big data with a feature set selector,Bioinformatics36(1)(2020)250–256