+ A

Mini Project Report

On

#### AUTOMATED MACHINE LEARNING: THE NEW WAVE OF MACHINE LEARNING

Submitted in partial fulfillment of the requirements for the award of Degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING (AI&ML)

by

SHEIK MALEKA SHAMEEM (217R1A6653)

BHAGYAWAR MANIDEEP (217R1A6607)

TELAKUNCHI VISHAL (217R1A6658)

Under the Guidance of

**MR. PRASHANT MUTALIK DESAI**

ASSISTANT PROFESSOR CSE(AI&ML)

##### **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)**

**CMR TECHNICAL CAMPUS UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

**2021-2025**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)**



#### CERTIFICATE

This is to certify that the project entitled **“AUTOMATED MACHINE LEARNING:THE NEW WAVE OF MACHINE LEARNING”** being submitted by **SHEIK MALEKA SHAMEEM (217R1A6653), BHAGYAWAR MANIDEEP (217R1A6607) & TELAKUNCHI VISHAL (217R1A6658)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering (AI&ML) to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

**MR. PRASHANT MUTALIK DESAI Dr. S RAO CHINTALAPUDI**

Assistant Professor CSE(AI&ML) HOD CSE(AI&ML)

INTERNAL GUIDE

**EXTERNAL EXAMINER**

**Submitted for viva voice Examination held on**

##### **ACKNOWLEDGEMENT**

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Prashant Mutalik Desai,** Associate Professor and HOD CSE(AI&ML) for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **Dr. G. Vinoda Reddy, Dr. K. Mahesh, Dr.V.Malsoru,Dr.Md.Shareef,M.Balaji & M. Mahesh** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. S Rao Chintalapudi,** Head, Department of Computer Science and Engineering (AI&ML) for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

###### **SHEIK MALEKA SHAMEEM (217R1A6653)**

**BHAGYAWAR MANIDEEP (217R1A6608)**

**TELAKUNCHI VISHAL (217R1A6658)**

**ABSTRACT**

With the explosion in the use of machine learning in various domains, the need for an efficient pipeline for the development of machine learning models has never been more critical. However, the task of forming and training models largely remains traditional, with a dependency on domain experts and time-consuming data manipulation operations. This impedes the development of machine learning models in both academia and industry.

This demand advocates for a new research era focused on fitting machine learning models fully automatically, known as Automated Machine Learning (AutoML). AutoML is an end-to-end process that aims at automating the model development pipeline without any external assistance. In this context, we first provide insights into AutoML and its significance.

Next, we delve into the individual segments of the AutoML pipeline and cover their approaches in brief. We also provide a case study on the industrial use and impact of AutoML, with a focus on its practical applicability in a business context. Finally, we conclude with the open research issues and future research directions in this evolving field.

**LIST OF FIGURES**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FIGURE NO** | | **FIGURE NAME** | | **PAGE NO** | |
| 3.1 | | Project Architecture For Automated Machine Learning The New Wave Of Machine Learning. | | 12 | |
|  | |  | |  | |
| 3.2 | | Use Case Diagram For Automated Machine Learning The New Wave Of Machine Learning. | | 15 | |
| 3.3 | | Class Diagram For Automated Machine Learning The New Wave Of Machine Learning. | | 17 | |
|  | |  | |  | |
| 3.4  4.3  4.5 | | Sequence diagram For Automated Machine Learning The New Wave Of Machine Learning.  Confusion Matrix  Result Analysis | | 19    27  34 | |
|  |
| 5.1 | | | | Main Home Page | | 37 | |
| 5.2 | | | | User Login Page | | 37 | |
| 5.3 | | | | User Register Page | | 38 | |
| 5.4 | | | | Adding Data To Be Tested Page | | 38 | |
| 5.5 | | | | Perform Predections Analysis Page | | 39 | |

5.6 Test Results Page 40

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **TABLE NAME** | **PAGE NO** |
| 6.3 | TESTCASES | 44 |

**TABLE OF CONTENTS**

**ABSTRACT** i

LIST OF FIGURES ii

LIST OF TABLES iii

1. [INTRODUCTION 1](#_bookmark0)
   1. [PROJECT SCOPE 1](#_bookmark1)
   2. [PROJECT PURPOSE 2](#_bookmark2)
   3. [PROJECT FEATURES 2](#_bookmark3)
2. [SYSTEM ANALYSIS 4](#_bookmark4)
   1. [PROBLEM DEFINITION 4](#_bookmark5)
   2. [EXISTING SYSTEM / LITERATURE REVIEW 4](#_bookmark6)
      1. EXISTING SYSTEM 6
      2. LIMITATIONS OF EXISTING SYSTEMS 7
   3. [PROPOSED SYSTEM](#_bookmark7) 7
      1. PROPOSED APPROACH 8
      2. ADVANTAGES OF PROPOSED SYSTEM 9
   4. [HARDWARE & SOFTWARE REQUIREMENTS](#_bookmark11) 9
      1. [HARDWARE REQUIREMENTS](#_bookmark12) 9
      2. [SOFTWARE REQUIREMENTS](#_bookmark13) 10
3. [ARCHITECTURE 1](#_bookmark14)1
   1. [PROJECT ARCHITECTURE 1](#_bookmark15)2
   2. [USE CASE DIAGRAM 1](#_bookmark17)5
   3. [CLASS DIAGRAM 17](#_bookmark18)
   4. [SEQUENCE DIAGRAM 1](#_bookmark19)9
4. [IMPLEMENTATION](#_bookmark21) 22
   1. SUPPORT VECTOR MACHINE 23
   2. AUTOML 24
   3. DATASET DESCRIPTION 25
   4. PERFORMANCE METRICS 26
   5. [SAMPLE CODE 2](#_bookmark22)8
   6. RESULT ANALYSIS 34
5. [SCREENSHOTS 36](#_TOC_250001)
6. [TESTING 41](#_bookmark23)
   1. [INTRODUCTION TO TESTING 42](#_bookmark24)
   2. [TYPES OF TESTING 42](#_bookmark25)
      1. [UNIT TESTING 42](#_bookmark26)
      2. [INTEGRATION TESTING 42](#_bookmark27)
      3. [FUNCTIONAL TESTING 43](#_bookmark28)
   3. [TEST CASES 44](#_bookmark29)

7.[CONCLUSION & FUTURE SCOPE 45](#_TOC_250000)

7.1 [CONCLUSION 46](#_bookmark31)

7.2 [FUTURE SCOPE 46](#_bookmark32)

BIBLIOGRAPHY 47

[GITHUB LINK 48](#_bookmark34)

1. **INTRODUCTION**

**1.** **INTRODUCTION**

##### **1.1 PROJECT SCOPE**

The scope of this project involves exploring and implementing Automated Machine Learning (AutoML) techniques to streamline the machine learning pipeline from data preprocessing to model deployment. It includes researching AutoML concepts, selecting and setting up AutoML tools, automating data cleaning and feature engineering, building and tuning models using AutoML capabilities, and evaluating their performance against traditional methods. The project will also involve deploying the best-performing models and exploring advanced features like deep learning integration and edge computing. Deliverables include a detailed project report, a summarizing presentation, a code repository, and deployed models demonstrating AutoML's practical applications. Activities like developing new AutoML tools, extensive customization of existing tools, large-scale industry deployment, and exhaustive evaluation of all AutoML tools are outside the project's scope.

#### 1.2 PROJECT PURPOSE

#### The purpose of this project is to explore and demonstrate the capabilities of Automated Machine Learning (AutoML) in automating the end-to-end process of developing machine learning models. By leveraging AutoML, the project aims to increase accessibility to machine learning for individuals and organizations with limited expertise, enhance efficiency by reducing the time and effort needed to build and deploy models, and optimize performance through automated algorithm selection and hyperparameter tuning. Additionally, the project seeks to compare the performance of AutoML-generated models with traditional methods, explore advanced features like deep learning integration and edge computing, and provide comprehensive documentation and insights into AutoML tools and their practical applications. Ultimately, the project aims to showcase the transformative potential of AutoML in making machine learning more efficient, effective, and widely accessible.

##### **1.3 PROJECT FEATURES**

The project will feature automated data preprocessing, including data cleaning and feature engineering, to handle missing values, outliers, and generate meaningful features from raw data. It will use AutoML tools for automated model selection, evaluating and selecting the best-performing algorithms, and employing ensembling techniques to improve robustness. Hyperparameter tuning will be automated to optimize model performance. The project will evaluate models using appropriate performance metrics and cross-validation, benchmarking AutoML-generated models against traditional methods. Automated deployment strategies will be implemented for real-world scenarios, with processes set up for monitoring and maintenance. Advanced features will be explored, such as integrating deep learning models, deploying on edge devices for real-time predictions, and combining automated processes with human expertise. Popular AutoML tools like Google Cloud AutoML, H2O.ai, auto-sklearn, TPOT, and Microsoft Azure AutoML will be utilized. The project will also include comprehensive documentation of research, methodologies, experiments, results, and conclusions, along with presentations and a maintained code repository. These features aim to showcase AutoML's efficiency, accessibility, and performance optimization potential in machine learning.

## **2.SYSTEM ANALYSIS**

##### **2. SYSTEM ANALYSIS**

**SYSTEM ANALYSIS**

System Analysis is the important phase in the system development process. TheSystem is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

##### **PROBLEM DEFINITION**

The goal of Automated Machine Learning (AutoML) is to democratize access to advanced machine learning by automating the entire process from data preprocessing to model deployment. This initiative addresses the challenge that many businesses and researchers face due to a lack of expertise in developing complex machine learning models manually. By leveraging diverse datasets from various domains such as healthcare and finance, and handling different data formats like tabular, images, and text, an effective AutoML system aims to achieve high model performance metrics (accuracy, F1 score, MAE) comparable to those crafted by experts. The success of such a system lies in its ability to significantly reduce development time and cost, providing interpretable results and enabling non-experts to utilize machine learning effectively and efficiently.

**2.2 EXISTING SYSTEM**

In the existing system the data preprocess has dine with structured data. Even though data pre-processing consumes a large chunk of time in an ML pipeline, it is astonishing to see the inadequate amount of work done to automate it. For data preprocessing, it can be noted that while the data pre-process approaches are adequate for structured data, work still needs to be done to assimilate on Structured data. We suggest the incorporation of data-mining methods as they can deal with such unformed data. This can allow AutoML pipelines to create models capable of learning from Internet sources. In feature engineering, it should be noted that most methods used until now adhere to supervised learning. However, dataset specificity is high, and therefore, AutoML pipelines should be as generic as possible to accommodate the diverse datasets. Therefore, a gradual paradigm shift towards unsupervised.

**2.2.1 EXISTING APPROACHS**

Existing AutoML systems are designed to streamline various stages of the machine learning pipeline, significantly enhancing efficiency and minimizing the need for human intervention. These systems typically encompass tasks such as data preprocessing, feature engineering, model selection, and hyperparameter optimization. For instance, H2O-AutoML and DataRobot focus on automating model selection and hyperparameter tuning using techniques like random search, grid search, and Bayesian optimization. Tools like TPOT employ evolutionary algorithms to optimize entire pipelines, while Auto-Keras and Auto-Sklearn emphasize automating feature engineering and model tuning with methods including reinforcement learning and meta-learning.

Each AutoML tool has distinct strengths, with some specializing in particular aspects of the pipeline, such as data preprocessing or feature engineering, while others provide comprehensive solutions covering the entire workflow. The advancements in AutoML have enabled these systems to achieve performance levels comparable to or even surpassing those of human experts in certain scenarios. This highlights the growing maturity and effectiveness of automated systems in handling complex machine learning tasks.

* + 1. **LIMITATIONS OF EXISTING APPROACHES**

Following are the disadvantages of existing approaches:

* Feature Generation is not up to the mark where domain experts excepted results.
* Most AutoML tools emphasize the performance but in the real world, that’s just one aspect being covered in machine learning projects.So the companies can’t compromise the computing plus storage specification sheet.
* CASH(Combined Algorithm Selection and Hyperparameter) problem considers model selection and hyperparameters optimization as a single hierarchical parameter.

**2.3 PROPOSED SYSTEM**

The proposed system aims at providing an overview of the advances seen in the realm of AutoML in recent years. We focus on individual aspects of AutoML and summarize the improvements achieved in recent years. The motivation of proposed system stems from the unavailability of a compact study of the current state of AutoML. While we acknowledge the existence of other surveys, their motive is to either provide an in-depth understanding of a particular segment of AutoML, provide just an experimental comparison of various tools used or are fixated towards deep learning models.

**2.3.1 PROPOSED APPROACH**

The proposed approach for the Automated Machine Learning (AutoML) project focuses on overcoming the limitations of existing systems by introducing a flexible and modular pipeline design that allows users to customize each step of the machine learning process, from data preprocessing to model deployment. This flexibility ensures that users can tailor the system to their specific needs, enhancing its applicability across various domains. To address interpretability challenges, the system integrates advanced tools such as SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations), providing clear and detailed explanations of model predictions, which is crucial for domains requiring transparency. Efficient resource management is achieved through the implementation of optimized algorithms for hyperparameter tuning and model selection, leveraging techniques like Bayesian

optimization and genetic algorithms, and utilizing distributed computing to handle large-scale data and complex computations. The approach also emphasizes domain adaptation and transfer learning to enhance the generalization capabilities of models across different datasets, ensuring robust performance in diverse scenarios. Automated data quality enhancement modules are developed to handle missing values, balance datasets, and select relevant features, ensuring high-quality input data. Fairness-aware algorithms and bias detection mechanisms are incorporated to identify and mitigate biases, promoting ethical AI practices. The system is designed to be scalable using advanced parallel processing and distributed training techniques, and offers cost-effective deployment options, including optimized cloud-based services and on-premise solutions. Continuous learning capabilities and regular updates ensure the AutoML system stays aligned with the latest advancements in machine learning algorithms and techniques. By focusing on customization, interpretability, efficient resource management, domain adaptation, data quality, bias mitigation, scalability, and continuous improvement, this approach seeks to enhance the accessibility, effectiveness, and impact of automated machine learning for a wide range of users and applications.

* + 1. **ADVANTAGES OF THE PROPOSED SYSTEM**

The proposed system implemented using the machine learning techniques, the proposed system is processing in the following way.

* We segment the AutoML pipeline into parts and review the contributions in each of these segments.
* We explore the various state-of-the-art tools currently available for AutoML and evaluate them.
* We also incorporate the advancements seen in machine learning which seems to be overshadowed by deep learning in recent years.

##### **HARDWARE & SOFTWARE REQUIREMENTS**

###### **HARDWARE REQUIREMENTS:**

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

* PROCESSOR : Intel i3
* RAM : 4GB (min)
* HARD DISK : 1 TB
* KEYBOARD : Standard Windows Keyboard
* MOUSE : Two or Three Button Mouse
* MONITOR : SVGA

##### **SOFTWARE REQUIREMENTS:**

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements

* OPERATING SYSTEM : Windows 10
* CODE LANGUAGE : Python
* LIBRARIES : Scikit-Learn, TensorFlow

pandas, SpaCy and Numpy

* FRONT-END : Python
* BACK-END : Django-ORM
* DESIGNING : HTML, CSS, JavaScript
* DATABASE : MySQL
* Web Server : WAMP Server

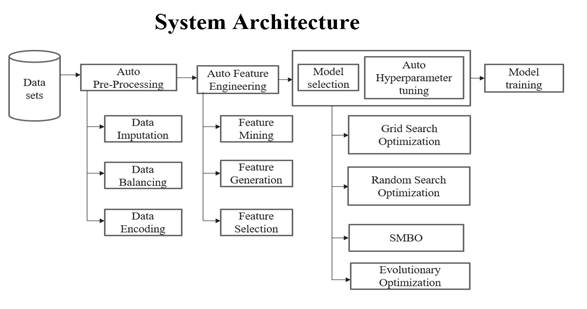
**3. ARCHITECTURE**

##### **3.ARCHITECTURE**

##### **PROJECT ARCHITECTURE**

This project architecture shows the procedure followed for classification,

starting from input to final prediction



3.1: Project Architecture of Automated Machine Learning The New Wave Of Machine Learning

###### **DESCRIPTION**

The provided system architecture diagram illustrates a comprehensive automated machine learning (AutoML) pipeline that streamlines various stages of the machine learning process. The pipeline begins with the input of datasets, which contain raw data that needs to be processed. The first major stage, auto pre-processing, includes tasks such as data imputation to handle missing values, data balancing to address class imbalances, and data encoding to convert categorical data into numerical formats suitable for machine learning algorithms.

Following data preprocessing, the pipeline advances to auto feature engineering, where it performs feature mining to extract relevant features, feature generation to create new features from existing data, and feature selection to identify the most important features for model training. This step is crucial for enhancing the predictive power of the model and reducing dimensionality.

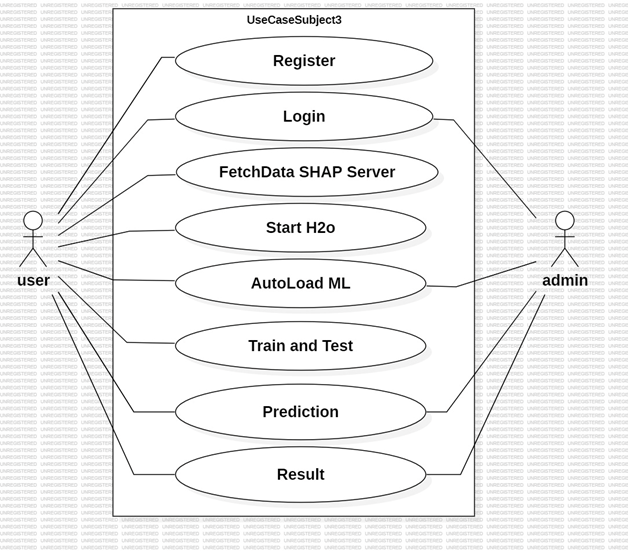
The next stage involves model selection, where the most appropriate machine learning model(s) are chosen based on the processed and engineered features. Once a model is selected, the pipeline proceeds to auto hyperparameter tuning. This step optimizes the model’s performance through various techniques, including grid search optimization, random search optimization, sequential model-based optimization (SMBO), and evolutionary optimization.

Finally, the optimized model undergoes training with the selected features and tuned hyperparameters. This systematic approach ensures that each step in the machine learning workflow is meticulously optimized, leading to improved model accuracy and efficiency while minimizing the need for manual intervention.

###### **3.2 USE CASE DIAGRAM**

In the use case diagram, we have basically two actors who are the user and admin in the trained model.

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.





3.2: Use Case Diagram for Automated Machine Learning The New Wave Of Machine Learning

#### DESCRIPTION

#### The use case diagram outlines the interactions between two primary actors, a user and an admin, with a system designed to facilitate various operations within the machine learning (ML) workflow. The system offers a range of functionalities that cater to both users and admins, ensuring efficient management and utilization of ML processes.

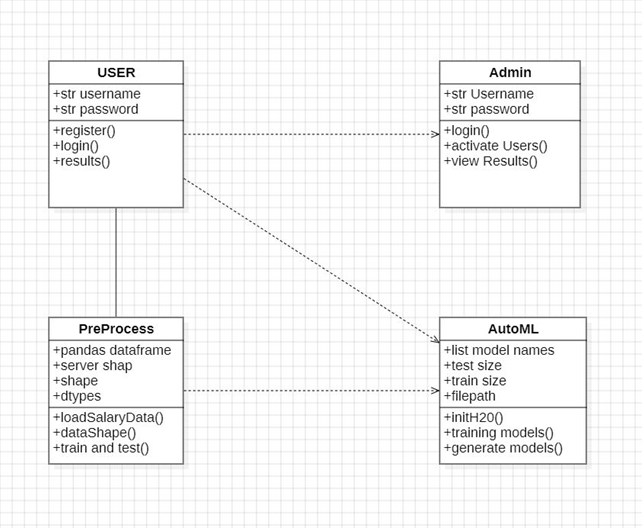
#### Users can register and log in to access the system's features. They can fetch data from the SHAP server for analysis, start the H2O machine learning platform, and automatically load ML models and datasets. Additionally, users can train and test models, perform predictions, and view the results of their predictions.

#### Admins have similar access but with additional responsibilities. They can register and log in to manage the system, fetch data from the SHAP server, and start the H2O platform. Admins also oversee the AutoLoad ML feature, ensuring smooth operation, and assist users in training and testing models. Furthermore, admins perform predictions and manage the results.

#### The diagram effectively captures the essential functionalities and highlights the distinct roles of users and admins. While users primarily focus on utilizing the ML features for their tasks, admins manage and oversee the entire workflow, ensuring the system operates efficiently for all users.

##### **3.3 CLASS DIAGRAM**

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, operations(or methods), and the relationships among objects.



3.3: Class Diagram for Automated Machine Learning The New Wave Of Machine Learning

**DESCRIPTION**

The class diagram represents a system with four main components: USER, Admin, PreProcess, and AutoML. The USER class has two attributes, username and password, and includes methods for user registration, login, and retrieving results. The dotted line indicates an association between the USER and Admin classes, suggesting that user management involves admin oversight.

The Admin class, with attributes Username and password, includes methods for admin login, activating users, and viewing user results. The association with the USER class implies that admins manage user accounts. Additionally, the Admin class is connected to the AutoML class, indicating that admins may also interact with automated machine learning processes.

The PreProcess class focuses on data preparation with attributes like pandas dataframe, server shap, shape, and dtypes. It includes methods for loading salary data, determining data shape, and splitting data into training and testing sets. This class is linked to the AutoML class, highlighting the importance of data preprocessing before applying automated machine learning.

The AutoML class has attributes for listing model names, test size, train size, and file path, and includes methods for initializing the H2O framework, training models, and generating machine learning models. The association with both Admin and PreProcess classes suggests that automated machine learning relies on preprocessed data and admin supervision for effective model management and deployment.

**3.4 SEQUENCE DIAGRAM**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

#### n

Figure 3.4.1: Sequence Diagram for Automated Machine Learning The New Wave Of Machine Learning

#### DESCRIPTION

#### 

#### The sequence diagram illustrates the interactions between the User, Admin, Datapreprocess, and AutoML components in a system. It begins with the User registering an account by sending a Register() request to the Admin. The Admin then activates the user's account and confirms this with an Activated() message back to the User. The User proceeds to log in by sending a Login() request to the Admin.

#### ‘

#### After logging in, the Admin requests adult data from the shap server by sending a get Adults Data from shap server() message to the Datapreprocess component. Following this, the Admin initiates the AutoML process by sending a loading AutoML() request to the AutoML component. The AutoML component responds with the available auto models using the Auto Models() message and starts the H2O framework with a Start H2O() message to the Admin.

#### Once the models are prepared, the Admin sends the results back to the User with a Results Models() message. The Admin also requests the loading of a salary dataset by sending a load salary dataset() message to the Datapreprocess component. This ensures that the necessary data is preprocessed for further analysis and model training.

#### 

#### Finally, the Datapreprocess component sends the Predictions() message to the AutoML component, which involves generating predictions based on the preprocessed salary dataset. This sequence of interactions captures the entire workflow from user registration, through data preprocessing and model training, to the final prediction generation using automated machine learning mode.

**4. IMPLEMENTATION**

**4.1** **SUPPORT VECTOR MACHINE**

Support Vector Machines (SVMs) play a significant role in Automated Machine Learning (AutoML), which aims to automate the end-to-end process of applying machine learning to real-world problems. SVMs are supervised learning models used for classification and regression analysis, working by finding the hyperplane that best divides a dataset into classes. Key concepts in SVMs include support vectors, hyperplanes, and the kernel trick, which transforms data into higher dimensions for better separation. In the context of AutoML, SVMs are included in model selection processes due to their robustness and effectiveness. AutoML systems evaluate multiple algorithms, including SVMs, to determine the best performing model for a given dataset, involving cross-validation and comparison of performance metrics. Hyperparameter tuning is crucial for SVMs, which have parameters like the regularization parameter C, kernel type, and gamma for RBF kernels. AutoML uses techniques like grid search, random search, or Bayesian optimization to optimize these parameters.

Additionally, feature engineering and preprocessing are automated, applying techniques like normalization or standardization to ensure optimal SVM performance. AutoML may also use ensembling, combining multiple models, including SVMs, for better performance. SVMs offer high accuracy, versatility through different kernel functions, and robustness in high-dimensional spaces. However, they can be computationally expensive and less scalable for large datasets, and their performance is sensitive to hyperparameter and kernel choices. Popular AutoML frameworks incorporating SVMs include Auto-sklearn, TPOT, and H2O.ai, which provide tools for automated model selection, feature engineering, and hyperparameter tuning, making advanced machine learning techniques more accessible and scalable.

**4.2 AutoML**

Automated Machine Learning (AutoML) is exceptionally useful in the "Automated Machine Learning: The New Wave of Machine Learning" project. It automates key steps of the machine learning pipeline, including data preprocessing, feature engineering, model selection, and hyperparameter tuning. This significantly speeds up model development and allows for rapid experimentation and deployment.

AutoML handles data preprocessing tasks such as imputation (mean, median, KNN), scaling (Min-Max, Z-score), and encoding (One-Hot, Label Encoding). For feature engineering, it employs feature selection methods like Chi-square test, Recursive Feature Elimination, and Lasso Regression, along with feature extraction techniques like PCA and t-SNE. Model selection includes a variety of algorithms: Linear Regression, Decision Trees, SVMs, Random Forests, Gradient Boosting Machines, k-Nearest Neighbors, Naive Bayes, and Neural Networks for supervised learning; and K-Means, DBSCAN, and PCA for unsupervised learning. Hyperparameter tuning methods include Grid Search, Random Search, and Bayesian Optimization.

AutoML ensures consistency, reduces human error, and handles large datasets, making it scalable for modern applications. It incorporates ensemble learning techniques like Bagging (Random Forests), Boosting (AdaBoost, XGBoost), and Stacking, and can adapt with online learning algorithms like Stochastic Gradient Descent (SGD). This reduces the need for specialized data scientists, lowering costs and making machine learning accessible to smaller organizations.

AutoML allows for quick model prototyping, faster development cycles, and performance benchmarking. It enables data scientists to focus on strategic tasks and includes tools for deploying and monitoring models. These advantages help accelerate the adoption and impact of machine learning across various industries.

**4.3 DATASET DESCRIPTION**

The "Years Experience and Salary" dataset comprises two main columns: "Years Experience," which captures the number of years an individual has been in their profession, and "Salary," which records their annual earnings in USD <https://www.kaggle.com/datasets/rohankayan/years-of-experience-and-salary-dataset> . These continuous numerical variables provide a foundation for analyzing the impact of professional experience on salary.

By leveraging automated machine learning (AutoML), this dataset can be utilized to construct predictive models that elucidate the relationship between work experience and salary. AutoML automates complex tasks like data preprocessing, feature selection, model selection, and hyperparameter tuning, making advanced predictive modeling accessible even to those with limited expertise in machine learning.

The principles and methodologies used in this analysis can be extended to other datasets with different features and target variables. AutoML can streamline the analysis process across various domains, enabling the development of robust models that can predict outcomes, uncover trends, and provide actionable insights for diverse datasets beyond just years of experience and salary. This versatility makes AutoML a powerful tool for a wide range of data analysis applications.

**4.4 PERFORMANCE METRICS**

**ACCURACY**

In the "Automated Machine Learning: The New Wave of Machine Learning" project, accuracy is a fundamental performance metric used to evaluate the effectiveness of machine learning models. Accuracy measures the ratio of correctly predicted instances to the total number of instances, making it particularly useful for classification tasks. The role of accuracy in this project includes evaluating model performance, guiding model selection, and optimizing hyperparameters. During the model selection phase, the AutoML system compares multiple algorithms and selects the one with the highest accuracy on the validation dataset. Accuracy also serves as a benchmark for model performance, providing a baseline against which improvements from feature engineering and model tuning can be measured. However, accuracy has limitations, especially with imbalanced datasets, where it might provide a misleading measure of performance if the model predominantly predicts the majority class correctly. To address this, accuracy is often used alongside other metrics like precision, recall, F1 score, and AUC-ROC to ensure a comprehensive evaluation. In the "Automated Machine Learning: The New Wave of Machine Learning" project, accuracy helps in evaluating initial models, guiding iterative improvements, and communicating results to stakeholders. This ensures the development of robust, accurate, and reliable machine learning models that can effectively address various real-world problems.

Accuracy= (Number of predictions/Total number of prediction)​×100%

**CLASSIFICATION REPORT**

In the "Automated Machine Learning: The New Wave of Machine Learning" project, a classification report provides a comprehensive evaluation of a model's performance by summarizing key metrics for each class. This report is especially valuable in classification tasks to understand how well the model is performing beyond just accuracy. The classification report typically includes metrics such as precision, recall, F1 score, and support.

**Precision**: Measures the accuracy of positive predictions. It is the ratio of true positives to the sum of true positives and false positives.

Precision= True Positives /( True Positives + False Positives )​

**Recall**:Measures the model’s ability to identify all relevant instances. It is the ratio of true positives to the sum of true positives and false negatives.

Recall= True Positives ​/( True Positives + False Negatives)

**F1 Score**: The F1 score is the harmonic mean of precision and recall, providing a balance between the two metrics. It takes into account both false positives and false negatives and is calculated as 2 \* (precision \* recall) / (precision + recall). F1 score ranges from 0 to 1, where higher values indicate better performance in terms of both precision and recall.

F1=2× (Precision×Recall​/ Precision+Recall)

**Support**: Support represents the number of occurrences of each class in the dataset. It provides insights into the distribution of classes and helps interpret the significance of precision, recall, and F1 score. For instance, if one class has significantly higher support than the other, it may influence the interpretation of the model's performance metrics.

**CONFUSION MATRIX**

In the project on Automated Machine Learning (AutoML), the confusion matrix will be used to evaluate and interpret the performance of classification models. It will summarize prediction results by showing counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). For example, a confusion matrix might reveal 100 true positives, 85 true negatives, 5 false positives, and 10 false negatives. This matrix helps assess the accuracy, precision, recall, and F1 score of the models, offering insights into their performance. It will be instrumental in comparing AutoML-generated models with traditional methods, identifying model biases, and guiding fine-tuning efforts. Overall, the confusion matrix will provide a clear picture of how well the models perform, facilitating better decision-making and improvements in model development.



4.3: Confusion matrix

##### **4.5 SAMPLE CODE**

User Side views.py

from django.shortcuts import render,HttpResponse,redirect

from django.contrib import messages

from .forms import UserRegistrationForm

from .models import UserRegistrationModel,AutoMLDataModel,MyPredectionsModels,ModelPredectionStoreModels

from .UserAutoMachineLearningProcess import StartProcessAutoML

import h2o

from django.core.paginator import Paginator, EmptyPage, PageNotAnInteger

import csv,io

from django\_pandas.io import read\_frame

import matplotlib.pyplot as plt

import numpy as np

# Create your views here.

def UserRegisterActions(request):

if request.method == 'POST':

form = UserRegistrationForm(request.POST)

if form.is\_valid():

print('Data is Valid')

form.save()

messages.success(request, 'You have been successfully registered')

form = UserRegistrationForm()

return render(request, 'UsersRegister.html', {'form': form})

else:

messages.success(request, 'Email or Mobile Already Existed')

print("Invalid form")

else:

form = UserRegistrationForm()

return render(request, 'UsersRegister.html', {'form': form})

def UserLoginCheck(request):

if request.method == "POST":

loginid = request.POST.get('loginname')

pswd = request.POST.get('pswd')

print("Login ID = ", loginid, ' Password = ', pswd)

try:

check = UserRegistrationModel.objects.get(loginid=loginid, password=pswd)

status = check.status

print('Status is = ', status)

if status == "activated":

request.session['id'] = check.id

request.session['loggeduser'] = check.name

request.session['loginid'] = loginid

request.session['email'] = check.email

print("User id At", check.id, status)

return render(request, 'users/UserHome.html', {})

else:

messages.success(request, 'Your Account Not at activated')

return render(request, 'UserLogin.html')

# return render(request, 'user/userpage.html',{})

except Exception as e:

print('Exception is ', str(e))

pass

messages.success(request, 'Invalid Login id and password')

return render(request, 'UserLogin.html', {})

def UserHome(request):

return render(request, 'users/UserHome.html', {})

def UserAutoMLTest(request):

obj = StartProcessAutoML()

html = ''

data1=''

try:

pass

lb = obj.startDataPreprocess()

data\_as\_df = h2o.as\_list(lb)

html = data\_as\_df.to\_html()

#data1 = data.to\_html()

except Exception as e:

pass

data\_list = AutoMLDataModel.objects.all()

#print("Lb type is ",type(lb))

return render(request,"users/AutoMachineLearning.html",{"html":html,"dataset":data\_list})

#return HttpResponse("Exit code 0")

#return redirect('AutoResponse')

def AutoResponse(request):

data\_list = AutoMLDataModel.objects.all()

page = request.GET.get('page', 1)

paginator = Paginator(data\_list, 10)

try:

users = paginator.page(page)

except PageNotAnInteger:

users = paginator.page(1)

except EmptyPage:

users = paginator.page(paginator.num\_pages)

return render(request, 'users/AutoMachineLearning.html', {'users': users})

def DataUploadForm(request):

return render(request,'users/useruploaddata.html',{})

def UploadDatatoServer(request):

AutoMLDataModel

# declaring template

template = "users/useruploaddata.html"

data = AutoMLDataModel.objects.all()

# prompt is a context variable that can have different values depending on their context

prompt = {

'order': 'Order of the CSV should be name, email, address, phone, profile',

'profiles': data

}

# GET request returns the value of the data with the specified key.

if request.method == "GET":

return render(request, template, prompt)

csv\_file = request.FILES['file']

# let's check if it is a csv file

if not csv\_file.name.endswith('.csv'):

messages.error(request, 'THIS IS NOT A CSV FILE')

data\_set = csv\_file.read().decode('UTF-8')

# setup a stream which is when we loop through each line we are able to handle a data in a stream

io\_string = io.StringIO(data\_set)

next(io\_string)

for column in csv.reader(io\_string, delimiter='\t', quotechar="|"):

print("Data is = ",column[0])

\_, created = AutoMLDataModel.objects.update\_or\_create(

Age=column[1],

Workclass=column[2],

EducationNum=column[3],

MaritalStatus=column[4],

Occupation=column[5],

Relationship=column[6],

Race=column[7],

Sex=column[8],

CapitalGain=column[9],

CapitalLoss=column[10],

Hoursperweek=column[11],

Country=column[12]

)

context = {}

return render(request, 'users/useruploaddata.html', context)

def UploadDatatoServerForPredections(request):

csv\_file = request.FILES['file']

# let's check if it is a csv file

if not csv\_file.name.endswith('.csv'):

messages.error(request, 'THIS IS NOT A CSV FILE')

data\_set = csv\_file.read().decode('UTF-8')

# setup a stream which is when we loop through each line we are able to handle a data in a stream

io\_string = io.StringIO(data\_set)

next(io\_string)

for column in csv.reader(io\_string, delimiter=',', quotechar="|"):

print("Data is = ", column[0])

\_, created = MyPredectionsModels.objects.update\_or\_create(

YearsExperience=column[0],

Salary=column[1]

)

context = {}

return render(request, 'users/useruploaddata.html', context)

def MyPredectionsSlot1(request):

data = MyPredectionsModels.objects.all()

return render(request,'users/MyPredections.html',{'data':data})

def MyPredectionsSlot2(request):

data = MyPredectionsModels.objects.all()

return render(request, 'users/DataSlot1.html', {'data': data})

def MyPredectionsSlot3(request):

if request.method=='POST':

splitsize = int(request.POST.get('testsize'))

testsize = splitsize/100

data = MyPredectionsModels.objects.all()

dataset = read\_frame(data)

X = dataset.iloc[:, :1].values

y = dataset.iloc[:, -1].values

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=testsize, random\_state=0)

#print('X\_train', X\_train)

#print('X\_test', X\_test)

#print('y\_train', y\_train)

from sklearn.linear\_model import LinearRegression

model = LinearRegression()

model.fit(X\_train, y\_train)

plt.scatter(X\_test, y\_test, color='red')

plt.plot(X\_train, model.predict(X\_train), color='blue')

plt.title('Salary vs Experience (Test set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.show()

score = model.score(X, y)

loginid = request.session['loginid']

email = request.session['email']

ModelPredectionStoreModels.objects.create(username=loginid, email=email, acheiveaccuracy=score,testsize=testsize)

y\_pred = model.predict(X\_test)

y\_pred = np.around(y\_pred, 1)

print('predected Result ', type(y\_pred.tolist()))

print('Original salary ', type(y\_test))

myDict = {'original':y\_test.tolist(),'predections':y\_pred.tolist()}

print("My Dict ",myDict)

return render(request,'users/DataSlot2.html',{'data':myDict})

#return HttpResponse(html)

def AddDataForm(request):

data = MyPredectionsModels.objects.all()

return render(request, 'users/AddDataForm.html', {'data': data})

def AddDataToDataset(request):

if request.method=='POST':

exp = request.POST.get('Experience')

salary = request.POST.get('salary')

MyPredectionsModels.objects.create(YearsExperience=exp,Salary=salary)

data = MyPredectionsModels.objects.all()

return render(request, 'users/DataSlot1.html', {'data': data})

UserAutoMachineLearningProcess.py

import sklearn

import pandas as pd

import numpy as np

import shap

import h2o

from h2o.automl import H2OAutoML

#h2o.init(nthreads=-1)

class H2OProbWrapper:

def \_\_init\_\_(self, h2o\_model, feature\_names):

self.h2o\_model = h2o\_model

self.feature\_names = feature\_names

def predict\_binary\_prob(self, X):

if isinstance(X, pd.Series):

X = X.values.reshape(1,-1)

self.dataframe= pd.DataFrame(X, columns=self.feature\_names)

self.predictions = self.h2o\_model.predict(h2o.H2OFrame(self.dataframe)).as\_data\_frame().values

return self.predictions.astype('float64')[:,-1] #probability of True class

class StartProcessAutoML:

def startDataPreprocess(self):

h2o.init(nthreads=-1) ### Start the h20 Server

X, y = shap.datasets.adult()

X\_display, y\_display = shap.datasets.adult(display=True)

#X\_display.to\_csv("output.csv", index=False)

#print(y\_display.shape)

print(X.head())

#print(X\_display.dtypes)

X\_train, X\_test, y\_train, y\_test = sklearn.model\_selection.train\_test\_split(\*shap.datasets.adult(),

test\_size=0.2, random\_state=7)

train\_indices = X\_train.index

test\_indices = X\_test.index

X\_train\_display = X\_display.iloc[train\_indices]

y\_train\_display = y\_display[train\_indices]

X\_test\_display = X\_display.iloc[test\_indices]

y\_test\_display = y\_display[test\_indices]

X\_train.reset\_index(drop=True, inplace=True)

X\_test.reset\_index(drop=True, inplace=True)

X\_train\_display.reset\_index(drop=True, inplace=True)

X\_test\_display.reset\_index(drop=True, inplace=True)

train\_h2o\_df = h2o.H2OFrame(X\_train)

train\_h2o\_df['labels'] = h2o.H2OFrame(y\_train)

train\_h2o\_df['labels'] = train\_h2o\_df['labels'].asfactor()

train\_h2o\_df['Workclass'] = train\_h2o\_df['Workclass'].asfactor()

train\_h2o\_df['Marital Status'] = train\_h2o\_df['Marital Status'].asfactor()

train\_h2o\_df['Relationship'] = train\_h2o\_df['Relationship'].asfactor()

train\_h2o\_df['Occupation'] = train\_h2o\_df['Occupation'].asfactor()

train\_h2o\_df['Sex'] = train\_h2o\_df['Sex'].asfactor()

train\_h2o\_df['Race'] = train\_h2o\_df['Race'].asfactor()

train\_h2o\_df['Country'] = train\_h2o\_df['Country'].asfactor()

test\_h2o\_df = h2o.H2OFrame(X\_test)

test\_h2o\_df['labels'] = h2o.H2OFrame(y\_test)

test\_h2o\_df['labels'] = test\_h2o\_df['labels'].asfactor()

test\_h2o\_df['Workclass'] = test\_h2o\_df['Workclass'].asfactor()

test\_h2o\_df['Marital Status'] = test\_h2o\_df['Marital Status'].asfactor()

test\_h2o\_df['Relationship'] = test\_h2o\_df['Relationship'].asfactor()

test\_h2o\_df['Occupation'] = test\_h2o\_df['Occupation'].asfactor()

test\_h2o\_df['Sex'] = test\_h2o\_df['Sex'].asfactor()

test\_h2o\_df['Race'] = test\_h2o\_df['Race'].asfactor()

test\_h2o\_df['Country'] = test\_h2o\_df['Country'].asfactor()

feature\_names = list(X\_train.columns)

aml = H2OAutoML(max\_runtime\_secs=50, seed=2)

#aml = H2OAutoML(max\_runtime\_secs=500, seed=42)

aml.train(x=feature\_names, y='labels', training\_frame=train\_h2o\_df)

lb = aml.leaderboard

print(lb)

bst\_model = aml.leader

h2o\_wrapper = H2OProbWrapper(bst\_model, feature\_names)

X\_train.shape[0]

explainer = shap.KernelExplainer(h2o\_wrapper.predict\_binary\_prob, X\_train.iloc[:100, :])

person = 0 # first person in test dataset

print('prediction (probability that this person earns more than $50k/year) =',

h2o\_wrapper.predict\_binary\_prob(X\_test.iloc[person])[0])

print('ground\_truth (this person earns more than $50k/year) =', y\_test\_display[person])

shap.initjs()

shap\_values = explainer.shap\_values(X\_test.iloc[person, :], nsamples=500)

shap.force\_plot(explainer.expected\_value, shap\_values, X\_test\_display.iloc[person, :])

person = 1 # second person in test dataset

print('prediction (probability that this person earns more than $50k/year) =',

h2o\_wrapper.predict\_binary\_prob(X\_test.iloc[person])[0])

print('ground\_truth (this person earns more than $50k/year) =', y\_test\_display[person])

shap.initjs()

shap\_values = explainer.shap\_values(X\_test.iloc[person, :], nsamples=500)

shap.force\_plot(explainer.expected\_value, shap\_values, X\_test\_display.iloc[person, :])

'''

h2o.save\_model(bst\_model)

X\_test.to\_pickle('X\_test.pkl')

X\_train.to\_pickle('X\_train.pkl')

np.save('y\_test.npy', y\_test)

np.save('y\_train.npy', y\_train)

X\_test\_display.to\_pickle('X\_test\_display.pkl')

X\_train\_display.to\_pickle('X\_train\_display.pkl')

np.save('y\_test\_display.npy', y\_test\_display)

np.save('y\_train\_display.npy', y\_train\_display)

X.to\_pickle('X.pkl')'''

#h2o.cluster().shutdown()

return lb

models.py

from django.db import models

# Create your models here.

# Create your models here.

class UserRegistrationModel(models.Model):

name = models.CharField(max\_length=100)

loginid = models.CharField(unique=True, max\_length=100)

password = models.CharField(max\_length=100)

mobile = models.CharField(unique=True, max\_length=100)

email = models.CharField(unique=True, max\_length=100)

locality = models.CharField(max\_length=100)

address = models.CharField(max\_length=1000)

city = models.CharField(max\_length=100)

state = models.CharField(max\_length=100)

status = models.CharField(max\_length=100)

def \_\_str\_\_(self):

return self.loginid

class Meta:

db\_table = 'AutoUsers'

class AutoMLDataModel(models.Model):

id =models.IntegerField(primary\_key=True)

Age = models.FloatField(default=0)

Workclass = models.CharField(max\_length=200)

EducationNum = models.FloatField(default=0)

MaritalStatus = models.CharField(max\_length=200)

Occupation = models.CharField(max\_length=200)

Relationship = models.CharField(max\_length=200)

Race = models.CharField(max\_length=200)

Sex = models.CharField(max\_length=200)

CapitalGain= models.FloatField(default=0)

CapitalLoss = models.FloatField(default=0)

Hoursperweek = models.FloatField(default=0)

Country= models.CharField(max\_length=200)

#def \_\_str\_\_(self):

#return self.id

class Meta:

db\_table = 'automldata'

class MyPredectionsModels(models.Model):

id = models.IntegerField(primary\_key=True)

YearsExperience = models.FloatField(default=0)

Salary = models.FloatField(default=0)

def \_\_str\_\_(self):

return self.id

class Meta:

db\_table = 'mypredections'

class ModelPredectionStoreModels(models.Model):

id = models.IntegerField(primary\_key=True)

username = models.CharField(max\_length=150)

email = models.CharField(max\_length=150)

acheiveaccuracy = models.FloatField()

testsize = models.FloatField()

cdata = models.DateTimeField(auto\_now\_add=True)

forms.py

from django import forms

from .models import UserRegistrationModel

class UserRegistrationForm(forms.ModelForm):

name = forms.CharField(widget=forms.TextInput(attrs={'pattern':'[a-zA-Z]+'}), required=True,max\_length=100)

loginid = forms.CharField(widget=forms.TextInput(attrs={'pattern':'[a-zA-Z]+'}), required=True,max\_length=100)

password = forms.CharField(widget=forms.PasswordInput(attrs={'pattern':'(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{8,}','title':'Must contain at least one number and one uppercase and lowercase letter, and at least 8 or more characters'}), required=True,max\_length=100)

mobile = forms.CharField(widget=forms.TextInput(attrs={'pattern':'[56789][0-9]{9}'}), required=True,max\_length=100)

email = forms.CharField(widget=forms.TextInput(attrs={'pattern':'[a-z0-9.\_%+-]+@[a-z0-9.-]+\.[a-z]{2,}$'}), required=True,max\_length=100)

locality = forms.CharField(widget=forms.TextInput(), required=True,max\_length=100)

address = forms.CharField(widget=forms.Textarea(attrs={'rows':4, 'cols': 22}), required=True,max\_length=250)

city = forms.CharField(widget=forms.TextInput(attrs={'autocomplete': 'off','pattern':'[A-Za-z ]+', 'title':'Enter Characters Only '}), required=True,max\_length=100)

state = forms.CharField(widget=forms.TextInput(attrs={'autocomplete': 'off','pattern':'[A-Za-z ]+', 'title':'Enter Characters Only '}), required=True,max\_length=100)

status = forms.CharField(widget=forms.HiddenInput(), initial='waiting' ,max\_length=100)

class Meta():

model = UserRegistrationModel

fields='\_\_all\_\_'

urls.py

"""AutomatedMachineLeanring URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:

https://docs.djangoproject.com/en/2.0/topics/http/urls/

Examples:

Function views

1. Add an import: from my\_app import views

2. Add a URL to urlpatterns: path('', views.home, name='home')

Class-based views

1. Add an import: from other\_app.views import Home

2. Add a URL to urlpatterns: path('', Home.as\_view(), name='home')

Including another URLconf

1. Import the include() function: from django.urls import include, path

2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))

"""

from django.contrib import admin

from django.urls import path

from AutomatedMachineLeanring import views as mainView

from users import views as usr

from admins import views as admins

urlpatterns = [

path('admin/', admin.site.urls),

path('',mainView.index, name='index'),

path('logout/',mainView.logout, name='logout'),

path('UserLogin/',mainView.UserLogin, name='UserLogin'),

path('AdminLogin/',mainView.AdminLogin, name='AdminLogin'),

path('UserRegister/',mainView.UserRegister, name='UserRegister'),

### User Side Views ####

path('UserRegisterActions/', usr.UserRegisterActions, name='UserRegisterActions'),

path('UserLoginCheck/', usr.UserLoginCheck, name='UserLoginCheck'),

path('UserHome/', usr.UserHome, name='UserHome'),

path('UserAutoMLTest/', usr.UserAutoMLTest, name='UserAutoMLTest'),

path('UploadDatatoServer/',usr.UploadDatatoServer, name='UploadDatatoServer'),

path('DataUploadForm/', usr.DataUploadForm, name='DataUploadForm'),

path('AutoResponse/', usr.AutoResponse, name='AutoResponse'),

path('UploadDatatoServerForPredections/', usr.UploadDatatoServerForPredections, name='UploadDatatoServerForPredections'),

path('MyPredectionsSlot1/', usr.MyPredectionsSlot1, name='MyPredectionsSlot1'),

path('MyPredectionsSlot2/', usr.MyPredectionsSlot2, name='MyPredectionsSlot2'),

path('MyPredectionsSlot3/', usr.MyPredectionsSlot3, name='MyPredectionsSlot3'),

path('AddDataToDataset/', usr.AddDataToDataset, name='AddDataToDataset'),

path('AddDataForm/', usr.AddDataForm, name='AddDataForm'),

### Admin Side Urls ####

path('AdminLoginCheck/', admins.AdminLoginCheck, name='AdminLoginCheck'),

path('AdminHome/', admins.AdminHome, name='AdminHome'),

path('ViewUsersList/', admins.ViewUsersList, name='ViewUsersList'),

path('AdminActivaUsers/', admins.AdminActivaUsers, name='AdminActivaUsers'),

path('UserPerformedOperations/', admins.UserPerformedOperations, name='UserPerformedOperations'),

]

Adminside views.py

from django.shortcuts import render,HttpResponse

from django.contrib import messages

from users.forms import UserRegistrationModel

from users.models import ModelPredectionStoreModels

# Create your views here.

def AdminLoginCheck(request):

if request.method == 'POST':

usrid = request.POST.get('loginname')

pswd = request.POST.get('pswd')

print("User ID is = ", usrid)

if usrid == 'admin' and pswd == 'admin':

return render(request, 'admins/AdminHome.html')

else:

messages.success(request, 'Please Check Your Login Details')

return render(request, 'AdminLogin.html', {})

def AdminHome(request):

return render(request,'admins/AdminHome.html',{})

def ViewUsersList(request):

data = UserRegistrationModel.objects.all()

return render(request, 'admins/RegisteredUsers.html',{'data':data})

def AdminActivaUsers(request):

if request.method == 'GET':

id = request.GET.get('uid')

status = 'activated'

print("PID = ", id, status)

UserRegistrationModel.objects.filter(id=id).update(status=status)

data = UserRegistrationModel.objects.all()

return render(request,'admins/RegisteredUsers.html',{'data':data})

def UserPerformedOperations(request):

data = ModelPredectionStoreModels.objects.all()

return render(request, 'admins/UsersOperations.html', {'data': data})

userbase.html

{%load static%}

<!DOCTYPE html>

<html lang="en">

<title>AutoML Template</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

<style>

body {font-family: "Lato", sans-serif}

.mySlides {display: none}

</style>

<body>

<!-- Navbar -->

<div class="w3-top">

<div class="w3-bar w3-black w3-card">

<a class="w3-bar-item w3-button w3-padding-large w3-hide-medium w3-hide-large w3-right" href="javascript:void(0)" onclick="myFunction()" title="Toggle Navigation Menu"><i class="fa fa-bars">Admin</i></a>

<a href="{%url 'UserHome'%}" class="w3-bar-item w3-button w3-padding-large">Home</a>

<a href="{%url 'UserAutoMLTest'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">Auto ML</a>

<a href="{%url 'MyPredectionsSlot1'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">Data</a>

<a href="{%url 'MyPredectionsSlot2'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">Predections</a>

<a href="{%url 'AddDataForm'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">Add Data</a>

<a href="{%url 'logout'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">logout</a>

<!--<a href="{%url 'DataUploadForm'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">Upload</a>-->

<!--<a href="{%url 'DataUploadForm'%}" class="w3-bar-item w3-button w3-padding-large w3-hide-small">My Pred Upload</a>-->

<a href="javascript:void(0)" class="w3-padding-large w3-hover-red w3-hide-small w3-right"><h5> Automated Machine Learning: The New Wave of

Machine Learning</h5></a>

</div>

</div>

<!-- Page content -->

<div class="w3-content" style="max-width:2000px;margin-top:46px">

<!-- Automatic Slideshow Images -->

<!-- The Band Section -->

{%block contents%}

{%endblock%}

<!-- End Page Content -->

</div>

<!-- Image of location/map -->

<img src="{%static 'images/map.jpg'%}" class="w3-image w3-greyscale-min" style="width:100%">

<!-- Footer -->

<footer class="w3-container w3-padding-64 w3-center w3-opacity w3-light-grey w3-xlarge">

<i class="fa fa-facebook-official w3-hover-opacity"></i>

<i class="fa fa-instagram w3-hover-opacity"></i>

<i class="fa fa-snapchat w3-hover-opacity"></i>

<i class="fa fa-pinterest-p w3-hover-opacity"></i>

<i class="fa fa-twitter w3-hover-opacity"></i>

<i class="fa fa-linkedin w3-hover-opacity"></i>

<p class="w3-medium">Powered by <a href="#" target="\_blank">Alex Corporation</a></p>

</footer>

<script>

// Automatic Slideshow - change image every 4 seconds

var myIndex = 0;

carousel();

function carousel() {

var i;

var x = document.getElementsByClassName("mySlides");

for (i = 0; i < x.length; i++) {

x[i].style.display = "none";

}

myIndex++;

if (myIndex > x.length) {myIndex = 1}

x[myIndex-1].style.display = "block";

setTimeout(carousel, 4000);

}

// Used to toggle the menu on small screens when clicking on the menu button

function myFunction() {

var x = document.getElementById("navDemo");

if (x.className.indexOf("w3-show") == -1) {

x.className += " w3-show";

} else {

x.className = x.className.replace(" w3-show", "");

}

}

// When the user clicks anywhere outside of the modal, close it

var modal = document.getElementById('ticketModal');

window.onclick = function(event) {

if (event.target == modal) {

modal.style.display = "none";

}

}

</script>

</body>

</html>

Add Data to Form

{% extends 'users/userbase.html'%}

{%block contents%}

<div class="w3-container w3-content w3-center w3-padding-64" id="band" style="max-width:800px">

<h2 class="w3-wide">Perform Predections Analysis</h2>

<p class="w3-opacity"><i>We love Machine Learning</i></p>

<p class="w3-justify">

<center>

<form action="{%url 'AddDataToDataset'%}" method="post">

{%csrf\_token%}

<label>Enter Experience</label>&nbsp;&nbsp;&nbsp;

<input type="number" required name="Experience" min="0" max="20" step=".01"> <strong>Like Training and testing Split(30,40,.....90)</strong>

<br/><br/>

<label>Enter Salary </label>&nbsp;&nbsp;&nbsp;

<input type="number" name="salary" required min="5000" max="100000" style="size:90;">

<br/><br/>

<button type="submit" name="Test">Add Data</button>

</form>

<table class="table table-bordered bg-light text-dark">

<thead>

<tr>

<th>S.No</th>

<th>Years of Experience</th>

<th>Salary</th>

</tr>

</thead>

<tbody>

{% for i in data %}

<tr style="color: Black">

<td>{{forloop.counter}}</td>

<td>{{i.YearsExperience}}</td>

<td>{{i.Salary}}</td>

</tr>

{% endfor %}

</tbody>

</table>

</center>

</h2>

</form>

</p>

<div class="w3-row w3-padding-32">

<div class="w3-third">

<p>Data Imputation</p>

<p>Often datasets, in reality, may contain missing values for some different reasons The randomness of MCAR data is high enough that there is no overall bias towards any particular class, unlike MAR data, which are responsible for causing an increase in bias</p>

</div>

<div class="w3-third">

<p>Data Balancing</p>

<p>Data imbalance is a condition when one or more classes in a categorical dataset have higher observations than the rest of the classes The sample handling approach for data balancing will preprocess the training set to minimize class differences, and this issue can be resolved.</p>

</div>

<div class="w3-third">

<p>Data Encoding</p>

<p>To make the data human-readable, the training data is often labelled in words. Data Encoding refers to converting the provided feature labels into numerical form to allow computer machines to interpret them.</p>

</div>

</div>

</div>

{%endblock%}

AdminLogin.html

{% extends 'base.html'%}

{%block contents%}

<div class="w3-container w3-content w3-center w3-padding-64" style="max-width:800px" id="band">

<h2 class="w3-wide">Admin Login Form</h2>

<p class="w3-opacity"><i>Be touch with us</i></p>

<p class="w3-justify ">

<center>

<div class="container">

<div class="row">

<div class="col-sm-9 col-md-7 col-lg-5 mx-auto">

<div class="card card-signin my-5">

<div class="card-body">

<h5 class="card-title text-center">Sign In</h5>

<form action="{%url 'AdminLoginCheck'%}" method="post" class="form-signin">

{%csrf\_token%}

<div class="form-label-group">

<input type="text" id="inputEmail" name="loginname" class="form-control" placeholder="Enter Login Id"

required autofocus>

<label for="inputEmail">Enter Login ID</label>

</div>

<div class="form-label-group">

<input type="password" name="pswd" id="inputPassword" class="form-control"

placeholder="Enter Password" required>

<label for="inputPassword">Enter Password</label>

</div>

<div class="custom-control custom-checkbox mb-3">

</div>

<button class="btn btn-lg btn-primary btn-block text-uppercase" type="submit">Sign in

</button>

<hr class="my-4">

</form>

{% if messages %}

{% for message in messages %}

<font color='RED'> {{ message }}</font>

{% endfor %}

{% endif %}

</div>

</div>

</div>

</div>

</div>

</center>

</p>

<div class="w3-row w3-padding-32">

<div class="w3-third">

<p>Data Imputation</p>

<p>Often datasets, in reality, may contain missing values for some different reasons The randomness of MCAR

data is high enough that there is no overall bias towards any particular class, unlike MAR data, which

are responsible for causing an increase in bias</p>

</div>

<div class="w3-third">

<p>Data Balancing</p>

<p>Data imbalance is a condition when one or more classes in a categorical dataset have higher observations

than the rest of the classes The sample handling approach for data balancing will preprocess the

training set to minimize class differences, and this issue can be resolved.</p>

</div>

<div class="w3-third">

<p>Data Encoding</p>

<p>To make the data human-readable, the training data is often labelled in words. Data Encoding refers to

converting the provided feature labels into numerical form to allow computer machines to interpret

them.</p>

</div>

</div>

</div>

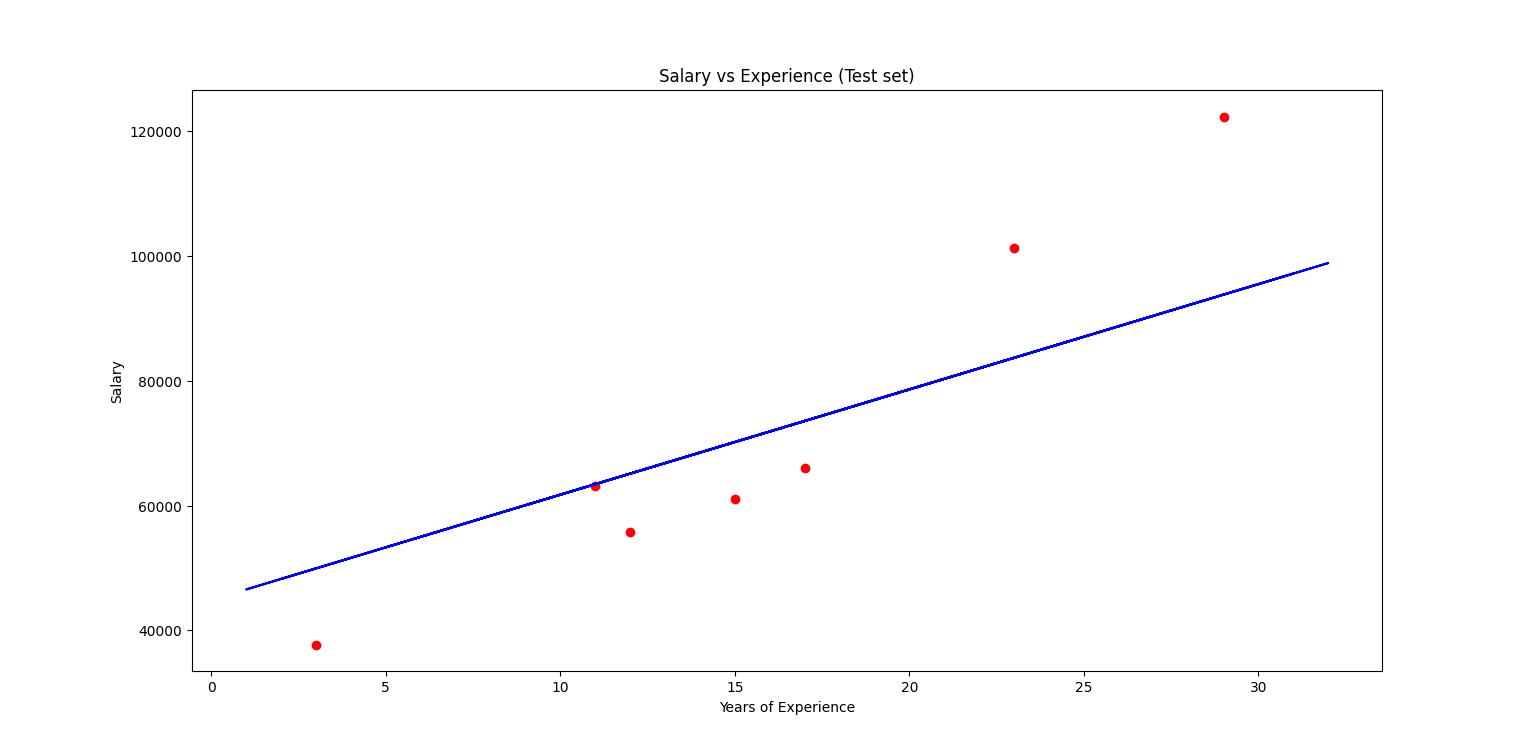
{%endblock%}

**4.6 RESULT ANALYSIS**

Data analysis is really important for making better decisions, improving business models, and refining products. However, it can be challenging because building and training machine learning models is complex and needs a lot of expertise and resources. Automated Machine Learning (AutoML) helps with these challenges by automating the whole machine learning process and breaking it down into simpler steps. This automation is especially useful as machine learning models become more complex, particularly in fields like image processing and Natural Language Processing, which require significant memory, powerful GPUs, and extensive training time.

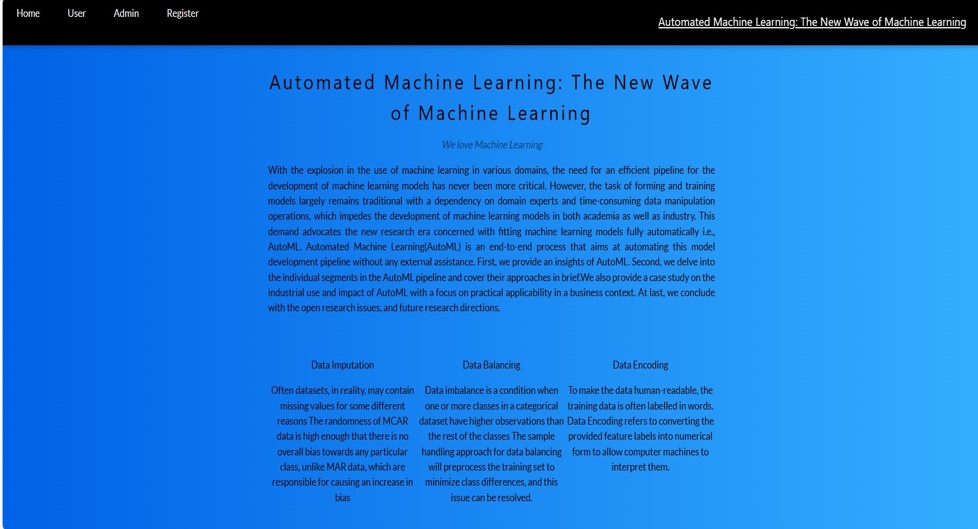
One major issue AutoML addresses is the lack of a formal method for setting model parameters, which traditionally involved a lot of trial and error. AutoML automates this process using techniques such as Grid Search, Random Search, Bayesian Optimization, and Hyperband to find the best parameters quickly. It also supports automated feature engineering and model selection through methods like Recursive Feature Elimination, Principal Component Analysis, and ensemble learning. These advancements have shown great results, even outperforming human experts in machine learning competitions like Kaggle.

AutoML’s benefits go beyond traditional fields and extend to emerging areas like autonomic cloud computing, intelligent vehicular networks, blockchain, and software-defined networking. The paper reviews recent progress in AutoML, analyzing its various components such as data preprocessing, feature engineering, model selection, and hyperparameter tuning. It also evaluates the latest AutoML tools and explores how advancements in machine learning often complement or even surpass deep learning techniques.



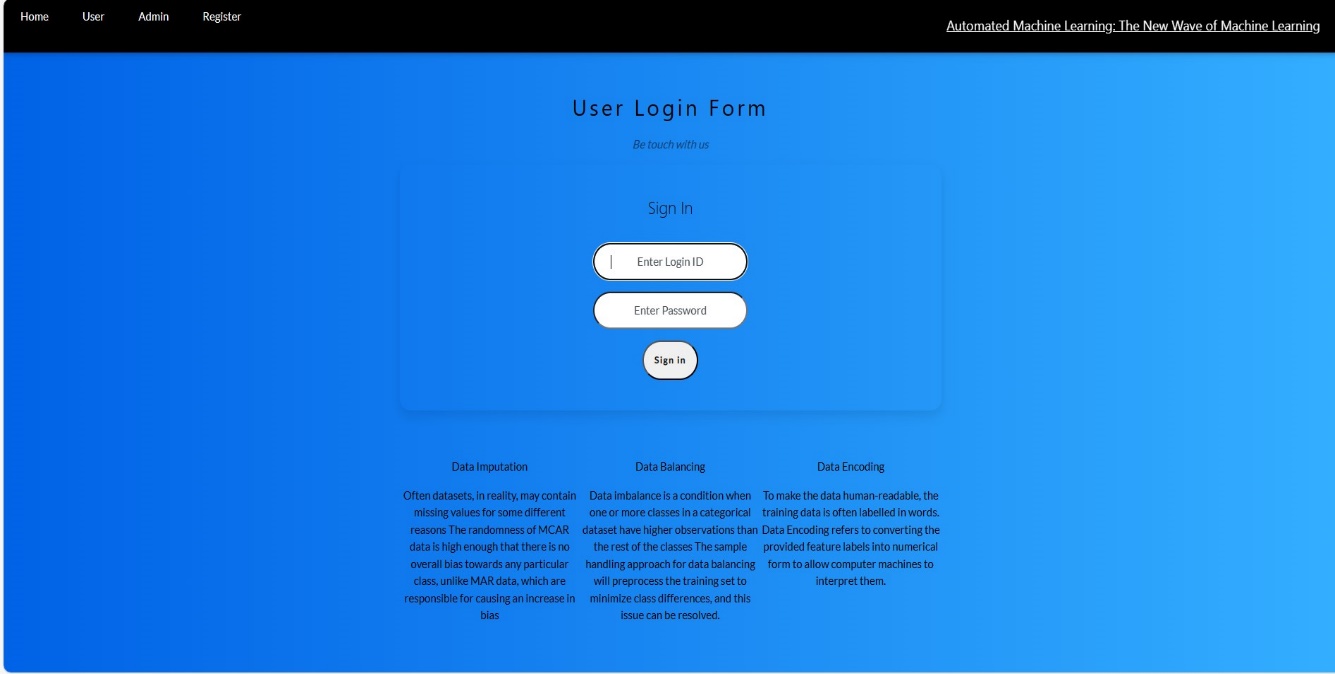
4.5: Result Analysis

**5. SCREENSHOTS**



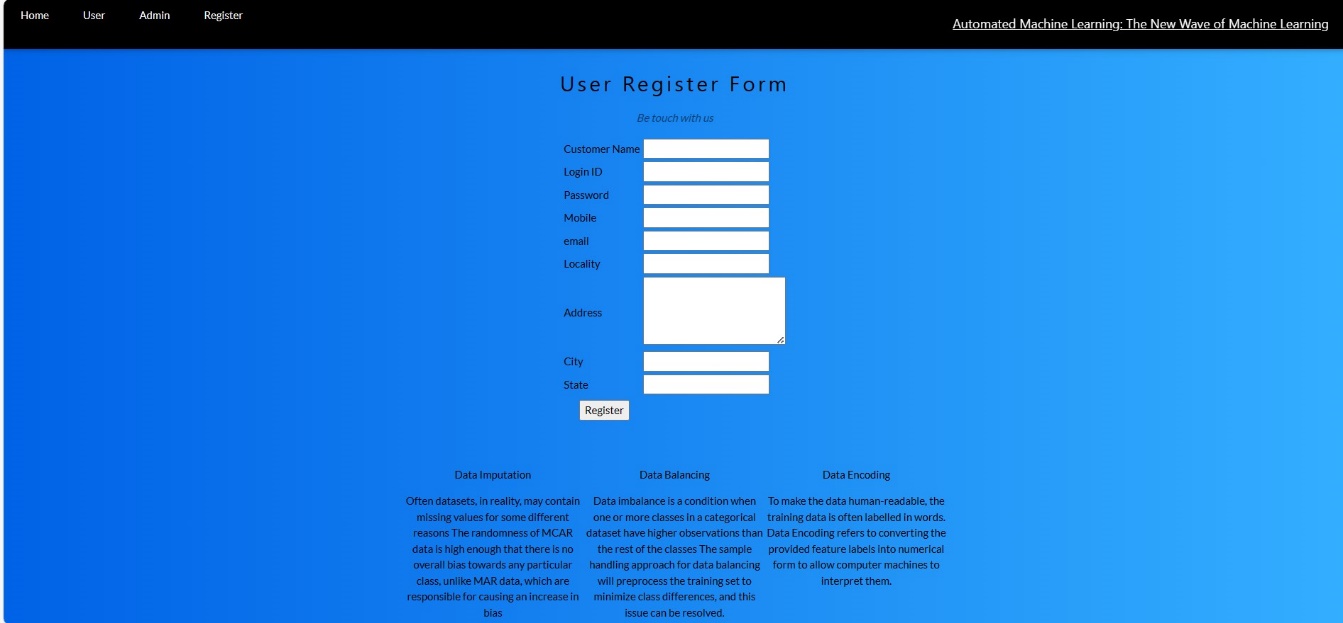
5.1: Home page

The project's homepage interface serves as the gateway for users, offering a seamless login experience. Users input their credentials in designated fields, ensuring secure access to the platform. With a focus on user-friendly design and robust security measures, the interface sets the stage for a positive user interaction.



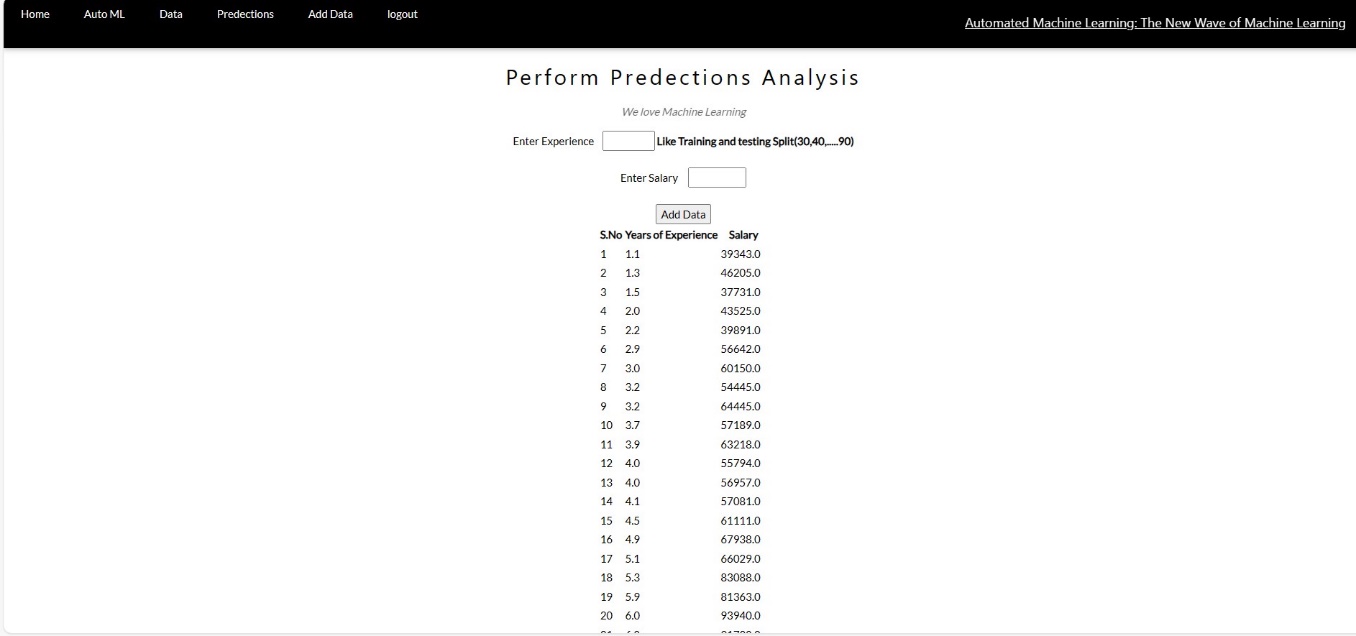
5.2: User login page

The service provider login page facilitates secure access for providers using their credentials. Users enter their login details in the designated fields, ensuring a streamlined and authenticated experience. With a focus on security and user-friendly design, the interface enhances the service provider's login process.



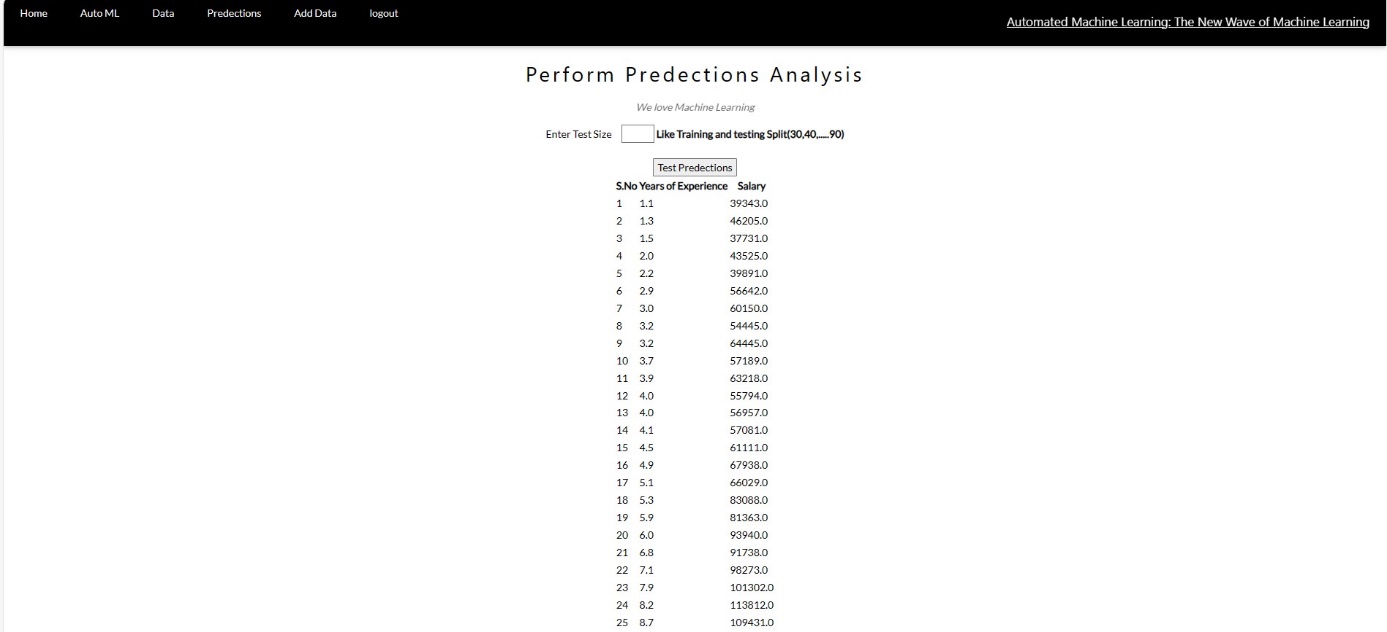
5.3: User register page

The user registration page allows new users to sign up by providing necessary details. Users input their information in the designated fields, ensuring a straightforward and secure registration process. With an emphasis on simplicity and data protection, the interface enhances the user's experience during registration.



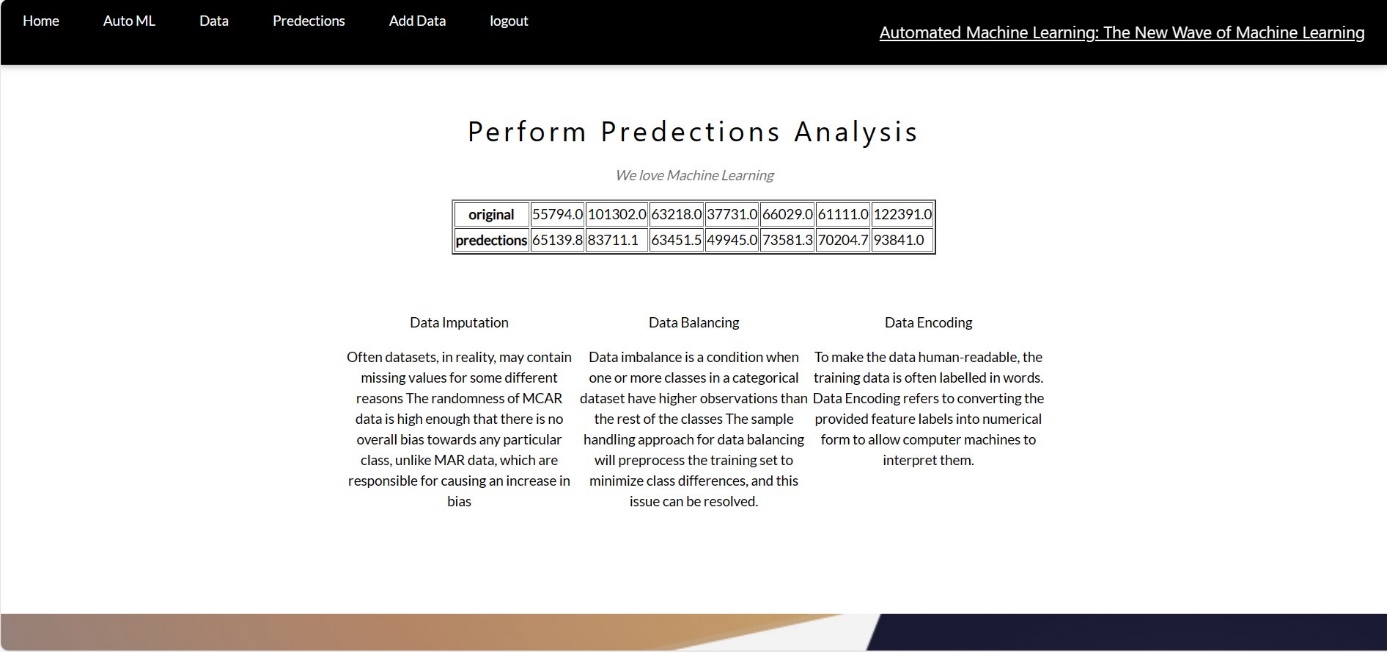
5.4: Adding data to be tested page

The Add Data page enables users to input new data. The user can add as much as data which needs to be tested.



5.5: Perform predections analysis page

The Perform predections analysis page enables the user to enter the test size for performing prediction.



5.6: Test results page

The Test results page provides insights into the accuracy of the algorithm utilized in our Automated Machine Learning The New Wave Of Machine learning. It presents the outcomes of original data and predicted data, offering a comprehensive view of the algorithm's performance.

**6. TESTING**

#### 

#### 6. TESTING

##### **INTRODUCTION TO TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

##### **TYPES OF TESTING**

###### **UNIT TESTING**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

###### **INTEGRATION TESTING**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

###### **FUNCTIONAL TESTING**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid input : identified classes of valid input must be accepted.

Invalid input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

###### **TEST CASES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.no** | **Test Case** | **Excepted Result** | **Result** | **Remarks(IF Fails)** |
| 1. | User Register | If User registration successfully. | Pass | If already user email exist then it fails. |
| 2. | User Login | If User name and password is correct then it will getting valid page. | Pass | Un Register Users will not logged in. |
| 3. | Start Data Pre process | Data preprocess will start from shap server | Pass | Itd depend upon bandwidth, no bandwith its failed |
| 4. | Start H2o Server | H2o Server will start when preprocess done | Pass | According to our python version and shap version its depend, if it starts it display url |
| 5. | Train the models | XGBoost Model try to load | Pass | It check available models for testing |
| 6. | Testing the models | Test Model Data generated | Pass | Test Model Data generated if no train no data |
| 7. | Files stored in local machine for further operations | Each files like training testing files will be pickled | Pass | All pickled file will store in the machine of model failed then failed |
| 8. | Predict Train and Test data | Predicted and original salary will be displayed | Pass | Trains and test size must be specify otherwise failed |
| 9. | Admin login | Admin can login with his login credential. If success he get his home page | Pass | Invalid login details will not allowed here |
| 10. | Admin can activate the register users | Admin can activate the register user id | Pass | If user id not found then it won’t login. |

6.3: TEST CASES

**7.CONCLUSION**

##### **7. CONCLUSION & FUTURE SCOPE**

##### **7.1 CONCLUSION**

In this paper, we provide insights to the readers about the various segments of AutoML with a conceptual perspective. Each of these segments has various approaches that have been briefly explained to provide a concise overview. We also discuss the various trends seen in recent years including suggestions of thirsty research areas which need attention.

##### 

##### **7.2 FUTURE SCOPE**

The future scope of the Automated Machine Learning (AutoML) project includes advancements in model interpretability, real-time data processing, and deployment on edge devices, enabling applications in IoT and smart cities. It aims to personalize AutoML solutions, integrate with emerging technologies like blockchain and quantum computing, and enhance cross-domain adaptability. Focus areas include ethical AI, automating the entire model lifecycle, and increasing accessibility through no-code platforms. Additionally, the project will improve collaboration features, ensure regulatory compliance, develop hybrid systems combining automation and human oversight, and create educational tools for broader adoption and understanding of machine learning.

**BIBLIOGRAPHY**

##### **BIBLIOGRAPHY**

**REFERENCES**

[1] Lukas Tuggener, Mohammadreza Amirian, Katharina Rombach, Stefan L¨orwald, Anastasia Varlet, Christian Westermann, and Thilo Stadelmann. Automated machine learning in practice: state of the art and recent results. In 2019 6th Swiss Conference on Data Science (SDS), pages 31–36. IEEE, 2019.

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[4] Avatar Jaykrushna, Pathik Patel, Harshal Trivedi, and Jitendra Bhatia. Linear regression assisted prediction based load balancer for cloud computing. In 2018 IEEE Punecon, pages 1–3. IEEE.

[5] Jitendra Bhatia, Ruchi Mehta, and Madhuri Bhavsar. Variants of software defined network (sdn) based load balancing in cloud computing: A quick review. In International Conference on Future Internet Technologies and Trends, pages 164–173. Springer, 2017.

[6] Ishan Mistry, Sudeep Tanwar, Sudhanshu Tyagi, and Neeraj Kumar. Blockchain for 5g-enabled iot for industrial automation: A systematic review, solutions, and challenges. Mechanical Systems and Signal Processing, 135:106382, 2020.

[7] Jitendra Bhatia, Yash Modi, Sudeep Tanwar, and Madhuri Bhavsar. Software defined vehicular networks: A comprehensive review. International Journal of Communication Systems, 32(12):e4005, 2019.

[8] Jitendra Bhatia, Ridham Dave, Heta Bhayani, Sudeep Tanwar, and Anand Nayyar. Sdn-based real-time urban traffic analysis in vanet environment. Computer Communications, 149:162 – 175, 2020.

[9] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. arXiv preprint arXiv:1908.00709, 2019.

[10] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. arXiv preprint arXiv:1906.02287, 2019.

[11] Anh Truong, Austin Walters, Jeremy Goodsitt, Keegan Hines, Bayan Bruss, and Reza Farivar. Towards automated machine learning: Evaluation and comparison of automl approaches and tools. arXiv preprint arXiv:1908.05557, 2019.

[12] Shichao Zhang, Chengqi Zhang, and Qiang Yang. Data preparation for data mining. Applied artificial intelligence, 17(5-6):375–381, 2003.

[13] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. IEEE Data Eng. Bull., 23(4):3–13, 2000.

[14] Dipali Shete and Sachin Bojewar. Auto approach for extracting relevant data using machine learning. International Journal of Electronics, 6:0, 2019.

[15] Carol M Musil, Camille B Warner, Piyanee Klainin Yobas, and Susan L Jones. A comparison of imputation techniques for handling missing data. Western Journal of Nursing Research, 24(7):815–829, 2002.

[16] RB Kline. Principles and practice of structural equation modeling. 1998. New York: Guilford, 1998.

[17] Joseph F Hair, Rolph E Anderson, Ronald L Tatham, and William C Black. Multivariate data analysis. englewood cliff. New Jersey, USA, 5(3):207–2019, 1998.

[18] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Practical automated machine learning for the automl challenge 2018. In International Workshop on Automatic Machine Learning at ICML, pages 1189–1232, 2018.

[19] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pages 785–794. ACM, 2016.

[20] Tpot: Skewed classes. https://github.com/EpistasisLab/tpot/blob/v0.9.5/ tpot/metrics.py. (Accessed: September 10, 2019).

[21] Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 847–855. ACM, 2013.

[22] Leo Breiman. Bagging predictors. Machine learning, 24(2):123–140, 1996.

[23] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In Workshop on learning from imbalanced datasets II, volume 11, pages 1–8. Citeseer, 2003.

[24] Mohamed Bekkar and Taklit Akrouf Alitouche. Imbalanced data learning approaches.

[25] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. Auto-sklearn: Efficient and robust automated machine learning. In Automated Machine Learning, pages 113–134. Springer, 2019.

[26] Ambika Kaul, Saket Maheshwary, and Vikram Pudi. Autolearnautomated feature generation and selection. In 2017 IEEE International Conference on Data Mining (ICDM), pages 217–226. IEEE, 2017.

[27] Udayan Khurana, Deepak Turaga, Horst Samulowitz, and Srinivasan Parthasrathy. Cognito: Automated feature engineering for supervised learning. In 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), pages 1304–1307. IEEE, 2016.

[28] Fatemeh Nargesian, Horst Samulowitz, Udayan Khurana, Elias B Khalil, and Deepak S Turaga. Learning feature engineering for classification. In IJCAI, pages 2529–2535, 2017.

[29] Udayan Khurana, Horst Samulowitz, and Deepak Turaga. Feature engineering for predictive modeling using reinforcement learning. In Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[30] Gilad Katz, Eui Chul Richard Shin, and Dawn Song. Explorekit: Automatic feature generation and selection. In 2016 IEEE 16th International Conference on Data Mining (ICDM), pages 979–984. IEEE, 2016.

[31] Hoang Thanh Lam, Johann-Michael Thiebaut, Mathieu Sinn, Bei Chen, Tiep Mai, and Oznur Alkan. One button machine for automating feature engineering in relational databases. arXiv preprint arXiv:1706.00327, 2017.

[32] Mohamed Maher and Sherif Sakr. Smartml: A meta learning-based framework for automated selection and hyperparameter tuning for machine learning algorithms. In EDBT: 22nd International Conference on Extending Database Technology, 2019.

[33] Steven M LaValle, Michael S Branicky, and Stephen R Lindemann. On the relationship between classical grid search and probabilistic roadmaps. The International Journal of Robotics Research, 23(7-8):673– 692, 2004.

[34] Francisco J Solis and Roger J-B Wets. Minimization by random search techniques. Mathematics of operations research, 6(1):19–30, 1981.

[35] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(Feb):281–305, 2012.

[36] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In International conference on learning and intelligent optimization, pages 507–523. Springer, 2011.

[37] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In International conference on machine learning, pages 2171–2180, 2015.

[38] Hector Mendoza, Aaron Klein, Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. Towards automatically-tuned neural networks. In Workshop on Automatic Machine Learning, pages 58–65, 2016.

[39] Dani Yogatama and Gideon Mann. Efficient transfer learning method for automatic hyperparameter tuning. In Artificial intelligence and statistics, pages 1077–1085, 2014.

[40] Randal S Olson and Jason H Moore. Tpot: A tree-based pipeline optimization tool for automating machine learning. In Automated Machine Learning, pages 151–160. Springer, 2019.

[41] Boyuan Chen, Harvey Wu, Warren Mo, Ishanu Chattopadhyay, and Hod Lipson. Autostacker: A compositional evolutionary learning system. In Proceedings of the Genetic and Evolutionary Computation Conference, pages 402–409. ACM, 2018.

[42] Jitendra Bhatia and Malaram Kumhar. Perspective study on load balancing paradigms in cloud computing. IJCSC, 6(1):112–120, 2015.

[43] Natalia Miloslavskaya and Alexander Tolstoy. Big data, fast data and data lake concepts. Procedia Computer Science, 88:300–305, 2016.

[44] Jitendra Bhagwandas Bhatia. A dynamic model for load balancing in cloud infrastructure. Nirma University Journal of Engineering and Technology (NUJET), 4(1):15, 2015.

[45] Jai Prakash Verma, Sudeep Tanwar, Sanjay Garg, Ishit Gandhi, and Nikita H Bachani. Evaluation of pattern based customized approach for stock market trend prediction with big data and machine learning techniques. International Journal of Business Analytics (IJBAN), 6(3):1– 15, 2019.

[6] Dataset link : <https://www.kaggle.com/datasets/rohankayan/years-of-experience-and-salary-dataset>

##### **GITHUB LINK**

[1] Project Code GitHub Link :