SUCCESSFUL ATTRIBUTES FOR CAREER OF STUDENTS USING AI TECHNOLOGY

A PROJECT REPORT

Submitted by

VAZHMUNI P (422323622022)

in partial fulfillment for the award of the degree of

MASTER OF COMPUTER APPLICATION

THIRUVALLUVAR COLLEGE OF ENGINEERING AND TECHNOLOGY, VANDAVASI-604505

ANNA UNIVERSITY: CHENNAI – 600025

JULY-2025

ANNA UNIVERSITY: CHENNAI – 600025

BONAFIDE CERTIFICATE

Certified that this project titled **"SUCCESSFUL ATTRIBUTES FOR CAREER OF STUDENTS USING AI TECHNOLOGY"** is the bonafide work of **VAZHMUNI P (422323622022)** who carried out the work under my supervision certified further that to best of my knowledge the work reported herein does not from part of any other project or dissertation on the basis of which a degree of award was conferred occasion on this or any other candidate

Prof.R.MALARVANNAN,MCA.,MPhil.,
HEAD OF THE DEPARTMENT
Department of MCA Thiruvalluvar College of College Engineering & Technology
Vandavasi – 604505.

Prof.R.MALARVANNAN,MCA.,MPhil.,
SUPERVISOR
Department of MCA Thiruvalluvar College of College Engineering & Technology
Vandavasi – 604505.

Date:10/07/2025
Place:Coimbatore

Project Completion Letter

To:

Head of the Department,
Department Master of Computer Applications,
Thiruvalluvar College of Engineering and Technology,
Vandavast-604505.

Respected Sir/Madam,

Sub: Project Completion Certificate.,

This is to certify that **Mr. P. VAZHMUNI (Reg. No: 422323622022),** pursuing Final Year MCA at **Thiruvalluvar College of Engineering and Technology** has successfully completed his Final Semester Project Work titled: **"Successful Attributes for Career of Students Using AI Technology"** in our organization Inmakes Infotech Pvt. Ltd., during the period from **February 2025 to July 2025.**

All the best
Sincerely,

mail@inmakes.com
www.inmakes.com
Inmakes Infotech Pvt.Ltd.
11/4, Kalapatti Main Rd, Indira Nagar, Civil Aerodrome Post, Coimbatore, Tamil Nadu 641014

VIVA - VOCE EXAMINATION

The Viva – Voce examination of the project work **"SUCCESSFUL ATTRIBUTES FOR CAREER OF STUDENTS USING AI TECHNOLOGY"** submitted by **VAZHMUNI P (422323622022),** held on……..

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

I express my sincere gratitude to our beloved Founder **Prof.Dr.S.ARUNACHALAM B.E.,M.sc.,Engg(Structures).,Ph.D (Structures & Foundation)., MIE., MISTE., MIWWA** for providing all the facilities and helped me to completed my project successfully.

My special gratitude to our respectful Chairman **Dr.A.GANESHKUMAR M.E., Ph.D.,** for rendering all the facilities and strengthening support to complete my project.

I express my sincere gratitude to our beloved Principal **Prof**. **Dr.S.K.PAZHANIVEL,M.Tech., Ph.D.,** for providing all the facilities and helped me to complete my project successfully.

I wish to place on record of my profound feeling of gratitude and heartfelt thanks to our Head of the Department **Dr.R.MALARVANNAN M.C.A., M.Phil.,** who has given me proper guidance and suggestion at all stages of this project work.

I am also thankful to guide **Dr.R.MALARVANNAN M.C.A., M.Phil.,** Asst Prof. Department of Computer Application who has given proper guidance and suggestions at all stages of this project.

I acknowledge the authors of those books which are referred to get the basic idea for my project work. Lastly, I would like to express out deep appreciation towards my classmates and my indebtedness to my mother, my father for providing the moral support and encouragement

(VAZHMUNI P)

ABSTRACT

Educational institutions aim to identify students at risk of academic failure early to improve learning outcomes. This project applies machine learning (ML) and deep learning (DL) techniques to analyze student data and predict academic success. By leveraging classification algorithms such as Naive Bayes, Decision Tree, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP), along with deep learning models like Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), the system enhances the accuracy of predictions. Methods for automatically identifying students in need of assistance have been studied for decades. Higher education institutions (HEIs) are often curious whether students will be successful or not during their study. Before or during their courses the academic institutions try to estimate the percentage of successful students. This study focuses on ways to examining successful attributes for undergraduate students using data mining techniques. The Student Education Data Mining works were organized as a raw data base. As result of the execution of classification processes, a set of educational functionalities was found, a realistic pattern of SEDM approaches was discovered. One key finding is most of the SEDM approaches are ground on a set composed by educational systems, test preparation, tasks, methods, and algorithms each. Student Educational Data Mining field concentrate on Prediction more often as compare to generate exact results for future purpose. The various machine learning algorithms and deep learning models are implemented and examining the attributes for undergraduate student. The prediction result based on accuracy, precision, recall, f1-score, specificity and sensitivity.

இயற்றுக்குறிப்பு

கல்வி நிறுவனங்கள் மாணவர்கள் கல்வியில் ததால்வியடைய வாய்ப்புள்ளவர்கடள முன்கணித்F, அவர்களF கல்வி முடிவுகடள தமம்படுத்த விரும்புகின்றன. இந்த திைம்ைத்தில், இயந்திர கற்றல் மற்றும் ஆழமான கற்றல் முடறகள் பயன்படுத்தப்படுகின்றன. மாணவர் ததாைர்புடைய தகவல்கடள ஆய்வு தெய்F, தவற்றி தபறக்கூடிய மாணவர்கடள கண்ைறிய வகுப்பூைம்டு முடறகள் வழியாக கணிக்கப்படுகிறF. இதில் நாய்தவ விதி, தீர்மான மரம், அருகிலுள்ள அண்டை முடற, ஆதரவு தவகுஜாதி கருவி, பன்மைங்கு அடுக்கு புரிதல் முடற தபான்றடவ, தமலும் தெயற்டக நரம்பியல் வடலப்பின்னல் மற்றும் சுருக்கத் ததாைர் நரம்பியல் வடலப்பின்னல்தபான்ற ஆழக்கற்றல்மாதிரிகள் பயன்படுத்தப்பைப்டுள்ளன.மாணவர்கள்உதவிக்குததடவப்படுப வர்கள் யார் என்படத தானாகக் கண்ைறியும் முடறகள் பல ஆண்டுகளாக ஆராயப்பைப்டுள்ளன. உயர்கல்வி நிறுவனங்கள், மாணவர்கள் தவற்றி தபறுமாரா என்படத ததரிந்F தகாள்ள முயற்சிக்கின்றன. இந்த ஆய்வு, பை்ைப்படிப்பு மாணவர்களின் தவற்றிக்கு காரணமான பண்புகடளத் தரவுத்தளவியல் முடறகள் மூலம் ஆய்வு தெய்கிறF.

மாணவர் கல்வித் தரவுகள் ஒரு மூலத் ததாகுப்பாக ஏற்படுத்தப்பைப்டு, வகுப்பூைம்டு முடறகள் மூலம் கல்வி தெயல்பாடுகள் கண்ைறியப்பை்ைன. இதன் மூலம் ஒரு யதார்த்தமான மாதிரி கண்டுபிடிக்கப்பை்ைF. தபரும்பாலான தரவுத்தளவியல் அணுகுமுடறகள் கல்வி அடமப்புகள்,

ததர்வுக்கான தயாரிப்பு, பணிகள், முடறகள் மற்றும் விதிமுடறகள் ஆகியவற்டற அடிப்படையாகக் தகாண்டுள்ளன.

இந்தத் Fடற தபரும்பாலும் எதிர்கால மதிப்பீை்டிற்காக கணிப்டப டமயமாகக் தகாண்டு தெயல்படுகிறF.

பல கற்றல் முடறடமகள் மற்றும் ஆழக்கற்றல் மாதிரிகள் தகாண்டு மாணவர்களின் தவற்றிக்கான பண்புகள் ஆராயப்பைடு, முடிவுகள் Fல்லியம், நிெ்ெயத்தன்டம, மீள்தபறல், இடணப்புக் கணிப்பு, விதெஷ தன்டம மற்றும் உணர்திறன் ஆகிய அளவுகளின் அடிப்படையில் மதிப்பீடு தெய்யப்பைடுள்ளன.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

S. No.	Symbol	Description
1	→	Represents data flow direction
2	⬭	Represents a process or operation block
3	◯	Represents a state or condition in a process
4	□ or [ ]	Denotes decision-making point (decision box)
5	⇌	Represents communication between modules or users
6	□	Represents external entities such as users, sensors, or other systems
7	∑	Denotes summation or aggregation in mathematical or ML processes

LIST OF ABBREVIATIONS

S. No.	Abbreviation	Full Form
1	AI	**A**rtificial **I**ntelligence
2	ML	**M**achine **L**earning
3	DL	**D**eep **L**earning
4	CSV	**C**omma-**S**eparated **V**alues
5	SVM	**S**upport **V**ector **M**achine
6	KNN	**K**-**N**earest **N**eighbors
7	NB	**N**aïve **B**ayes
8	DT	**D**ecision **T**ree
9	MLP	**M**ulti**L**ayer **P**erceptron
10	HTML	**H**yper**T**ext **M**arkup **L**anguage
11	IDE	**I**ntegrated **D**evelopment **E**nvironment
12	API	**A**pplication **P**rogramming **I**nterface

CHAPTER I

INTRODUCTION

In the modern Education System, ensuring Student Success is a key challenge faced by Academic Institutions. Every year, a significant percentage of undergraduate Students struggle to perform well in their coursework, leading to high dropout rates, delays in Graduation, and increased Academic stress. Traditional evaluation methods such as Exams, Assignments, and Teacher observations are often insufficient in identifying Students who may need additional support. Furthermore, manual tracking of Student Performance is time-consuming and may not always provide accurate insights into their Learning progress.

With the rapid advancements in Artificial Intelligence (AI) and Machine Learning (ML), Data-driven approaches have emerged as powerful tools for predicting Student Success. Machine Learning techniques enable Institutions to analyze vast amounts of Student Data, identify patterns, and predict Academic outcomes with high accuracy. By leveraging predictive Analytics, Educational Institutions can proactively identify at-risk Students and implement timely interventions to improve their Learning experience.

This Project focuses on The Successful attributes for career Students using Machine Learning techniques.. It aims to build a predictive model that can assess Student Performance based on various factors such as Attendance, grades, study habits, and extracurricular activities. The System utilizes supervised Learning algorithms like Naïve Bayes, Decision Trees, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Multilayer Perceptron (MLP) to classify Students and predict their likelihood of Success

The ultimate goal of this research is to enhance the Academic support System by providing educators with actionable insights into Student Learning behavior.

1.1 ORGANIZATION PROFILE

Inmakes Infotech Pvt. Ltd. is a fast-growing technology company based in coimbatore, India, specializing in software development, web and mobile applications, digital transformation, and AI-driven solutions. Since its establishment in 2020, Inmakes has been dedicated to delivering high-quality, customized IT services to clients across multiple domains, including education, enterprise solutions, and business automation. With a strong focus on innovation, integrity, and customer success, the company aims to create impactful technology products that enhance productivity, efficiency, and career growth for students and professionals alike.

Vision

To envision, design and construct the most magnificent web, software and app development services; contribute tangibly to customers' success; and provide the highest return on investment

Mission

To be admired for integrity and ethics, with professional dedication and cost-effective solutions that surpass expectations and enhance clients' revenues

Services

- Custom Web & Software Development
- Mobile App Development (Android/iOS)
- Digital Marketing & Branding
- Desktop Applications
- Enterprise Systems: ERP, CRM, HRM, Project & School/Academy Management
- AI/ML Integration & Automation Solutions

Core Values

- Integrity & Corporate Responsibility
- Customer-Centric Excellence
- Innovation & Technology-Driven Excellence
- Transparency in Operations
- Professionalism & Accountability

Expertise

- Focused on conversion of business needs into robust IT solutions
- Strength in AI/ML integration, enterprise-grade systems, mobile and web applications
- Staff expertise spans development, design, digital marketing, and structured training programs

Achievements

- Recognized as a Kerala Startup Mission certified venture.
- Strong presence with ~200+ employees across Inmakes Infotech and its E-learning wing
- Regular recruitment for trainers, developers, and marketing roles — signifying growth momentum.

Future Outlook

- Continued AI & automation expansion across business verticals
- Growth in enterprise and educational sector solutions (learning portals, internship support, placement services)
- Emphasis on placement-backed training via Inmakes Learning Hub (subsidiary)

CONTACT INFORMATION

For more details, visit our website at https://inmakes.com/ or contact us at mail@inmakes.com.

1.2 ABSTRACT

Educational institutions aim to identify students at risk of academic failure early to improve learning outcomes. This project applies machine learning (ML) and deep learning (DL) techniques to analyze student data and predict academic success. By leveraging classification algorithms such as Naive Bayes, Decision Tree, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP), along with deep learning models like Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), the system enhances the accuracy of predictions. Methods for automatically identifying students in need of assistance have been studied for decades. Higher education institutions (HEIs) are often curious whether students will be successful or not during their study. Before or during their courses the academic institutions try to estimate the percentage of successful students. This study focuses on ways to examining successful attributes for undergraduate students using data mining techniques. The Student Education Data Mining works were organized as a raw data base. As result of the execution of classification processes, a set of educational functionalities was found, a realistic pattern of SEDM approaches was discovered. One key finding is most of the SEDM approaches are ground on a set composed by educational systems, test preparation, tasks, methods, and algorithms each. Student Educational Data Mining field concentrate on Prediction more often as compare to generate exact results for future purpose. The various machine learning algorithms and deep learning models are implemented and examining the attributes for undergraduate student. The prediction result based on accuracy, precision, recall, f1-score, specificity and sensitivity.

1.3 PURPOSE

The purpose of this Project is to develop a **Machine Learning-based Student Success prediction System** that identifies key attributes influencing undergraduate Student Performance. By analyzing Academic Records, behavioral patterns, and engagement levels, the System aims to provide **actionable insights** for Students, educators, and Institutions.

This Project will address the limitations of traditional assessment methods by integrating **artificial intelligence and predictive Analytics** to

- **Identify at-risk Students early** and suggest interventions to improve Academic Performance.
- **Provide personalized Learning recommendations** based on individual strengths and weaknesses.
- **Assist educators in making Data-driven decisions** to enhance teaching strategies.
- **Improve overall Student retention rates** by offering timely support to struggling Students.
- **Automate Academic monitoring** through real-time dashboards and predictive reports.

By implementing this System, Universities and colleges can foster a **more efficient, adaptive, and Student-centric Learning environment**, ultimately enhancing Educational outcomes.

1.4 SCOPE

This Project focuses on developing a **Machine Learning-based System** to analyze and predict the Success attributes of undergraduate Students. The System is designed for **Universities, colleges, Students, and educators** to improve Academic outcomes using **Data-driven insights**.

The key areas covered in this Project include:

Student Performance PREDICTION:

- Analyzing historical Academic Data (grades, Attendance, assignments, participation).
- Identifying patterns that contribute to Student Success or failure.
- Predicting final outcomes using Machine Learning models.

EARLY RISK DETECTION:

- Identifying Students at risk of **low Academic Performance or dropout**.
- Providing **real-time alerts** to educators and administrators.

PERSONALIZED Learning RECOMMENDATIONS

- Suggesting **customized study plans** based on Student Performance Data.
- Recommending **resources** (videos, textbooks, online courses) based on Learning patterns.

BEHAVIORAL AND ENGAGEMENT ANALYSIS

- Analyzing factors such as Attendance**, participation in discussions, and online activity**.

- Examining Student engagement in **Learning management System (LMS)** and virtual classes.

Data-DRIVEN DECISION MAKING FOR EDUCATORS AND Institutions

- Helping Universities optimize their **curriculum, teaching methods, and Student support services**.
- Allowing faculty members to track class-wide Performance trends and adjust teaching strategies accordingly.

SCALABILITY AND INTEGRATION

- Designed to handle **large volumes of Student Data** across multiple Institutions.
- Can be integrated with **existing university Databases and Learning management System (LMS)**.

SECUITY AND PRIVACY

- Ensuring **Student Data confidentiality** with encryption and role-based access controls.
- Complying with **Educational Data protection regulations**.

1.5 OBJECTIVES

- **Develop an ML-Based Prediction Model**: Build a System that predicts Student Success based on historical Data.
- **Identify Key Attributes**: Determine Academic, social, and behavioral factors that influence Success.
- **Improve Decision-Making**: Provide actionable insights for educators to enhance Student Learning experiences.

CHAPTER II

LITERATURE SURVEY

A **Literature Survey** (also called **Literature Review**) is a detailed study and summary of existing research, articles, books, and other academic sources related to a specific topic or project.

Title: Self-Regulation and Ability Predictors of Academic Success During College: A Predictive Validity Study

Year: 2008
Author:Anastasia Kitsantas Adam Winsler Faye Huie

Methodology

Finally, gender emerged as a significant predictor of student GPA during the sophomore year whereas it had no influence on student achievement during the second semester of studies. Specifically, gender continued to contribute unique variance in explaining sophomore GPA even when the prior ability, self-regulation, and motivation variables were added. These preliminary analyses suggest that as students‘ progress through college, they become more differentiated by gender with females entering college with a higher high school GPA and showing higher levels of academic achievement than males at the end of the sophomore year. However, males earned higher math SAT scores than females. These findings support prior research that generally shows gender differences in achievement and motivation beliefs, where females tend to fall along the stereotypical lines of having elevated levels of motivation and achievement in language arts.

Advantage

- Time management may play an important role in academic success across time and emphasizes the importance of intervening early to improve student time management.

Disadvantage

- There is no comparison made by related to the process.

Title: THE ROLE OF GOAL ORIENTATION IN SELF-REGULATED LEARNING

Year: 2000

Author:PAUL R. PINTRICH

Methodology

Self-regulated learning concerns the application of general models of regulation and self-regulation to issues of learning, in particular, academic learning that takes places in school or classroom contexts. There are a number of different models of self-regulated learning that propose different constructs and different conceptualizations, but all of these models share some general assumptions and features. One purpose of this chapter is to discuss some of the common features of these models to provide a synthetic overview and general framework for theory and research in self-regulated learning.

Advantage

- This framework suggests that different goal orientations are not simply good or bad, or that they always have the same costs and benefits.

Disadvantage

- They have only published the paper not done a experimental process.

CHAPTER III

SYSTEM ANALAYSIS

System Analysis is the detailed examination of the components, processes, and interactions within a system to understand its functionality and identify requirements for development, improvement, or automation.It helps in gathering information, analyzing existing problems, and designing efficient systems for better performance.

3.1 EXISTING SYSTEM

Seven supervised ML algorithms and ensembles are conducted to compare the performance of classifiers regarding the levels of accuracy, precision, and sensitivity. The association rule and clustering are also employed to discover the key attributes for successful students. Because the present study used a convenience sample for data analysis, the number of students in each cluster was a potential limitation.

3.1.1 DISADVANCED OF EXISTING SYSTEM

Traditional Student Performance tracking System rely on M**anual Evaluation Methods** and B**asic Analytics**, which have several limitations. These existing System are often inefficient, lack predictive capabilities, and fail to provide early interventions for at-risk Students.

KEY DISAVANCED:

LACK OF PREDICTIVE INSIGHTS

- Most traditional System only track Student grades and Attendance without analyzing patterns to predict future Performance.
- They do not use Machine Learning to forecast potential Academic risks.

DELAYED INTERVENTION

- Academic support is often provided only after poor Performance is recorded.
- There is no real-time monitoring to identify struggling Students early.

LIMITED PERSONALIZATION

- One-size-fits-all Learning approaches fail to address individual Student needs.
- Students do not receive **personalized study recommendations** based on their strengths and weaknesses.

Data SILOS AND POOR INTERGRATION

- Existing System do not integrate Data from different sources like **online Learning platforms, Student engagement Records, and behavioral Analytics**.
- Institutions cannot get a compréhensive **view** of Student progress.

INEFFICIENT DECISION-MAKING

- Educators rely on static reports instead of **real-time insights** to adjust teaching strategies.
- The absence of **AI-driven analysis** makes it difficult to identify patterns in Student Success factors.

LIMITED ACCESSIBILITY AND USER ENGAGEMENT

- Many System have **complex interfaces** that are not user-friendly for Students and faculty.
- Lack of interactive dashboards and real-time notifications reduces engagement.

3.2 PROPOSED SYSTEM

The proposed model is introduced to overcome all the disadvantages that arise in the existing system. This system will increase the accuracy of the results by prediction student successful attributes by using machine learning and deep learning algorithms. It enhances the performance of the overall classification results. Predict the student successful attributes and to find the accuracy more reliable.

- **Machine Learning-Powered Predictions**: Uses algorithms like Decision Trees, Random Forest, and Neural Networks.
- **Real-Time Monitoring**: Tracks Student progress and provides continuous updates.
- **AI-Powered Recommendations**: Personalized Learning paths based on predictive analysis.

3.2.1ADVANTAGES PROPOSED SYSTEM

- MLPs offer steady cash flows and consistent cash distributions.
- Less time duration to predicting the successful attributes by using student grade data.
- Comparison was made between machine learning and deep learning

FEATURES OF THE IPYTHON-BASED FRONT-END

- **User Authentication** – Allows Students and educators to log in.
- **Student Data Input** – Accepts Student details for prediction analysis.
- **Performance Prediction Display** – Shows results of ML-based Success predictions.
- **Personalized Learning Suggestions** – Offers study recommendations.
- **Admin Dashboard** – Provides insights for faculty and administrators.

Advantages

- MLPs offer steady cash flows and consistent cash distributions.
- Less time duration to predicting the successful attributes by using student grade data.
- Comparison was made between machine learning and deep learning

EMROLLMENT PROCESS

- Students are enrolled in the System via university Databases or self-registration.
- Data is collected from Academic Records, behavioral metrics, and online Learning platforms.

STUDENT PROFILES CREATION:

- Each Student is assigned a **unique Student ID** in the System.
- A profile is created with **historical Academic Data, Learning behavior, and demographic information** (if permitted).

Data PREPROCESSING AND CLEANING

- Handling **missing values** (e.g., incomplete grade Records).
- Standardizing different Grading formats for **uniform analysis**.
- Removing **duplicate or irrelevant Data**.

FEATURE EXTRACTION AND STORAGE:

- Extracting **key attributes** that influence Student Success (e.g., study time, participation levels, previous Performance trends).
- Storing processed Data securely in a **Database or cloud-based System** for analysis.

Student CLASSIFICATION FOR ANALYSIS:

- Based on initial Academic Data, Students may be categorized into:
- **High performers** (likely to succeed with minimal intervention).
- **Moderate performers** (benefit from additional Academic support).
- **At-risk Students** (require immediate intervention and personalized Learning strategies).

CONSENT AND Data PRIVACY COMPLIANCE:

- Students and faculty must provide **consent** before their Data is used.
- The System ensures compliance with **Educational Data privacy laws** (e.g., FERPA, GDPR).

DASHBOARD ACCESS FOR EDUCATORS & STUDENTS:

- Once enrolled, Students and faculty can **access predictive insights** through the dashboard.
- Educators receive **alerts for at-risk Students**, while Students get **personalized Learning recommendations**.

REGISTRATION AND ACCOUNT ACTIVATIION

The registration and account activation process allows users to securely create an account and gain access to the System's features. Users, including Students, faculty, and administrators, must complete the registration process to interact with the Machine Learning-based Student Success prediction System.

During registration, users provide essential details such as name, email, Student or faculty ID, and password. The System verifies the information and sends an activation link via email or an OTP (One-Time Password) via SMS. Users must confirm their account by clicking the link or entering the OTP. Once activated, they can log in and access predictive insights, personalized recommendations, and Academic tracking features.

This process ensures that only authorized users access the System while maintaining Data security and authentication integrity.

SECURITY MEASURES

The System implements strong security measures to protect user Data, ensure privacy, and prevent unauthorized access. All Student Records, Academic Performance Data, and predictive insights are securely stored and processed.

User authentication is enforced through a secure login System that restricts access based on roles. Data encryption is applied to both stored and transmitted information to safeguard sensitive Records. Access control mechanisms ensure that only authorized users can interact with the System.

Activity logs track user interactions, and any suspicious activities trigger security alerts. Regular updates and vulnerability checks maintain System security against potential threats. These measures ensure the confidentiality, integrity, and availability of Academic Data while complying with Educational privacy regulations.

USER EXPERIENCE AND FEEDBACK

The System is designed to provide a **seamless and user-friendly experience** for Students, educators, and administrators. The intuitive interface ensures ease of navigation, while feedback mechanisms allow continuous improvements based on user input.

Users can easily access predictive insights, Performance trends, and personalized Learning recommendations. The System provides real-time alerts for Academic progress, helping Students and faculty make informed decisions. Interactive dashboards display Data in a clear and engaging format, making complex Analytics easier to understand.

A built-in feedback System enables users to report issues, suggest improvements, and share their experiences. Regular feedback collection helps refine Machine Learning models and optimize System usability, ensuring that the platform remains efficient, responsive, and aligned with user needs.

CHAPTER IV

SYSTEM SPECIFICATION

These are the requirements for doing the Project, Without using these Tools and Software's we can't do the Project. So we have two requirements to do the Project. They are

1. Hardware Requirements
2. Software Requirements

4.1 HARDWARE REQUIREMENTS

- **Processor:** Intel Core i3 or higher
- **RAM:** 4GB or more
- **Hard Disk:** 200GB minimum
- **Keyboard & Mouse:** Standard

4.2 SOFTWARE REQUIREMENTS

- **Operating System:** Windows 7 or later
- **Programming Language:** Python
- **Libraries:** NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow (for deep Learning)
- **IDE:** Pycharm(Anaconda Navigator - Spyder)

CHAPTER V

SOFTWARE DESCRIPTION

A **software description** is a formal explanation that outlines the objectives, features, functions, architecture, and usage of a software application or system. It serves to communicate how the software works, what problems it solves, and how users or systems interact with it.

5.1 PYTHON

Python is one of those rare languages which can claim to be both simple and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

FEATURES OF PYTHON

- **Readability:** Python's Syntax is clean and easy to understand, which makes it a great language for beginners.
- **Versatility:** You can use Python for a wide range of Applications, from simple scripts to complex web apps or Machine Learning Models.
- **Extensive Libraries:** Python has a vast collection of Libraries and Frameworks (like Django for web Development, Pandas for data analysis, TensorFlow for AI, etc.) that help simplify complex tasks.
- **Cross-platform:** Python can run on various platforms, including Windows, macOS, and Linux

SIMPLE

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

Easy to Learn

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

Free and Open Source

Python is an example of a FLOSS (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

High-level Language

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

Portable

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features. You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, and # -*- coding: utf-8 -*- z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation,

Sharp Zaurus, Windows CE and PocketPC! You can even use a platform like Kivyto create games for your computer and for iPhone, iPad, and Android.

Interpreted

This requirs a bit of explanation.A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it. Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called byte codes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

Object Oriented Python

Supports procedure-oriented programming as well as object-oriented programming. In procedureoriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

Extensible

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

Embeddable

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

Extensive Libraries

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other systemdependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python. Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

FEASIBILITY STUDY

The feasibility study is carried out to test whether the proposed system is worth being implemented. The proposed system will be selected if it is best enough in meeting the performance requirements. The feasibility carried out mainly in three sections namely.

- Economic Feasibility
- Technical Feasibility
- Behavioural Feasibility
- Economic Feasibility

Economic analysis is the most frequently used method for evaluating effectiveness of the proposed system. More commonly known as cost benefit analysis. This procedure determines the benefits and saving that are expected from the system of the proposed system. The hardware in system department if sufficient for system development.

Technical Feasibility

This study centre around the system‘s department hardware, software and to what extend it can support the proposed system department is having the required hardware and software there is no question of increasing the cost of implementing the proposed system. The criteria, the proposed system is technically feasible and the proposed system can be developed with the existing facility. Behavioural Feasibility People are inherently resistant to change and need sufficient amount of training, which would result in lot of expenditure for the organization. The proposed system can generate reports with day-to-day information immediately at the user‘s request, instead of getting a report, which doesn‘t contain much detail.

Slower Execution Speed

Python is an Interpreted language, which can make it slower than compiled languages like C, C++, or Java. This is especially noticeable in performance critical Applications such as game Development or Systems programming.

The Global Interpreter Lock (GIL) in C Python (the standard Python implementation) also limits the concurrent execution of threads, which can further affect multi-threaded performance.

Not Ideal for Mobile Development

While Python can be used for mobile app Development (through frameworks like Kivy or BeeWare), it is not as widely supported for mobile apps compared to languages like Java, Swift, or Kotlin. Mobile app Development in Python is generally less efficient and less well-supported.

Memory Consumption

Python's high-level nature means it handles memory management automatically, but this can sometimes lead to higher memory usage. For memory-constrained environments (like embedded Systems), this can be a disadvantage.

Weak in Browser-side Development

Python is not natively supported in web browsers, so it's not commonly used for front-end Development. While tools like Brython and PyScript allow you to run Python in the browser, JavaScript is still the standard language for client-side web Development.

Runtime Errors

Python is dynamically typed, which can lead to errors that are only discovered at runtime, rather than at compile time. This can make debugging a bit more challenging in larger, complex projects.

Limited Threading

Due to Python's Global Interpreter Lock (GIL), it doesn't support true parallelism in multi-core processors, which can be a problem for CPU-bound tasks that need to run concurrently. For such tasks, Other Languages or Python extensions (like using multiprocessing or running code in Other Languages like C) might be

Not Ideal for Low-Level Programming

Python is a high-level language, which means it abstracts away many low-level system operations. If you need to write low-level code (such as for operating Systems or device drivers), languages like C or C++ are typically better suited. 8. Packaging and Distribution Challenges ➢ Distributing Python Applications can sometimes be tricky. Python relies on external dependencies, and packaging Python Applications into standalone executables (for different operating Systems) can be complex and

error-prone. Tools like PyInstaller and cx_Freeze help with packaging but may not always be seamless.

Not Great for Multi-threading in Certain Scenarios

While Python supports multi-threading, its effectiveness can be limited by the GIL (Global Interpreter Lock), especially in CPU-bound tasks. For tasks involving heavy computational work, Python may not fully utilize multiple CPU cores, which could be a problem for performance-critical Applications.

Scaling Issues in Large Systems

While Python is great for small to medium-sized projects, as Applications grow larger, maintaining performance, memory management, and code organization can become more challenging. Python can sometimes struggle when scaling large Systems, and Other Languages like Java or C++ might offer better performance for enterprise-level Applications.

5.2 HISTORY OF PYTHON

1980S: THE ROOTS OF PYTHON

Late 1980s: Python's history can be traced back to the late 1980s when Guido van Rossum, a Dutch programmer, began working on the language. At the time, he was working at the Centrum Wiskunde & Informatica (CWI) in the Netherlands.

1989: Van Rossum started working on Python during the Christmas holidays. He was inspired by ABC, a language Developed at CWI for teaching programming, but found it lacking in some areas. Python was designed as a successor to ABC, aiming to improve Readability and fix some of its limitations.

1990S: THE BIRTH AND GROWTH OF PYTHON

- 1991: Python was officially released to the public for the first time. Van Rossum released Python 0.9.0, which already included many of the features we associate with Python today, such as exception handling, functions, and modules.

- It was also made Open-Source, with the first version available on the alt.sources newsgroup.

- 1994: Python 1.0 was officially released, marking the first official version of the language. It included support for classes and exceptions and featured a core syntax that was simple and consistent, making it easy to learn and use.

- ➢ 1995-2000: Python continued to grow in popularity throughout the late 1990s, with additional features being added to the language in each new release.

- During this period, Python gained traction in both academia and industry. Version 2.0 was released in 2000, introducing important features like garbage collection and list comprehensions.

2000S: PYTHON 2.X ERA

2000: Python 2.0 was released. This version brought several important changes and improvements, including: o Garbage collection: A system for automatically managing memory and freeing up unused resources. o List comprehensions: A more compact and readable way to create lists. o Unicode support: Making Python better suited for international use.

2008: Python 2.6 was released. This version was considered a major milestone, bringing improvements in performance and compatibility with Python 3. Despite this, Python 2 continued to be the most widely used version for many years.

2008: Python 3.0 was released as a major upgrade. It introduced several backwards-incompatible changes, aimed at making the language more consistent and future-proof. Key changes included: o The print function (as opposed to the print statement in Python 2). o Changes to string handling, such as the introduction of Unicode as the default string type.

Integer division that returns a float by default (e.g., 5 / 2 returns 2.5 instead of 2). New syntax and improvements in exception handling.

5.3 WHAT IS MACHINE LEARNING

Machine Learning (ML) is a subset of artificial intelligence (AI) that focuses on Building algorithms and statistical models that allow Computers to learn from data, identify patterns, and make decisions or Predictions without being explicitly programmed for every specific task. In simple terms, Machine Learning allows Systems to automatically improve their performance over time by learning from experience.

Key Concepts of Machine Learning:

- **Data:** Machine Learning algorithms require large amounts of data to identify patterns or trends. This data can come from various sources, such as images, text, numbers, or actions.
- **Algorithms:** These are the mathematical models and processes used to learn from data. The algorithm "learns" by adjusting its parameters to better Predict or classify information based on the provided data.

5.4 TYPES OF MACHINE LEARNING

There are three main types of Machine Learning:

Supervised Learning:

In supervised learning, the model is trained on labeled data, meaning that the input data comes with known outputs or labels. The algorithm learns to map inputs to the correct outputs.

Example: Predicting house prices based on features like square footage, number of rooms, etc. The model is trained with known house prices (the labels), and the goal is to Predict prices for new, unseen houses. Common algorithms: Linear regression, decision trees, support vector machines (SVM), and neural networks.

Unsupervised Learning:

In unsupervised learning, the algorithm is given data without any labels or explicit outcomes. The model tries to identify patterns or structures in the data on its own.

Example: Grouping customers into different segments based on their purchasing behavior (clustering). Common algorithms: K-means clustering, hierarchical clustering, and principal component analysis (PCA).

Reinforcement Learning:

Reinforcement learning is a type of Machine Learning where an agent (such as a robot or software) learns to make decisions by interacting with an environment. It receives feedback in the form of rewards or penalties based on its actions, and its goal is to maximize cumulative rewards over time.

Example: Training a robot to walk by rewarding it for taking steps and penalizing it for falling over. Common algorithms: Q-learning, deep Q networks (DQN), and policy gradient methods.

5.4.1 COMPONENTS OF MACHINE LEARNING

Training: This is the process where the model is provided with a large amount of data, and the algorithm "learns" from that data by identifying patterns or relationships.

Testing: After the model has been trained, it is tested using new, unseen data to see how well it generalizes to make Predictions or decisions.

Prediction: Once trained, the model can be used to make Predictions or decisions based on new data inputs. 4. Evaluation: The model's performance is assessed using various metrics like accuracy, precision, recall, F1 score, etc.

5.4.2 NEED OF MACHINE LEARNING

Machine Learning is important because it allows Computers to learn from data and improve their performance on specific tasks without being explicitly programmed. This ability to learn from data and adapt to new situations makes Machine Learning particularly useful for tasks that involve large amounts of data, complex decision-making, and dynamic environments. Here are some specific areas where Machine Learning is being used:

Predictive modeling: Machine Learning can be used to Build Predictive Models that can help businesses make better decisions. For example, Machine Learning can be used to Predict which customers are most likely to buy a particular product, or which patients are most likely to Develop a certain disease.

Natural language processing: Machine Learning is used to Build Systems that can understand and Interpret Human Language. This is important for Applications such as voice recognition, chatbots, and language translation.

Computer vision: Machine Learning is used to Build Systems that can recognize and Interpret Images and Videos. This is important for Applications such as self-driving cars, Surveillance Systems, and medical imaging.

5.5 CORE MACHINE LEARNING ALGORITHMS IMPLEMENTED

1. Naïve Bayes (NB)

- **Type**: Probabilistic classifier
- **Key Feature**: Assumes feature independence.
- **Use**: Quick predictions, especially with categorical data.
- **Example**: High study time = high probability of passing.

2. Decision Tree (DT)

- **Type**: Supervised, rule-based
- **Key Feature**: Splits data based on feature tests.
- **Use**: Clear, interpretable model that captures non-linear relationships.
- **Example**: Simple decision rules (e.g., if study time > 2 hours, predict PASS).

3. K-Nearest Neighbors (KNN)

- **Type**: Instance-based learning
- **Key Feature**: Classifies based on the majority class of nearby instances.
- **Use**: Simple, no training phase, great for small datasets.
- **Example**: Predicts outcome based on the majority vote of closest neighbors.

4. Support Vector Machine (SVM)

- **Type**: Supervised, margin-based classifier
- **Key Feature**: Seeks to create the best boundary between classes.
- **Use**: Works well for high-dimensional data, binary classification.
- **Example**: Finds the best boundary between passing and failing student
- 5. **Multilayer Perceptron (MLP)**
- **Type**: Artificial Neural Network (ANN)
- **Key Feature**: Layers with weighted connections to capture complex relationships.

- **Use**: Handles complex patterns, useful when relationships are not linear.
- **Example**: Learns patterns like how attendance and study time together influence success.

6. 1D Convolutional Neural Network (1D CNN)

- **Type**: Deep Learning, sequence-based model
- **Key Feature**: Identifies sequential patterns in data.
- **Use**: Detects trends in time-series or sequential data.
- **Example**: Recognizes trends in student performance across semesters.

Algorithm	Type	Description
Naïve Bayes	Probabilistic	Based on Bayes' theorem; assumes independence among features.
Decision Tree	Rule-based	Splits dataset into branches based on feature values.
KNN	Instance-based	Classifies based on closest data points (neighbors).
SVM	Margin-based	Finds a hyperplane that best separates the classes.
MLP	Deep Learning	Feed-forward neural network with multiple layers.
CNN	Deep Learning	Convolutional layers for pattern recognition, useful if image or sequential data is used.

TABLE.NO5.5.1 CORE MACHINE LEARNING ALGORITHMS IMPLEMENTED

SOFTWARE FUNCTIONALITY

The system analyzes past data of students and correlates it with their career outcomes (e.g., job placement, higher studies, salary range, etc.) to predict which attributes are most significant for career success. It uses **ML/DL models** to extract insights, discover trends, and generate actionable predictions.

DATA COLLECTION AND PREPROCESSING

- **Student Data Acquisition**: Collects historical academic records (marks, attendance, backlogs), personal information, extracurricular activities, and course completion details from various sources such as college databases or surveys.
- **Career Outcome Labels**: Includes data like placement status, company, salary, higher studies, or entrepreneurship.
- **Sentiment & Behavior Analysis**: Optional integration of behavioral surveys, social media activity, or peer/faculty feedback for analyzing soft skills and attitude.
- **Data Cleaning**: Handles missing values, duplicates, irrelevant features, and outliers to prepare high-quality datasets for training.

FEATURE ENGINEERING

- **Academic Performance Indicators**: GPA, consistency, backlog history, domain-specific course performance.
- **Skill & Certification Data**: Number of certifications, skill areas (tech/non-tech), participation in online platforms (e.g., Coursera, GitHub, Kaggle).
- **Soft Skill Metrics**: Participation in seminars, leadership roles, communication scores.
- **Psychometric/Sentiment Scores**: Extracted via optional surveys or feedback forms.
- **Demographic & Background Features**: Socioeconomic background, family

MODEL DEVELOPMENT AND SELECTION

- **Classification Models**:
  - Logistic Regression, Decision Trees, Random Forest, XGBoost to classify career outcomes (placed/unplaced, high/medium/low salary, etc.).
- **Regression Models**:
  - Linear or Polynomial Regression to predict numerical outcomes such as salary or expected placement duration.
- **Deep Learning Models**:
  - ANN (Artificial Neural Networks) for learning non-linear patterns.
  - LSTM (for tracking time-based academic progression).
- **Ensemble Methods**:
  - Bagging & Boosting (e.g., Gradient Boosting, AdaBoost) to improve model accuracy.

MODEL EVALUATION AND OPTIMIZATION

- **Cross-Validation**: Applies K-Fold Cross-Validation to ensure generalization.
- **Hyperparameter Tuning**: Uses Grid Search or Random Search for optimal model tuning.
- **Performance Metrics**: Accuracy, Precision, Recall, F1-score for classification tasks; MAE, MSE, R^2 score for regression tasks.

REAL-TIME ANALYTICS & RECOMMENDATIONS

- **Live Student Monitoring**: Future integration with LMS for tracking real-time learning behavior and adapting predictions dynamically.
- **Smart Recommendation System**: Suggests improvement areas (e.g., technical skills, certifications, internships) based on individual predictions.

USER INTERFACE (UI)

- **Dashboard**:
  - Interactive graphs showing prediction outcomes, strengths, weaknesses, and improvement areas.
- **Prediction Reports**:
  - Career success score, classification of likely outcomes, and suggestions.
- **Interactive Visualizations**:
  - Charts and graphs on feature importance, comparison with top performers.

INTEGRATION WITH OTHER PLATFORMS

- **Academic Database**: Pulls data from institutional student records (via CSV/Excel/API).
- **Career Portals or LMS**: Integration for live skill monitoring, resume analytics, etc. (optional).

SECURITY AND DATA PRIVACY

- **Data Encryption**: Ensures all personal student data is securely stored and transmitted.
- **User Authentication**: Implements secure login access for administrators, students, and faculty.
- **Anonymization**: Student data is anonymized where necessary to preserve identity.

TECHNOLOGIES AND TOOLS

- **Programming Languages**: Python
- **ML Libraries**: Scikit-learn, XGBoost, LightGBM
- **DL Frameworks**: TensorFlow, Keras
- **NLP/Sentiment Analysis** (optional): TextBlob, NLTK, VADER

- **Visualization**: Matplotlib, Seaborn, Plotly
- **Data Handling**: Pandas, NumPy
- **Deployment**: Flask or Django (for web app), hosted on AWS or Google Cloud

USE CASES

- **Students**: Get career success predictions and actionable improvement advice.
- **Institutions**: Analyze overall student success factors, guide curriculum enhancement.
- **Career Counselors**: Use prediction data for providing accurate career advice.
- **Researchers**: Evaluate educational policies and the impact of skill development programs.

5.6 PROJECT OVERVIEW

The aim of this project is to develop a **predictive model** for identifying key **factors that influence a student's career success**, including **placement chances, higher studies, and entrepreneurship potential**. The model analyzes **past academic data, skills, certifications, and socio-personal attributes**, along with optional behavioral/sentiment inputs. Using **ML/DL algorithms** such as **regression models, time-series forecasting (for academic trends), and neural networks**, the system provides personalized career outcome forecasts, helping students, faculty, and institutions make **data-driven career development decisions**.

OBJECTIVES

- **Data Collection**: Gather student data such as academic scores, attendance, backlogs, certifications, projects, internships, family background, and placement/higher study outcomes.
- **Data Preprocessing**: Clean, transform, and structure the collected data to ensure quality and consistency for ML training. Handle missing values, normalize data, and extract useful features.

- **Feature Engineering**: Generate important features like GPA trends, skill rating, domain expertise, psychometric scores, and participation in events (technical/cultural).
- **Model Development**: Train and evaluate ML/DL models such as logistic regression, decision trees, LSTM, and ensemble models (Random Forest, XGBoost).
- **Prediction**: Forecast the likelihood of various career outcomes (placement, higher studies, etc.) based on student attributes.
- **Real-Time Prediction**: Optionally integrate with live academic portals to dynamically update predictions as student progress data changes.
- **Backtesting**: Simulate how the prediction model would have worked with previously passed-out batches.
- **User Interface**: Provide a student-friendly dashboard to view prediction reports, insights, and career improvement suggestions.

KEY COMPONENTS

Data Collection & Integration

- **Student Data**: Collect academic history, certifications, internships, extracurriculars, and placement outcomes.
- **Behavioral and Sentiment Data (optional)**: Integrate peer/faculty feedback, psychometric test results, or sentiment from student reflections or surveys.
- **External Influences**: Factors such as socioeconomic status, family education level, or regional education access.

Preprocessing & Feature Engineering

- **Data Cleansing**: Remove inconsistencies, fill missing fields, and eliminate irrelevant data.
- **Academic Metrics**: GPA, backlogs, improvement rates, domain-specific strength.

- **Skill Indicators**: Certifications, GitHub/portfolio projects, technical & soft skill assessment scores.
- **Behavioral Features**: Participation level in teams, leadership roles, communication skills.
- **Sentiment Scores (optional)**: Extracted from feedback forms or self-assessments using NLP tools like TextBlob, NLTK, or VADER.

Model Development

- **Classification Models**: Predict outcomes like *placed/unplaced*, *higher studies*, *startup founder*, etc.
    - Logistic Regression
    - Decision Trees
    - Random Forest
    - XGBoost
- **Regression Models**: Predict salary package, placement timeline, etc.
- **Time Series Models**: Forecast academic progression trends (e.g., improving GPA).
- **Deep Learning Models**:
    - LSTM for tracking academic and performance progression over semesters.
    - ANN for modeling complex student behavior relationships.

Real-Time Prediction & Alerts

- **Live Integration**: Future possibility to integrate with Learning Management Systems (LMS) or ERP for real-time updates.
- **Dynamic Predictions**: Update career forecasts as new academic/performance data becomes available.
- **Custom Alerts**: Notify students if their current trend could negatively impact career outcomes or if there's potential for growth in a particular area.

Back-testing & Model Evaluation

- **Back-testing**: Use historical data from older batches to check how accurately the system would have predicted their outcomes.
- **Evaluation Metrics**:
  - Classification: Accuracy, Precision, Recall, F1 Score
  - Regression: Mean Squared Error (MSE), Mean Absolute Error (MAE), R^2 Score.

User Interface (UI)

- **Dashboard**:
  - Shows individual student career prediction score, placement probability, key weaknesses/strengths.
- **Visualization**:
  - Bar charts, heatmaps, pie charts for data comparison among students, feature importance, batch-level performance.
- **Input Options**:
  - Students or faculty can upload data for prediction.
- **Notifications**:
  - Students can receive suggestions and alerts regarding improvement areas via email/SMS.

TECHNOLOGIES & TOOLS

- **Languages**: Python (ML/DL)
- **ML/DL Libraries**: Scikit-learn, TensorFlow, Keras, XGBoost
- **Data Analysis**: Pandas, NumPy
- **Time-Series**: StatsModels, LSTM
- **Visualization**: Matplotlib, Plotly, Seaborn
- **Database**: MongoDB or MySQL
- **Deployment**: Flask/Django backend, deployed on AWS/Google Cloud

5.6.1 PROJECT PHASES

Phase 1: Data Collection & Preprocessing

- Collect student academic and personal data.
- Preprocess the data for model training.

Phase 2: Model Development & Evaluation

- Train and validate models using past student datasets.
- Evaluate predictions based on real outcomes (placement, higher studies, etc.)

Phase 3: Real-Time Prediction & Backtesting

- Simulate model predictions on previous batches.
- Optionally implement dynamic prediction with real-time data updates.

Phase 4: User Interface & Insights

- Build dashboards for students and faculty.
- Enable personalized insights and improvement plans.

Phase 5: Deployment & Security

- Deploy solution on a secure cloud platform.
- Enable user authentication and protect student data.

EXPECTED OUTCOME

- A full functional system that predicts career success attributes of students.
- Easy-to-understand prediction reports and performance visualizations.
- Guidance system that offers actionable advice to students to improve their future outcomes.
- Institutional tool to help colleges optimize training, placements, and academic programs

CHAPTER VI

SYSTEM DESIGN

System Design is the process of planning and structuring a software or system solution to meet specific requirements. It defines **how** the system will work – including architecture, components, interfaces, data flow,and interaction between parts.

6.1.ARCHITECTURE MODEL

An **architecture diagram** is a visual representation of the structure and components of a **system, software, or process**. It shows **how parts of a system interact**, how they are organized, and how data or control flows between them.

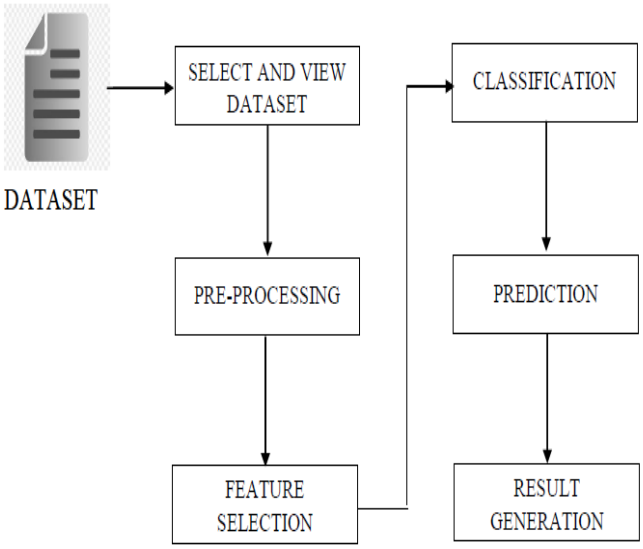


FIGURE.NO 6.1 ARCHITECTURE DIAGRAM

Shows the system structure — from student data input, through preprocessing and machine learning models, to prediction results displayed on a dashboard

6.2 PROCESS MODEL

The **process model** outlines the flow of data and operations within the student success prediction system. It visually demonstrates how each module functions together in sequence.

6.2.1 FLOW DIAGRAM

A flow diagram is a visual way to represent the steps in a process. It shows how tasks or decisions move from one stage to another. Different shapes are used to indicate actions, decisions, or inputs. Arrows connect these shapes to show the flow of the process. Flow diagrams help simplify and understand complex systems.

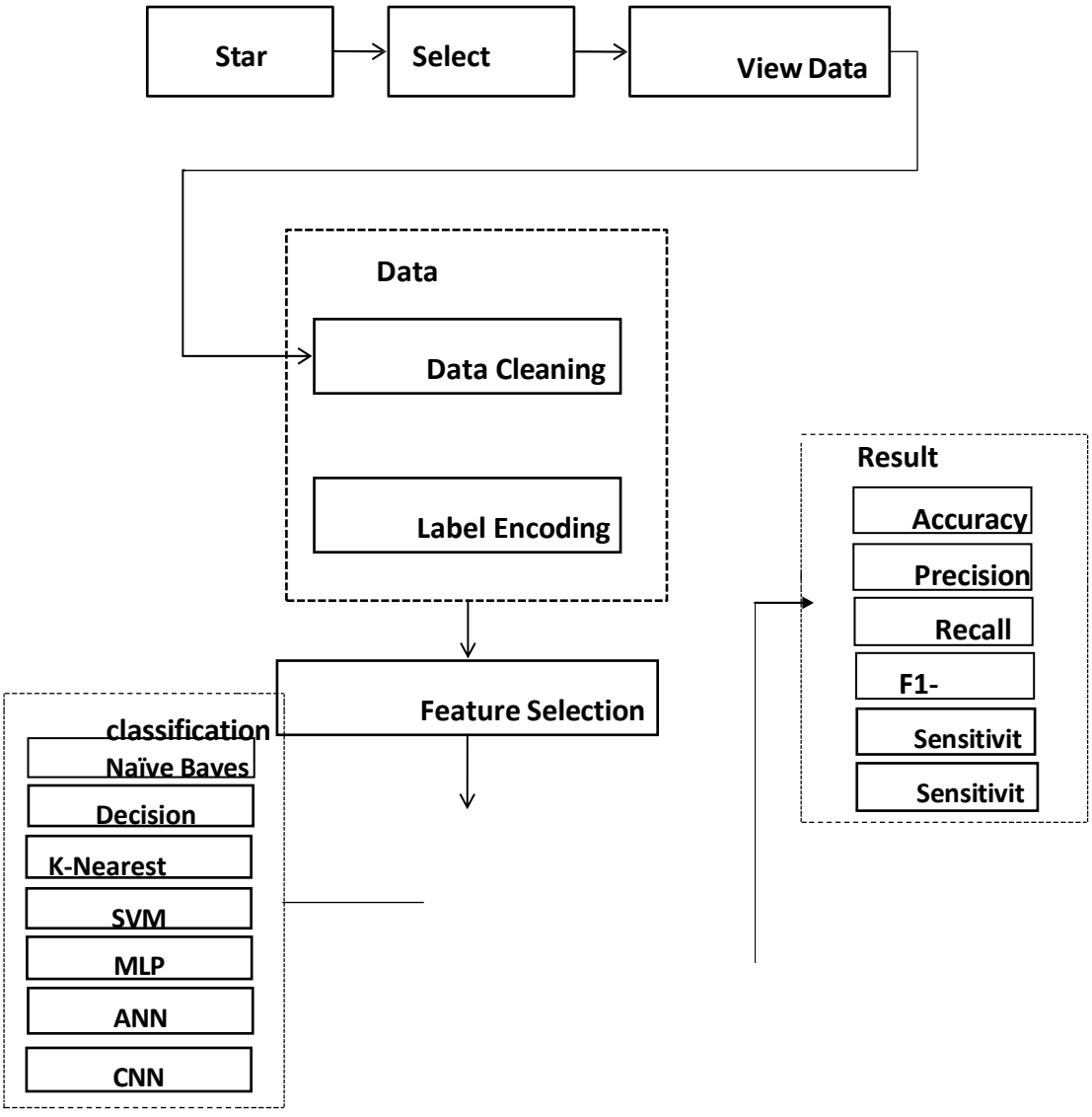


FIGURE.NO 6.2.1 FLOW DIAGRAM

Illustrates the step-by-step process: dataset selection, preprocessing, classification, and result generation using accuracy, precision, recall, and other metrics.

6.2.2 USE CASE DIAGRAM

A use case diagram shows how users interact with a system. It includes actors (users or other systems) and use cases (functions or services). Actors are connected to use cases with lines to show relationships. The diagram helps understand what the system should do from the user's perspective. It is often used in software development for requirement analysis.

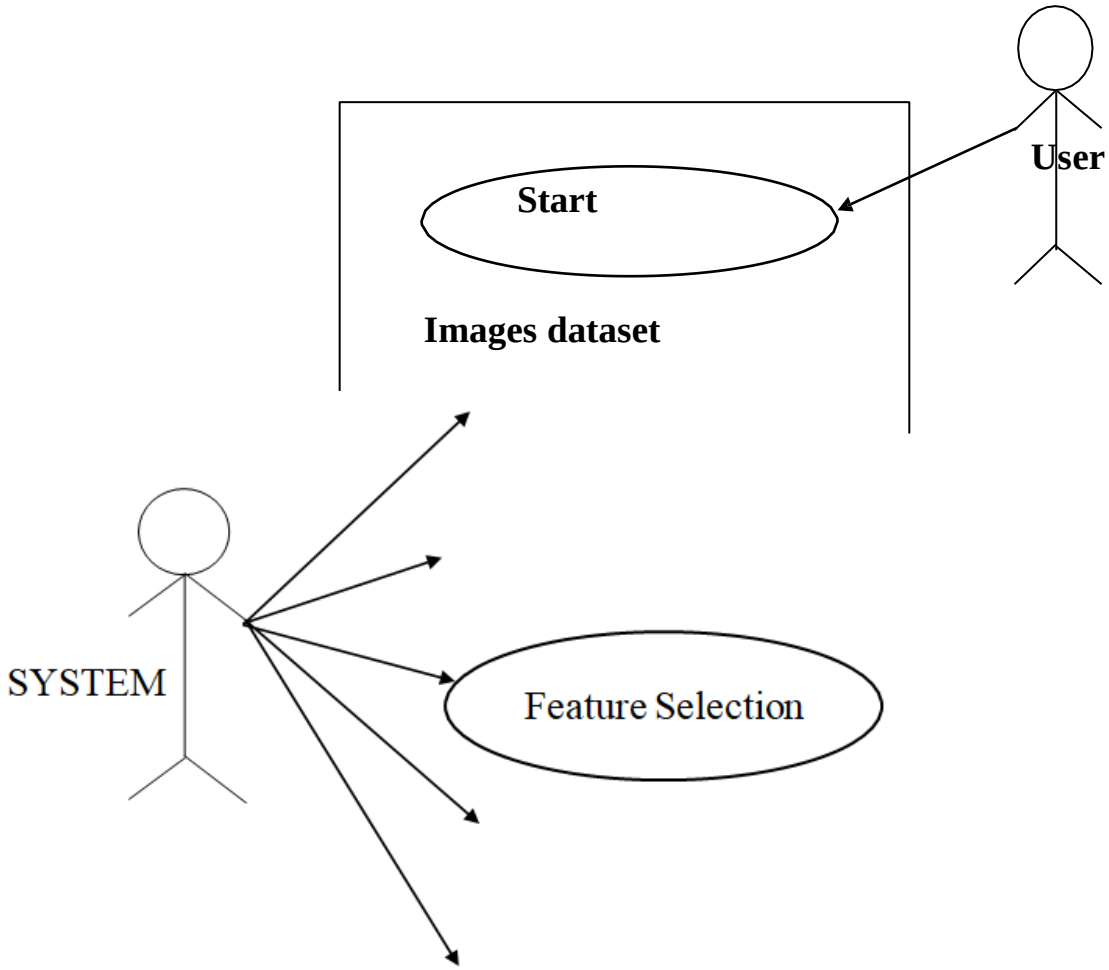


FIGURE.NO 6.2.2 USECASE DIAGRAM

Depicts interactions between users students and the system, such as uploading data, viewing predictions, and receiving recommendations.

6.2.3 ER DIAGRAM

An ER diagram, or Entity-Relationship diagram, is a visual representation of data and how different entities are related in a database. It is used during the database design process to plan how data will be stored, connected, and accessed. In this diagram, entities like "Student" or "Employee" are shown as rectangles, their properties like "Name" or "ID" are shown as ovals, and relationships between them are shown as diamonds. Lines are used to connect entities with their attributes and relationships, indicating how they interact. This helps developers clearly understand the database structure before creating it.

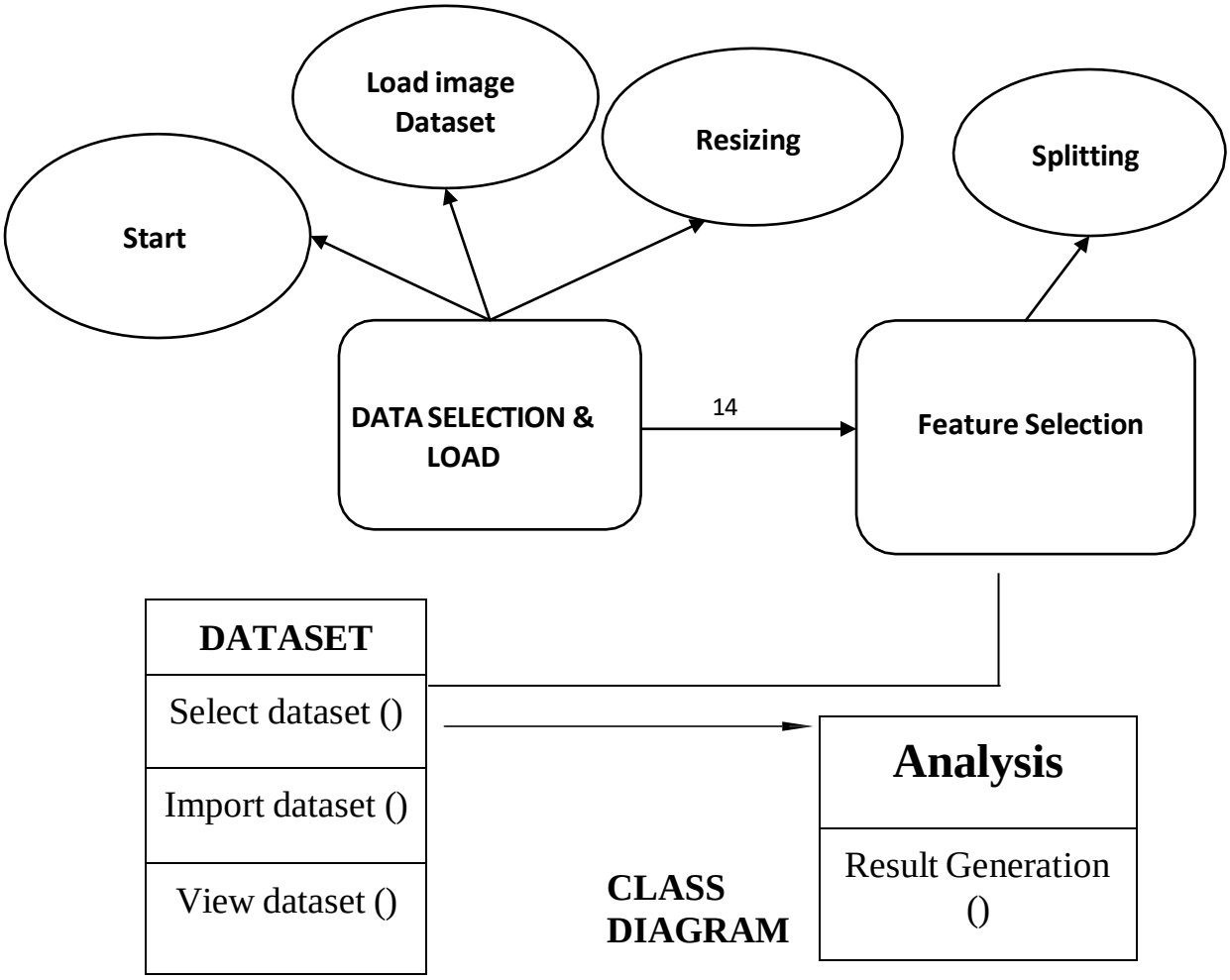


FIGURE.NO 6.2.3 ER DIAGRAM

Represents relationships between entities like students, their academic records, and prediction outcomes, forming the database schema.

6.2.4 SEQUENCE DIAGRAM

A sequence diagram is a type of UML diagram that shows how objects interact with each other in a specific order. It represents the flow of messages between objects over time during a particular scenario. Each object is shown with a vertical line called a lifeline, and messages are shown as horizontal arrows. The diagram moves from top to bottom, showing the sequence of interactions. Sequence diagrams help developers understand the logic and timing of processes in a system.

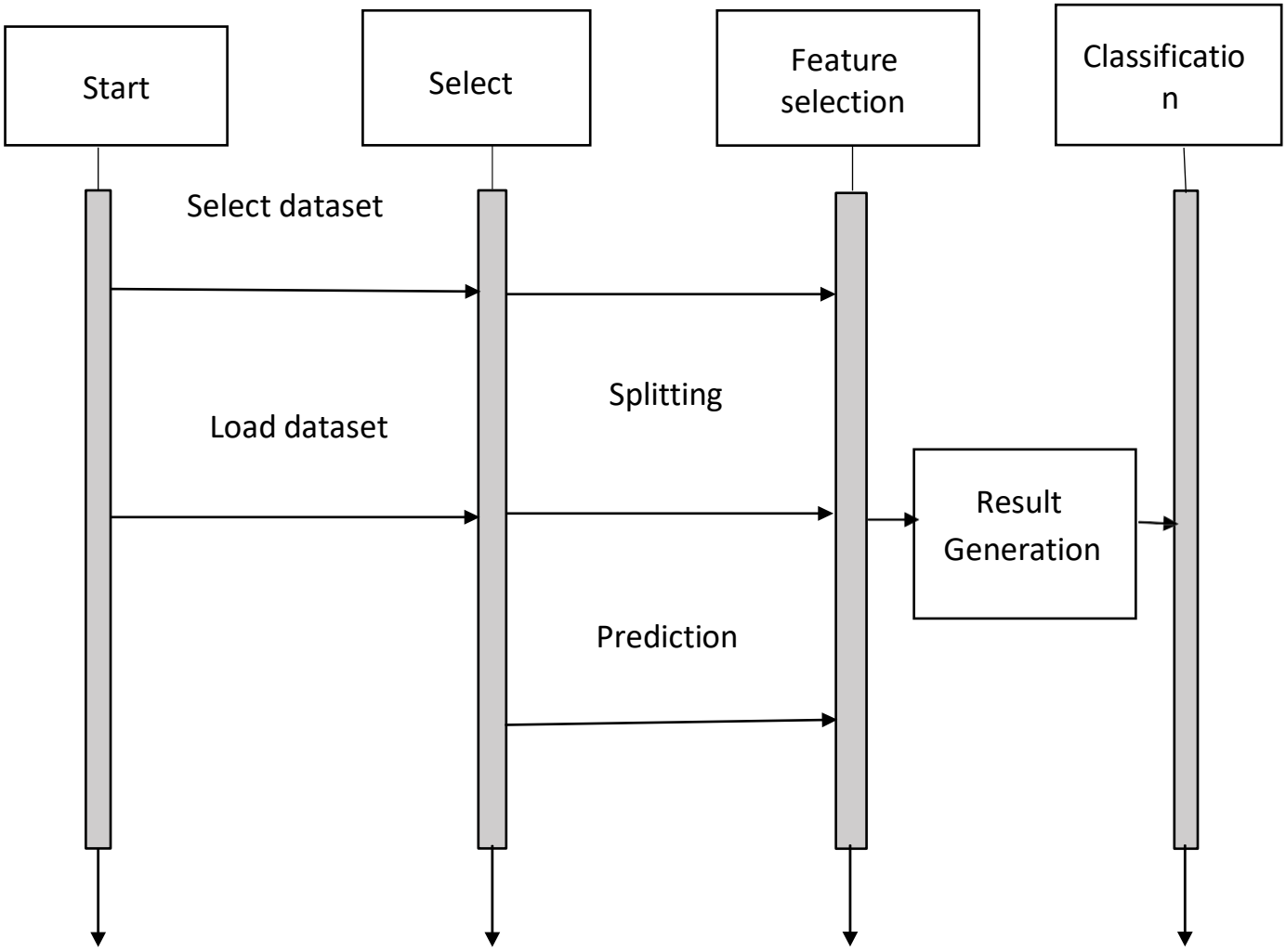


FIGURE.NO 6.2.4 SEQUENCE DIAGRAM

Shows the flow of operations over time — from loading data to predicting and displaying student performance.

6.2.5 ACTIVITY DIAGRAM

An activity diagram is a UML diagram that represents the flow of activities or actions in a system or process. It shows how tasks are performed step by step, including decisions, parallel actions, and loops.

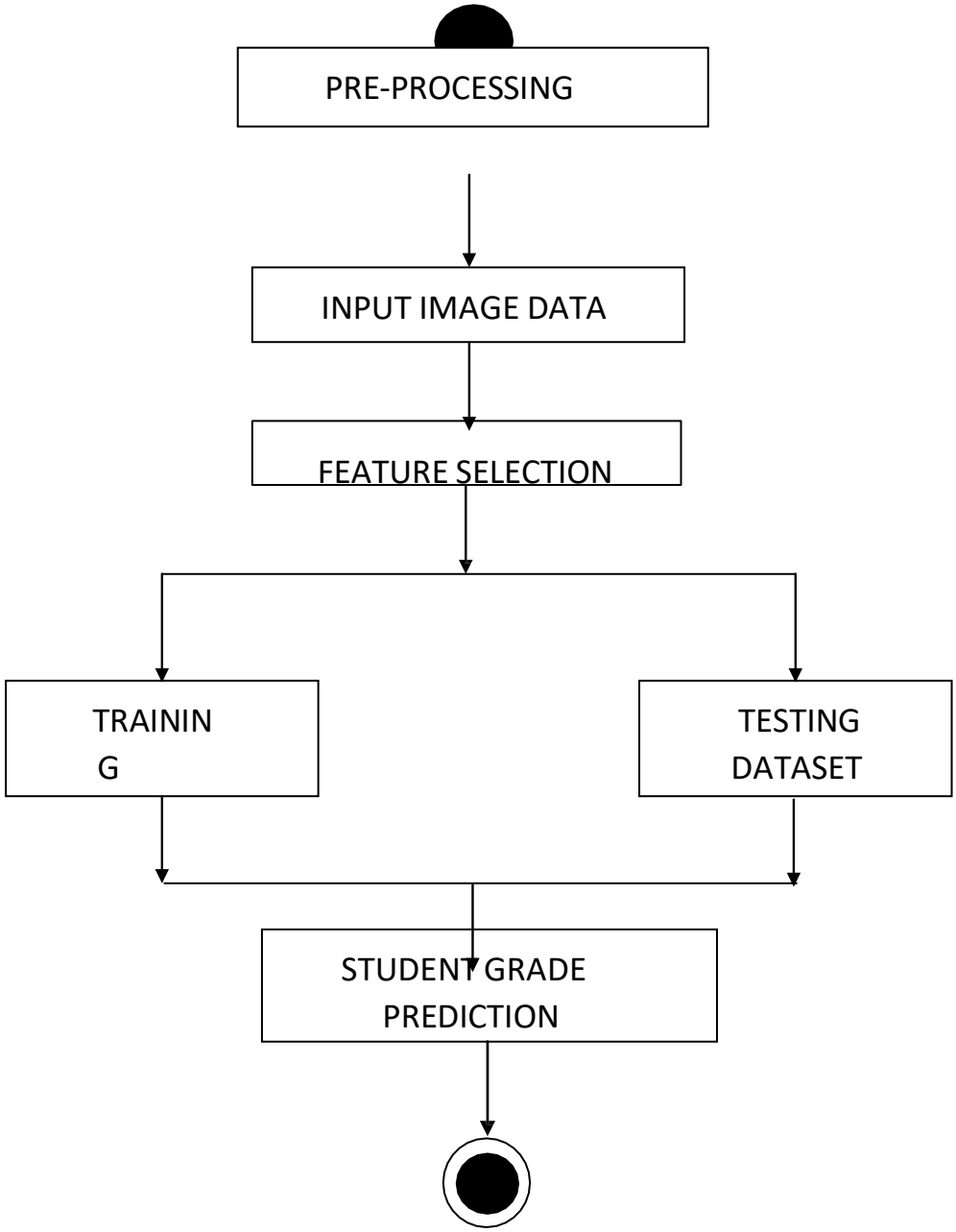


FIGURE.NO 6.2.5 ACTIVITY DIAGRAM

Describes the workflow — input, preprocessing, training/testing, prediction, and final result display.

IMPLEMENTATIONS

Modules

- Data Selection and Loading
- Data Preprocessing
- Future Selection
- Classification
- Prediction
- Result Generation

DATA SELECTION AND LOADING

- The data selection is the process of selecting the data for Student Grade dataset.
- In this project, examine the undergraduate student.
- The dataset which contains the information about the school, sex, age, address, famsize, Pstatus, Medu, Fedu, Mjob, Fjob, reason, guardian, traveltime, studytime, failures, schoolsup, famsup, paid, activities, nursery, higher, internet, romantic, famrel, freetime, goout, Dalc, Walc, health, absences, passed.

DATA PREPROCESSING

- pre- is the process of removing the unwanted data from the dataset.
- Missing data processing removal
- Encoding Categorical data
- Missing data removal: In this process, the null values such as missing values and Nan values are replaced by 0.
- Encoding Categorical data: That categorical data is defined as variables with a finite set of label values. That most machine learning algorithms require numerical input and output variables.

SPLITTING DATASET INTO TRAIN AND TEST DATA

- Data splitting is the act of partitioning available data into two portions, usually for cross-validate purposes.
- One Portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
- Separating image data into training and testing sets is an important part of evaluating image processing models.
- Typically, when you separate a data set into a training set and testing set, most of the image data is used for training, and a smaller portion of the data is used for testing.

CLASSIFICATION

Naive Bayes classifiers are a collection of classification algorithms based on Bayes‘ Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. An example of a decision tree can be explained using above binary tree.

K nearest neighbours is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). KNN has been used in statistical estimation and pattern recognition and non-parametric technique.

A Support vector machine (SVM) model is basically a representation of different classes in a hyper plane in multidimensional space. The hyper plane will be generated in an iterative manner by SVM so that the error can be minimized. The goal of SVM is to divide the datasets into classes to find a maximum marginal hyper plane

A multilayer perceptron is a class of feed forward artificial neural network. The term MLP is used ambiguously, sometimes loosely to mean any feed forward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptron.

1D CNN can perform activity recognition task from accelerometer data, such as if the student is pass or fail. This data has 2 dimensions. Similarly, 1D CNNs are also used on audio and text data since we can also represent the sound and texts as a time series data.

In the machine learning terminology Classification refers to a predictive modelling problem where the input data is classified as one of the predefined labelled classes.

PREDICTION

- It‘s a process of predicting the student successful attributes from the dataset.
- This project will effectively predict the data from dataset by enhancing the performance of the overall prediction results.

RESULT GENERATION

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

- Accuracy
- Precision
- Recall
- F1-measure
- Sensitivity
- Specificity

6.2 DATA MODEL

The **data model** defines how student-related data is structured, stored, and related. It forms the foundation for creating databases and AI model input/output.

DATASET CREATION TABLE

Dataset Creation Table refers to a structured table used to define and organize the data collected for a project or software system. It contains columns (fields/attributes) and rows (records) that represent the data to be processed or analyzed.

6.2.1 STUDENT DATASET

#	Gender	Race/Ethnicity	Parental Level of Education	Lunch	Test Preparation	Math Score	Reading Score	Writing Score
1	Female	Group B	Bachelor's Degree	Standard	None	72	72	74
2	Female	Group C	Some College	Standard	Completed	69	90	88
3	Female	Group B	Master's Degree	Standard	None	90	95	93
4	Male	Group A	Associate's Degree	Free/Reduced	None	47	57	44
5	Male	Group C	Some College	Standard	None	76	78	75
6	Female	Group B	Associate's Degree	Standard	None	71	83	78
7	Female	Group B	Some College	Standard	Completed	88	95	92
8	Male	Group B	Some College	Free/Reduced	None	40	43	39
9	Male	Group D	High School	Free/Reduced	Completed	64	64	67
10	Female	Group B	High School	Free/Reduced	None	38	60	50
11	Male	Group C	Associate's Degree	Standard	None	58	54	52
12	Male	Group D	Associate's Degree	Standard	None	40	52	43
13	Female	Group B	High School	Standard	None	65	81	73
14	Male	Group A	Some College					

TABLE.NO 6.2.1 STUDENT DATASET

A **Dataset Creation Table** is a tabular format where data is collected and arranged with specific attributes (columns) and values (rows), used for training, testing, or storing information in software applications or machine learning projects.

6.2.2 STUDENT DATA STRUCTURE

A **Student Data Structure** is a structured format used in programming or databases to store and manage information related to students. It typically includes various attributes like name, ID, age, gender, marks, and more.

Column	Data Type	Explanation
school	VARCHAR(10)	Stores school code like 'GP' or 'MS' (text).
sex	CHAR(1)	Stores gender, like 'M' or 'F'.
age	INT	Student's age in years.
address	CHAR(1)	Urban ('U') or rural ('R').
famsize	VARCHAR(5)	Family size, like 'LE3' or 'GT3'.
Pstatus	CHAR(1)	Parent status ('T' for together, 'A' for apart).
Medu, Fedu	INT	Education level of mother and father.
Mjob, Fjob	VARCHAR(20)	Job title of mother and father.
reason	VARCHAR(20)	Reason for school choice.
guardian	VARCHAR(20)	Guardian name (mother, father, other).
traveltime to Walc	INT	Numeric values for different personal and academic factors.
schoolsup to romantic	VARCHAR(5)	Yes/No values for support and activity participation.
health	INT	Health condition rating (1–5).
absences	INT	Number of school absences.
G1, G2, G3	INT	Grades for 3 evaluation periods.

TABLE.NO 6.2.2 STUDENT DATA STRUCTURE

6.2.3 STUDENT_PERFORMANCE_DATA

This name reflects that the table is designed to hold student information related to their academic performance, which is used for AI-based prediction.

Field Name	Data Type	Description
`student_id`	INT	Unique ID for each student
`gender`	VARCHAR(10)	Gender of the student
`age`	INT	Age in years
`parental_education`	VARCHAR(50)	Education level of parent
`lunch`	VARCHAR(20)	Type of lunch received
`test_prep_course`	VARCHAR(10)	Whether test preparation was completed
`math_score`	INT	Math score out of 100
`reading_score`	INT	Reading score out of 100
`writing_score`	INT	Writing score out of 100
`result`	VARCHAR(10)	Predicted result (Pass/Fail)

TABLE.NO 6.2.3 STUDENT_PERFORMANCE_DATA

CHAPTER VII

SYSTEM TESTING

7.1. SOFTWARE TESTING

Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is Defect free. It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest.

The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements. Some prefer saying Software testing definition as a White Box and Black Box Testing. In simple terms, Software Testing means the Verification of Application Under Test (AUT). This Software Testing course introduces testing software to the audience and justifies the importance of software testing.

Test plan

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

Validation

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

Basics of software testing

There are two basics of software testing: black box testing and white box testing.

Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

White box Testing

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing.
Black box testing is often used for validation and white box testing is often used for verification.

Types of testing

There are many types of testing like

- UNIT TESTING
- INTEGRATION TESTING
- FUNCTIONAL TESTING
- SYSTEM TESTING
- STRESS TESTING
- PERFORMANCE TESTING
- USABILITY TESTING
- ACCEPTANCE TESTING
- REGRESSION TESTING
- BETA TESTING

7.1.1 UNIT TESTING

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

7.1.2 INTEGRATION TESTING

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

7.1.3 FUNCTIONAL TESTING

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

7.1.4 SYSTEM TESTING

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

7.1.5 STRESS TESTING

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

7.1.6 PERFORMANCE TESTING

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

7.1.7 USABILITY TESTING

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use,how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

7.1.8 ACCEPTANCE TESTING

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

7.1.9 REGRESSION TESTING

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing

CHAPTER VIII

CONCLUSION

In this study, the different types of machine learning classifier is predict the successful attributes from the dataset. The student grade data is taken as input data and applied into pre-processing method. In pre-processing method the clean the dataset and label encoding the dataset. Then it processed into feature selection method, in this method the dataset is split into training dataset and testing dataset. Finally the classification method is used to predict the successful attributes. Machine learning algorithmslike NB, DT, KNN, SVM and MLP and Deep learning algorithm like, ANN and CNN is implemented and predict the result based on accuracy, precision, recall and f1-measure.

8.1 FUTURE ENHANCEMENT

We are tuning the predictive models using additional data, seeking to further understand the students‘ behavior by delving deeper into their programming process, and conducting interviews that hopefully will shed further light on students‘ working practices as well asto those students who were misclassified. We are also per-forming targeted interventions within the studied context

CHAPTER XI

SYSTEM IMPLEMENTATION

9.1 SOURCE CODES

```
import pandas as pd
importmatplotlib.pyplot as plt
importseaborn as sns
import warnings
warnings.filterwarnings('ignor')
#Read in csv file into pandas
dataframe data =
pd.read_csv("studentdata.csv")
data1=data.cop
print(data.head())
#check missing values
print ('Dataset contain null:\t',data.isnull().values.any())
#Label Encoding
fromsklearn.preprocessing
import LabelEncoder
le=LabelEncoder()
```

```
data.iloc[:,0]=le.fit_transform(d
ata.iloc[:,0])
data.iloc[:,1]=le.fit_transf
data.iloc[:,3]=le.fit_transform(d
ata.iloc[                    :,3])
data.iloc[:,4]=le.fit_transform(d
ata.iloc[:,4])
data.iloc[:,5]=le.fit_transform(d
ata.iloc[:,5])
data.iloc[:,8]=le.fit_transform(d
ata.iloc[:,8])
data.iloc[:,9]=le.fit_transform(d
ata.iloc[:,9])
data.iloc[:,10]=le.fit_transform(
data.iloc[:,10])
data.iloc[:,11]=le.fit_transform(
data.iloc[:,11])
data.iloc[:,15]=le.fit_transform(
data.iloc[:,15])
data.iloc[:,16]=le.fit_transform(
```

```
data.iloc[:,16])
data.iloc[:,17]=le.fit_transform(
data.iloc[:,17])
orm(data.iloc[:,1])
data.iloc[:,18]=le.fit_transform(data.iloc
[:,18])
data.iloc[:,19]=le.fit_transform(data.iloc
[:,19])
data.iloc[:,20]=le.fit_transform(data.iloc
[:,20])
data.iloc[:,21]=le.fit_transform(data.iloc
[:,21])
data.iloc[:,22]=le.fit_transform(data.iloc
[:,22])
data.iloc[:,30]=le.fit_transform(data.iloc
[:,30])
print(data.head())
x = data.drop("passed",axis =
1).values y = data['passed']
```

```
fromsklearn.model_selection import

train_test_split #Split the dataset

x_train,x_test,y_train,y_test=train_test_split(x, y, test_size=0.2, random_state=7)

fromsklearn.naive_bayes import GaussianNB

fromsklearn.metrics import accuracy_score, confusion_matrix,

classification_report importscikitplot as skplt

'"NAIVE BAYES"'

nb_clf = GaussianNB()

nb_clf.fit(x_train,y_train)

nb_ypred=nb_clf.predict(x_test

) print('\n')

print("------Accuracy ")

nb=accuracy_score(y_test, nb_ypred)*100
NB=('NAIVE BAYES Accuracy:',accuracy_score(y_test,

nb_ypred)*100,'%') print(NB)

print('\n')

print("------Classification Report")
print(classification_report(nb_ypred,y_te

st)) print('\n')

print('Confusion_matrix')

nb_cm = confusion_matrix(y_test,
```

```
nb_ypred) print(nb_cm)

print('\n')

tn              =

nb_cm[0][0]

fp              =

nb_cm[0][1]

fn              =

nb_cm[1][0]

tp              =

nb_cm[1][1]

Total_TP_FP=nb_cm[0][0]+nb_cm[0][1]

Total_FN_TN=nb_cm[1][0]+nb_cm[1][1] specificity = tn / (tn+fp)

nb_specificity=format(specificity,'.3f') sensitivity = tp / (fn + tp)


nb_sensitivity=format(sensitivity,'.

3f')

print('NB_specificity:',nb_specificit

y)                              print('\n')

print('NB_sensitivity:',nb_sensitivit

y) print('\n')

plt.figure()
```

```
skplt.estimators.plot_learning_curve(GaussianNB(),

x_train, y_train, cv=7, shuffle=True, scoring="accuracy",

n_jobs=-1, figsize=(6,4), title_fontsize="large", text_fontsize="large",

title="Naive Bayes Digits Classification Learning Curve");
plt.figure()

sns.heatmap(confusion_matrix(y_test,nb_ypred),a

nnot = True) plt.title("Confusion Matrix")

plt.xlabel("Predicte

d")

plt.ylabel("True")

plt.show()


fromsklearn.tree import DecisionTreeClassifier

'''Decision Tree'''

# Create Decision Tree

classifer object print('Decision

Tree')


'''Analysis Report'''

print()

a =
```

```
DecisionTreeClassifie
r()          a          =
a.fit(x_train,y_train)
dt_ypred                =
a.predict(x_test)
print()
print("------Accuracy       ")
DT=accuracy_score(y_test, dt_ypred)*100
dt=("DECISION      TREE      Accuracy:",accuracy_score(y_test,
dt_ypred)*100,'%') print(dt)
print('\n')
print("------Classification Report        ")
print(classification_report(dt_ypred
,y_test)) print('\n')
print('Confusion_matrix')
dt_cm      =      confusion_matrix(y_test,
dt_ypred) print(dt_cm)
print('\n')
tn          =
dt_cm[0][0]
fp          =
```

```
dt_cm[0][1]
fn =
dt_cm[1][0]
tp =
dt_cm[1][1]
Total_TP_FP=dt_cm[0][0]+dt_cm[
0][1]
Total_FN_TN=dt_cm[1][0]+dt_cm
[1][1] specificity = tn / (tn+fp)
dt_specificity=format(specificity,'.
3f') sensitivity = tp / (fn + tp)
dt_sensitivity=format(sensitivity,'.3
f')
print('DT_specificity:',dt_specificit
y) print('\n')
print('DT_sensitivity:',dt_sensitivit
y) plt.figure()

skplt.estimators.plot_learning_curve(DecisionTreeClassifier() ,
x_train, y_train, cv=7, shuffle=True, scoring="accuracy",
```

```
n_jobs=-1, figsize=(6,4), title_fontsize="large", text_fontsize="large",

title="Decision Tree Digits Classification Learning Curve");
plt.figure()

sns.heatmap(confusion_matrix(y_test,dt_ypred),a

nnot = True) plt.title("Confusion Matrix")

plt.xlabel("Predicte

d")

plt.ylabel("True")

plt.show()

fromsklearn.neighbors import KNeighborsClassifier

'"KNN"'

print('K-Nearest Neighbour')

"'Analysis     Report"'

print('\n')

"'KNeighbour_classif

ier"'

knn                    =

KNeighborsClassifier()

knn.fit(x_train,      y_train)

knn_ypred              =

knn.predict(x_test)
```

```
print('\n')

print("------Accuracy       ")

KNN=accuracy_score(y_test, knn_ypred)*100

knn=('K NEAREST Neighbors Accuracy:',accuracy_score(y_test,

knn_ypred)*100,'%') print(knn)

print('\n')

print("------Classification Report       ")
print(classification_report(knn_ypred,y_t

est)) print('\n')

print('Confusion_matrix')

knn_cm = confusion_matrix(y_test,

knn_ypred) print(knn_cm)

print('\n')

tn =

knn_cm[0][0]

fp =

knn_cm[0][1]

fn =

knn_cm[1][0]

tp =

knn_cm[1][1]
```

```
Total_TP_FP=knn_cm[0][0]+knn_
cm[0][1]
Total_FN_TN=knn_cm[1][0]+knn
_cm[1][1] specificity = tn / (tn+fp)
knn_specificity=format(specificity,'
.3f') sensitivity = tp / (fn + tp)
knn_sensitivity=format(sensitivity,'
.3f')
print('KNN_specificity:',knn_speci
ficity) print('\n')
print('KNN_sensitivity:',knn_sensit
ivity) print('\n')
plt.figure()
skplt.estimators.plot_learning_curve(KNeighborsClassifier(n_neighbors=5)
, x_train, y_train, cv=7, shuffle=True, scoring="accuracy",
n_jobs=-1, figsize=(6,4), title_fontsize="large",
text_fontsize="large", title="KNearestNeighbors Digits
Classification Learning Curve");
plt.figure()
sns.heatmap(confusion_matrix(y_test,knn_ypred),
```

```
annot = True) plt.title("Confusion Matrix")
plt.xlabel("Predicte
d")
plt.ylabel("True")
plt.show()
fromsklearn.svm import SVC
"'SVM algorithm"'
print('Super Vector
Machine')
svclassifier =
SVC(kernel='linear')
svclassifier.fit(x_train, y_train)
svm_ypred =
svclassifier.predict(x_test)
"'Analysis Report"'
print()
print("------Accuracy ")
SVM=accuracy_score(y_test,svm_ypred)*100
svm=('SVM
Accuracy:',accuracy_score(y_test,svm_ypred)*100,'
```

```
%') print(svm)

print('\n')

print("------Classification Report ")

print(classification_report(svm_ypred,y_t

est)) print('\n')

print('Confusion_matrix')

svm_cm = confusion_matrix(y_test,

svm_ypred) print(svm_cm)

print('\n')

tn =

svm_cm[0][0]

fp =

svm_cm[0][1]

fn =

svm_cm[1][0]

tp =

svm_cm[1][1]

Total_TP_FP=svm_cm[0][0]+svm_cm[

0][1]

Total_FN_TN=svm_cm[1][0]+svm_cm
```

```
[1][1] specificity = tn / (tn+fp)
SVM_specificity=format(specificity,'.3f
') sensitivity = tp / (fn + tp)
SVM_sensitivity=format(sensitivity,'.3f
')print('SVM_specificity:',SVM_specifi
city) print('\n')
print('SVM_sensitivity:',SVM_sensitivi
ty) plt.figure()
skplt.estimators.plot_learning_curve(SVC(kernel='linear') ,
x_train, y_train, cv=7, shuffle=True, scoring="accuracy",
n_jobs=-1, figsize=(6,4), title_fontsize="large",
text_fontsize="large", title="SVM Digits Classification
Learning Curve");
plt.figure()
sns.heatmap(confusion_matrix(y_test,svm_ypred),a
nnot = True) plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("True")
plt.show()
fromsklearn.neural_network import
MLPClassifier mlp = MLPClassifier()
```

```
mlp.fit(x_train,y_train)
mlp_pred =
mlp.predict(x_test)
MLP=
accuracy_score(y_test,mlp_pred)*
100 print("Multi
LayerPerceptron:",MLP,'%')
print("------Classification Report ")
print(classification_report(mlp_pred,y_te
st)) print('\n')
print('Confusion_matrix')
mlp_cm = confusion_matrix(y_test,
mlp_pred) print(mlp_cm)
print('\n')
tn =
mlp_cm[0][0]
fp =
mlp_cm[0][1]
fn =
mlp_cm[1][0]
```

```
tp =
mlp_cm[1][1]
Total_TP_FP=mlp_cm[0][0]+mlp_cm[0]
[1]
Total_FN_TN=mlp_cm[1][0]+mlp_cm[1
][1] specificity = tn / (tn+fp)
mlp_specificity=format(specificity,'.3f')
sensitivity = tp / (fn + tp)
mlp_sensitivity=format(sensitivity,
.3f')
print('MLP_specificity:',mlp_specif
icity) print('\n')
print('MLP_sensitivity:',mlp_sensit
ivity) plt.figure()
skplt.estimators.plot_learning_curve(MLPClassifier() ,
x_train, y_train, cv=7, shuffle=True, scoring="accuracy",
n_jobs=-1, figsize=(6,4), title_fontsize="large",
text_fontsize="large", title="MLP Digits Classification
Learning Curve");
plt.figure()
```

```
sns.heatmap(confusion_matrix(y_test,mlp_pred),a

nnot = True) plt.title("Confusion Matrix")

plt.xlabel("Predict)

plt.ylabel("True")

plt.show()
```

9.2 SCREEN SHOTS

INDEX PAGE

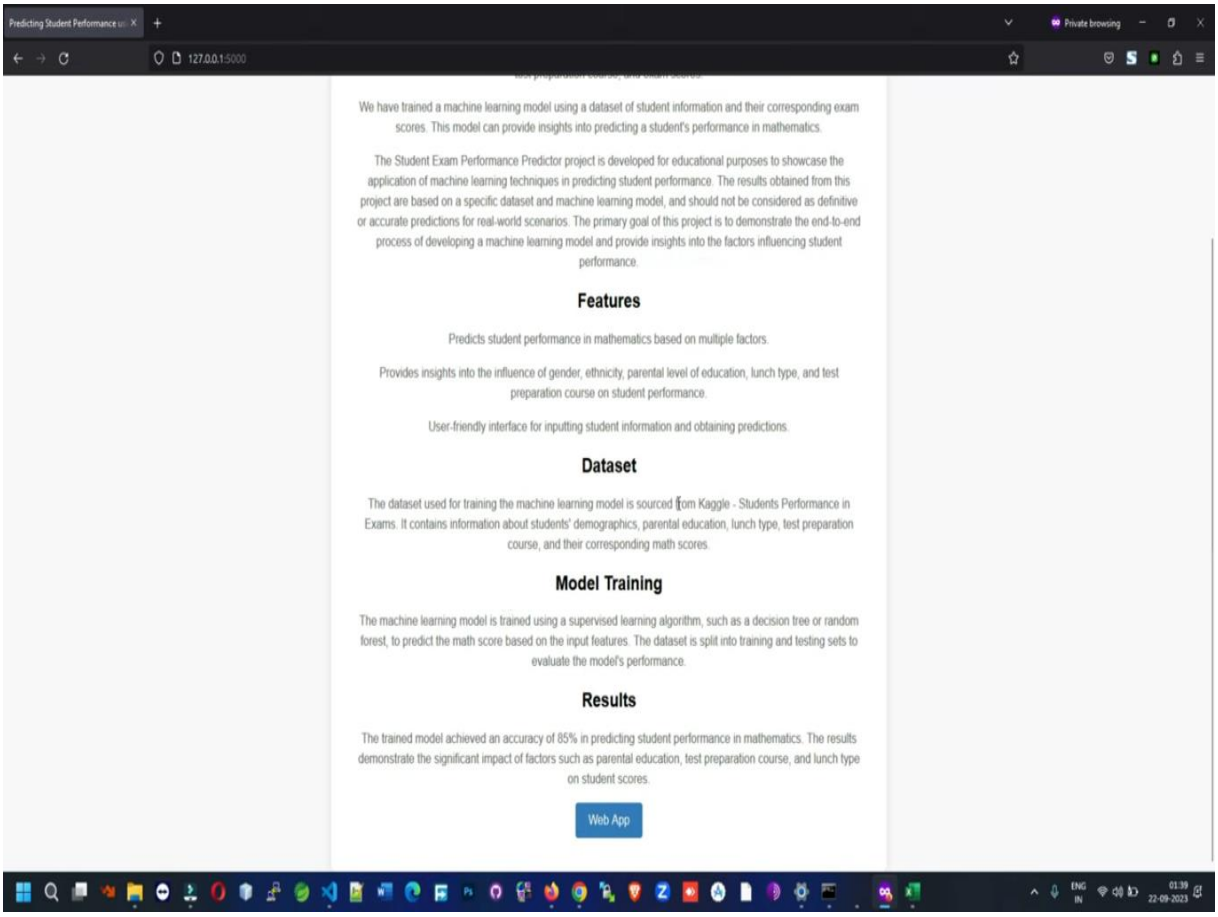


FIGURE.NO 9.2.1 INDEX PAGE

- displays the welcome or login screen of the application.
- provides access to start data upload, view predictions, or go to the home page.
- ensures only authorized users access the system.

HOME PAGE

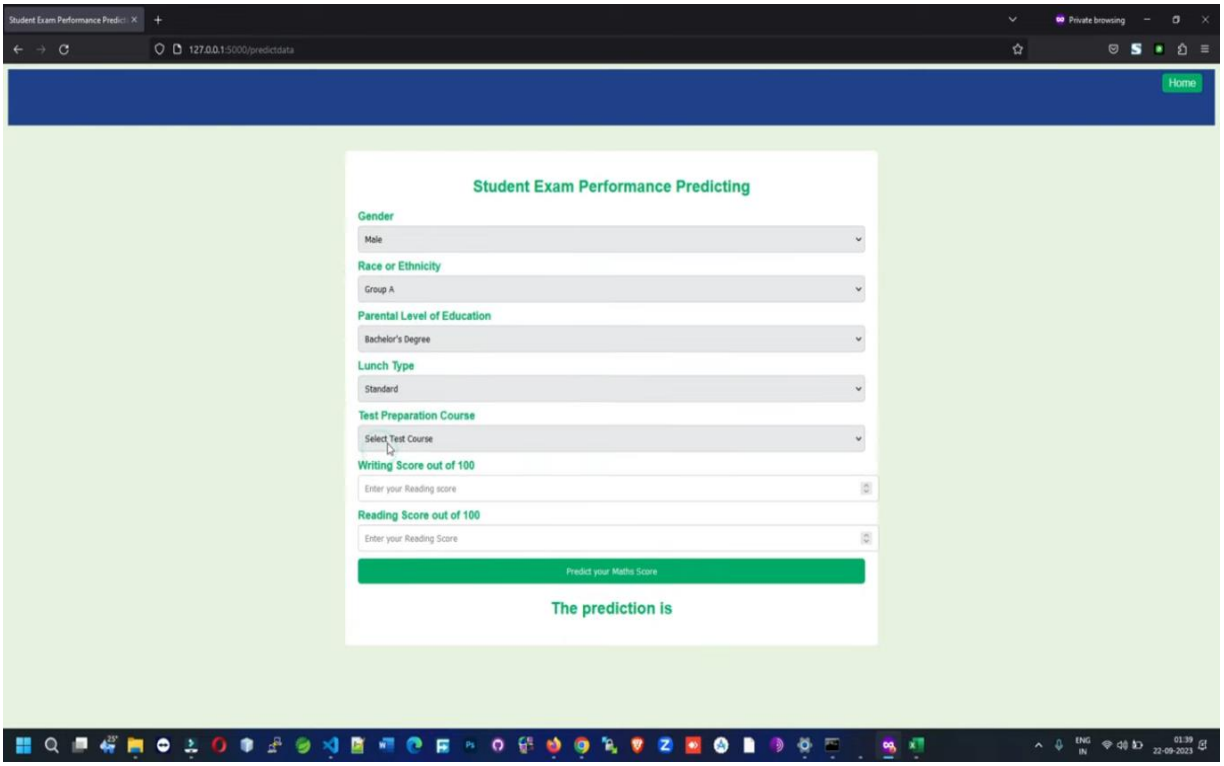


FIGURE.NO 9.2.2 HOME PAGE

- Shows the main dashboard layout after login.
- Contains navigation to modules like data upload, analysis, result, and logout.
- Acts as a central hub for interacting with the prediction system.

DATAFRAME

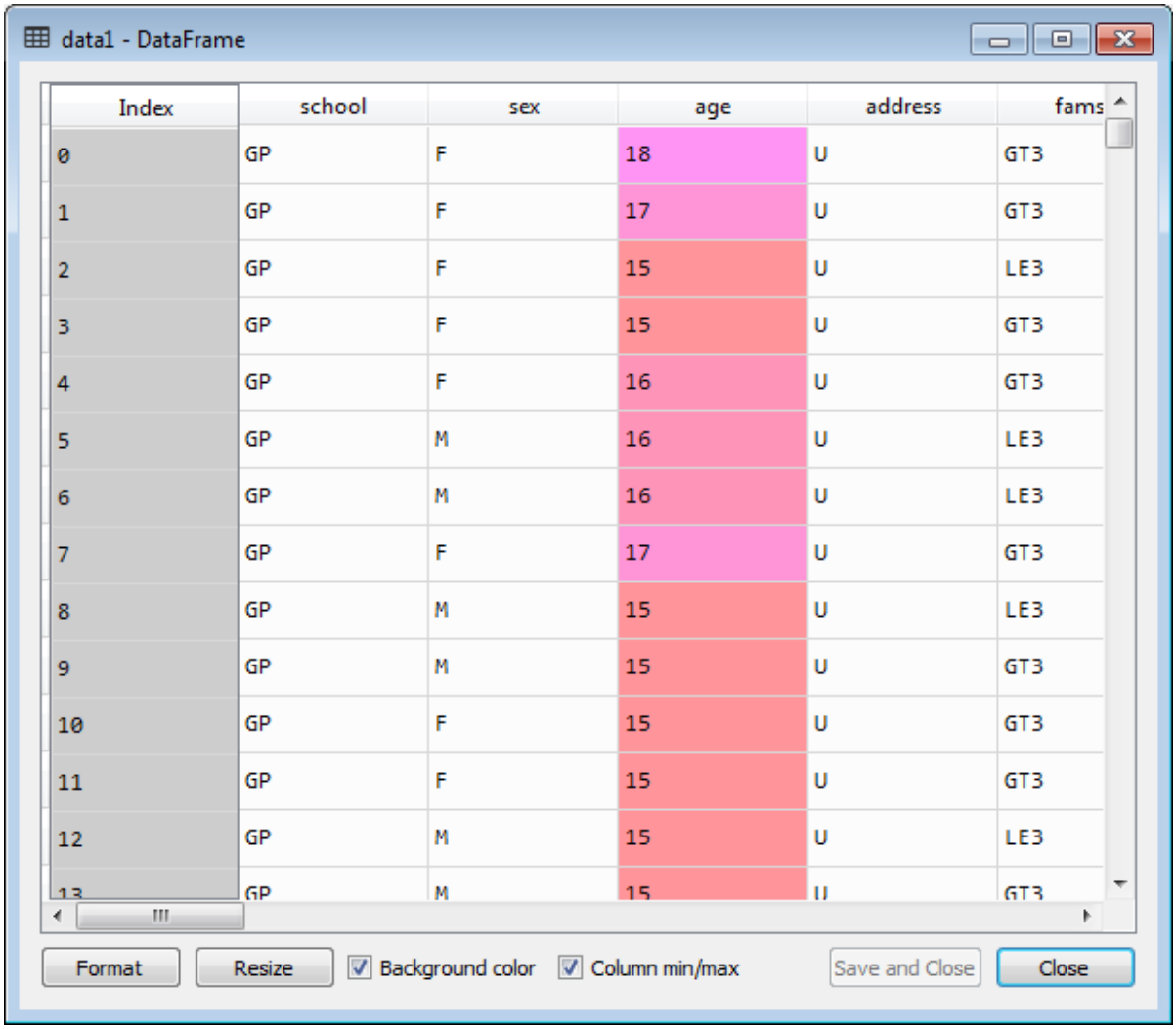
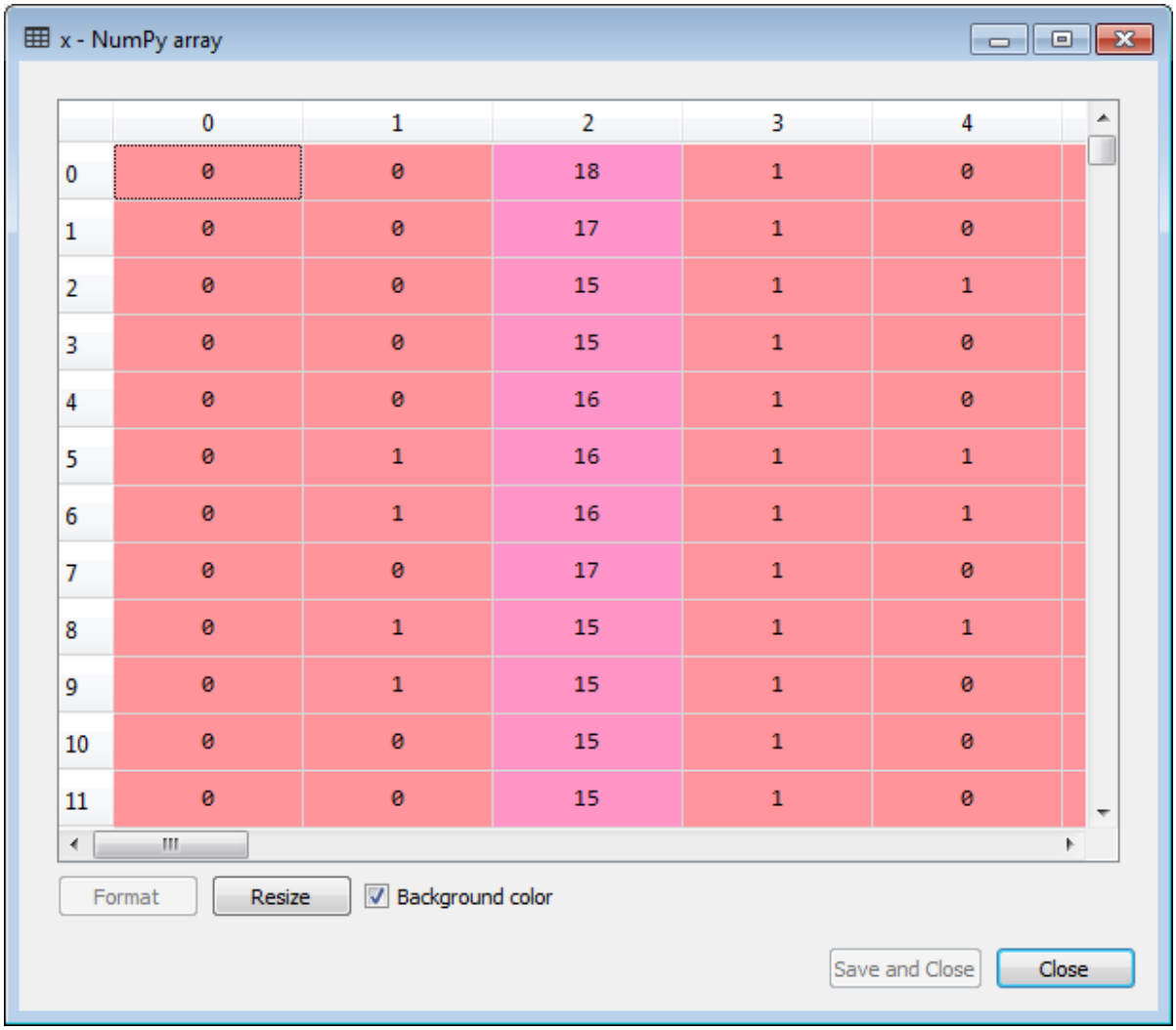


FIGURE.NO 9.2.3 DATAFRAME

- Displays student data in tabular format using pandas.
- Used to confirm successful loading of datasets (e.g., gender, scores, attendance).
- Enables quick preview before training.

NUMPYARRAY



	0	1	2	3	4
0	0	0	18	1	0
1	0	0	17	1	0
2	0	0	15	1	1
3	0	0	15	1	0
4	0	0	16	1	0
5	0	1	16	1	1
6	0	1	16	1	1
7	0	0	17	1	0
8	0	1	15	1	1
9	0	1	15	1	0
10	0	0	15	1	0
11	0	0	15	1	0

FIGURE.NO 9.2.4 NUMPYARRAY

Description: Shows the dataset converted into a NumPy array.

Function: Required format for machine learning models in Python (Scikit-learn).

Purpose: Ensures numerical data is ready for classification.

ARRAY Y-SERIES

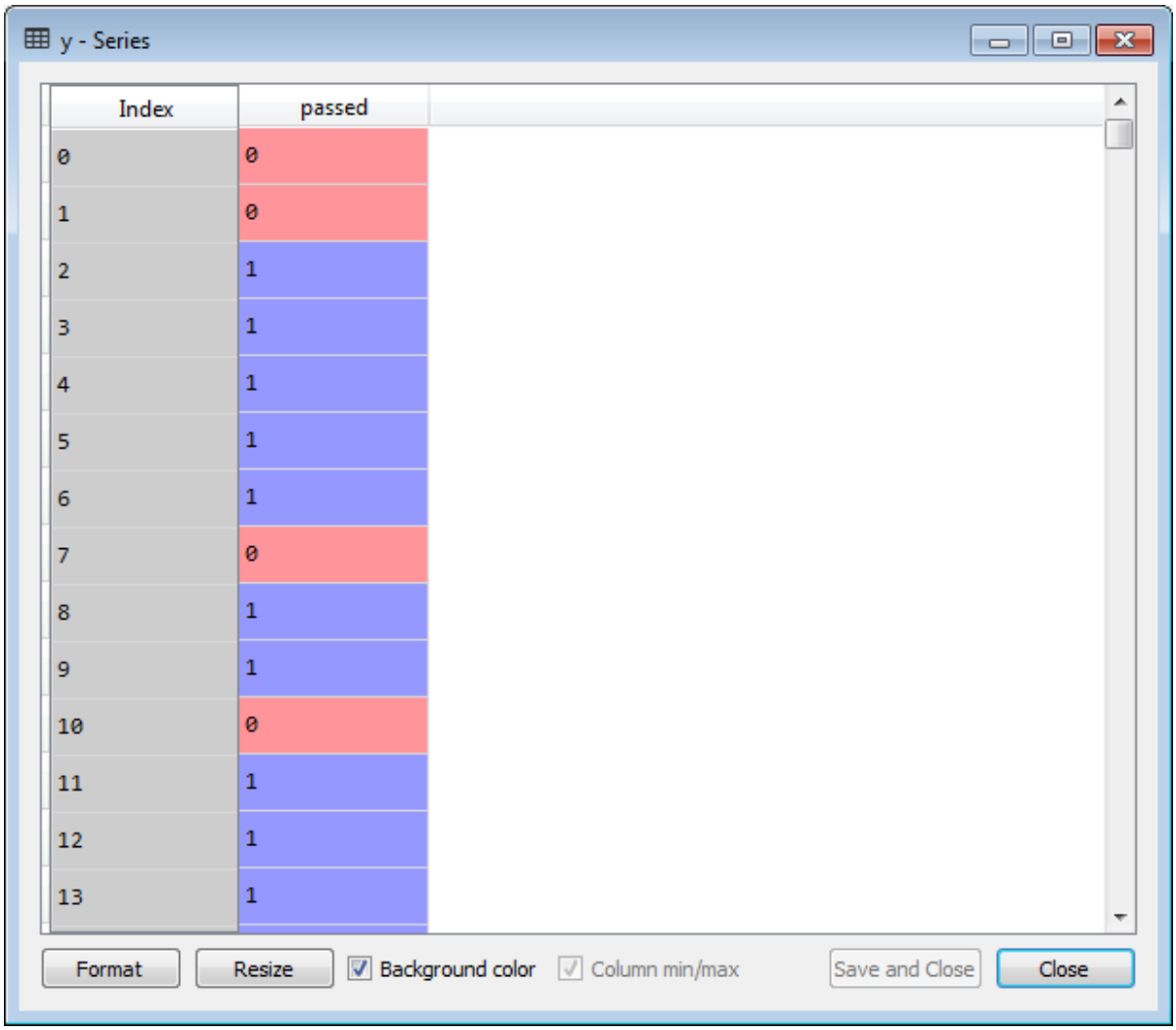
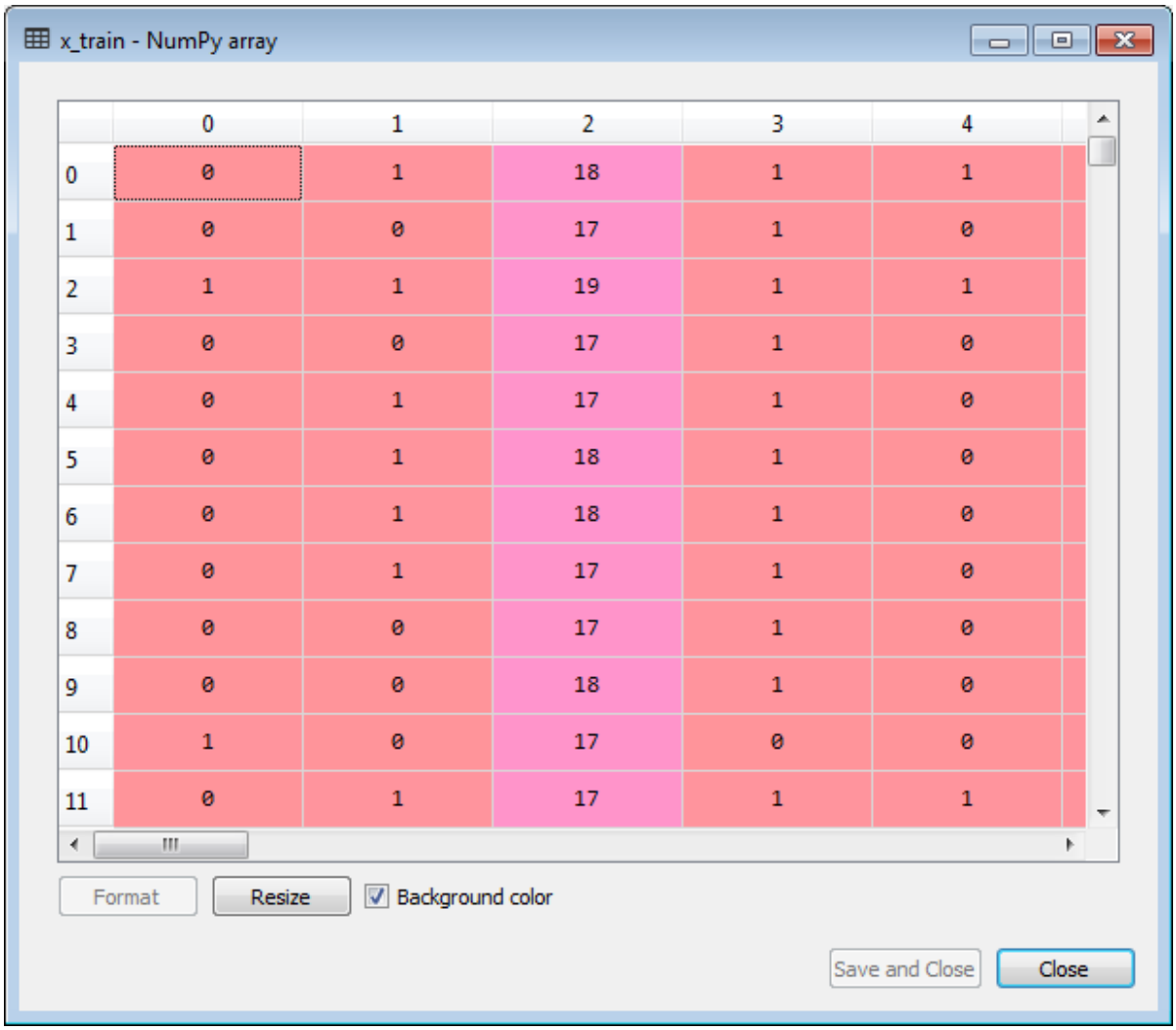


FIGURE.NO 9.2.5 ARRAY Y-SERIES

- **Description**: Displays the target/output variable, usually the `passed` column (Yes/No).
- **Function**: Used for training the models to predict if a student will pass or fail.
- **Purpose**: Ground truth for evaluating model performance

X-TRAIN NUMPY-ARRAY



	0	1	2	3	4
0	0	1	18	1	1
1	0	0	17	1	0
2	1	1	19	1	1
3	0	0	17	1	0
4	0	1	17	1	0
5	0	1	18	1	0
6	0	1	18	1	0
7	0	1	17	1	0
8	0	0	17	1	0
9	0	0	18	1	0
10	1	0	17	0	0
11	0	1	17	1	1

FIG.NO 9.2.6 X-TRAIN NUMPY-ARRAY

Description: Shows features used for training the model (e.g., scores, attendance).

Function: Divided from the full dataset using train_test_split.

Purpose: Teaches the model using historical student data.

Y-TRAIN -SERIES

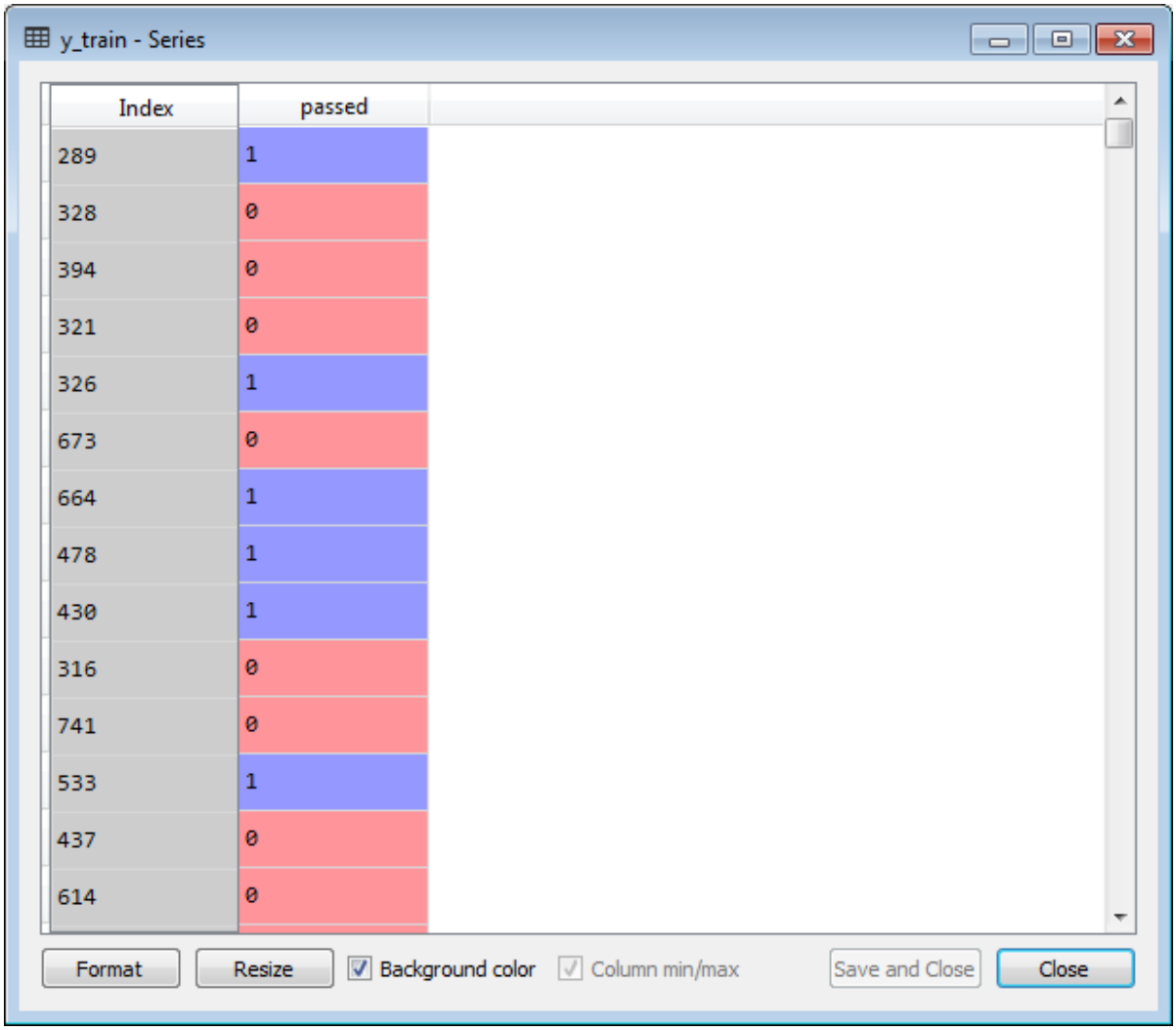


FIGURE.NO 9.2.7 Y-TRAIN -SERIES

Description: Output labels corresponding to the training data.

Function: Helps models learn relationships between features and final outcomes (pass/fail).

Purpose: Builds model prediction capability.

Y-TEST-SERIES

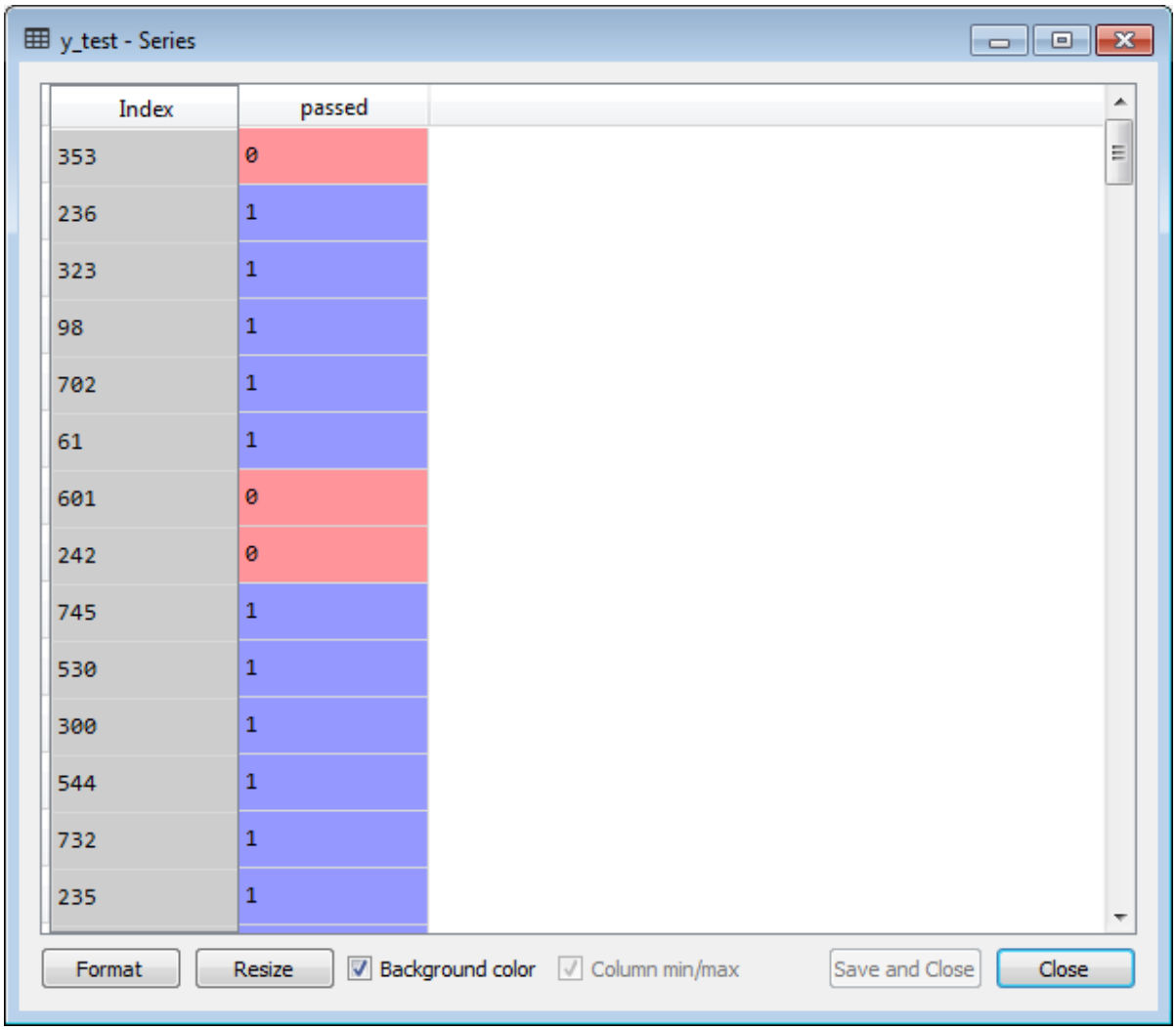


FIGURE.NO 9.2.8 Y-TEST-SERIES

Description: Actual pass/fail labels for the testing data.

Function: Used to compare model predictions and evaluate metrics like accuracy.

Purpose: Validates model performance on new data.

NB-NUMPY-ARRAY

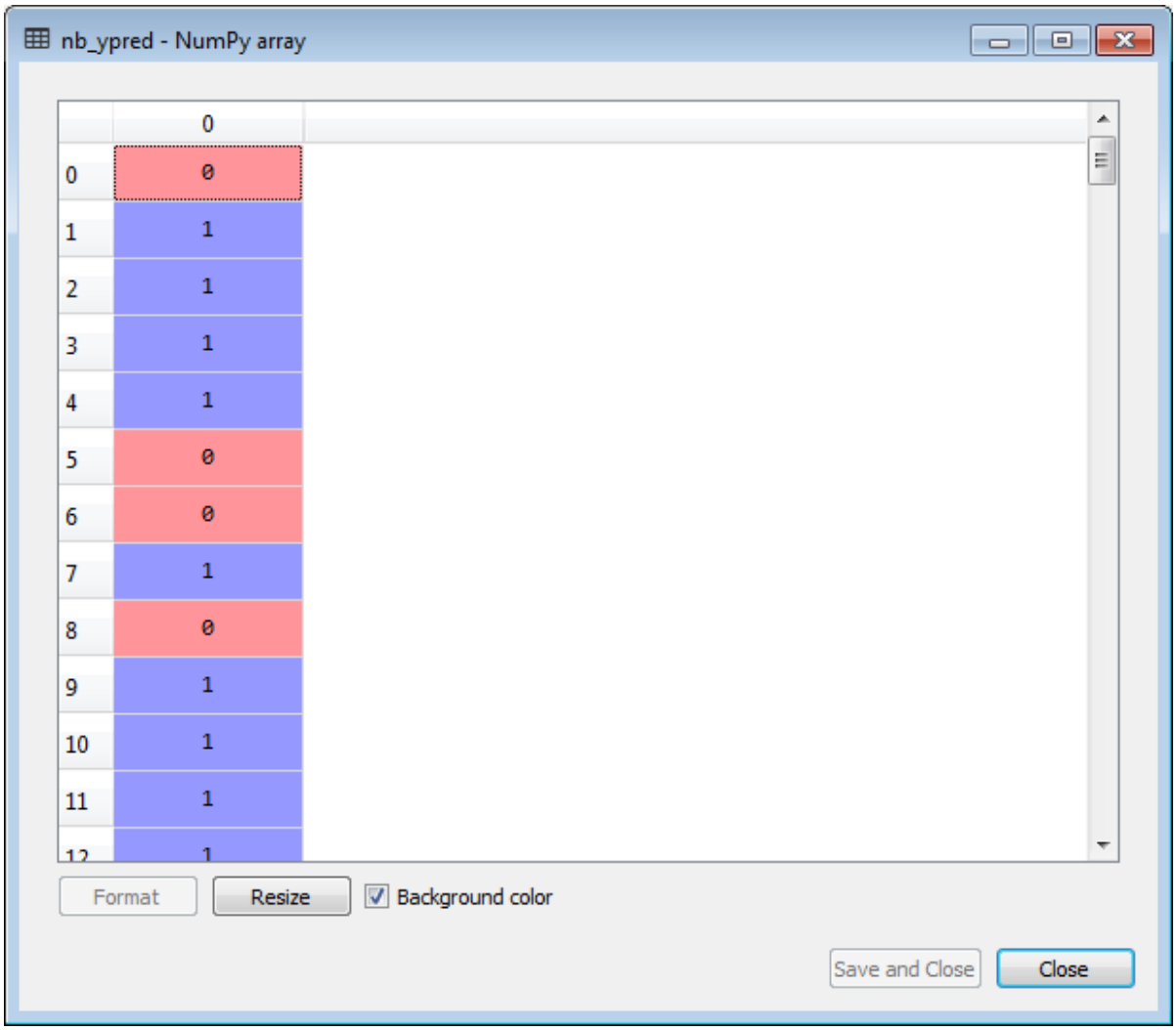


FIGURE.NO 9.2.9 NB-NUMPY-ARRAY

Description: Displays Naïve Bayes model predictions.

Function: Includes accuracy score, confusion matrix, and classification report.

Purpose: Understand model behavior and how it classifies students.

DT-NUMPY-ARRAY

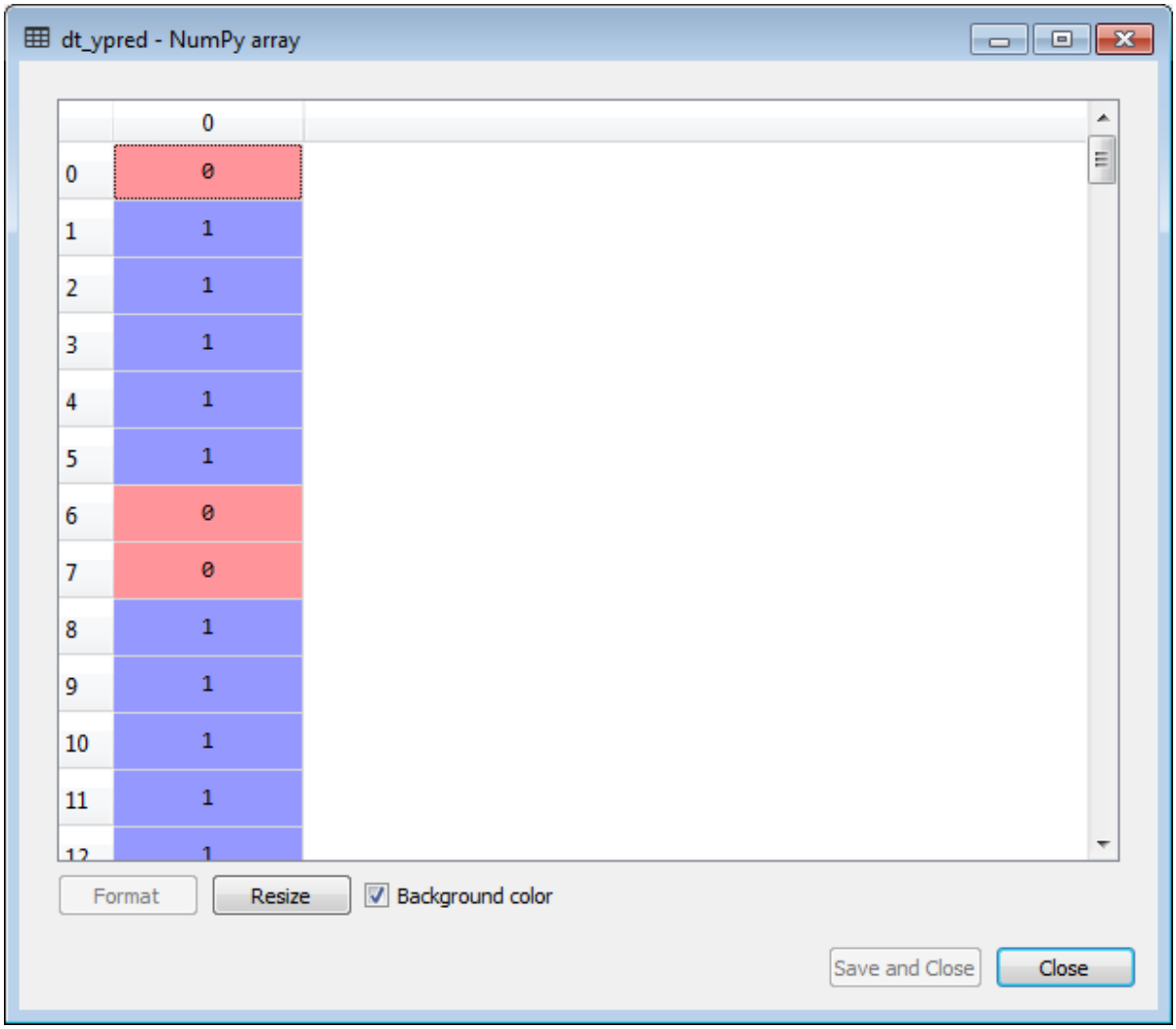


FIGURE.NO 9.2.10 DT-NUMPY-ARRAY

Description: Results from Decision Tree algorithm.

Function: Shows learning curve, performance metrics, and confusion matrix.

Purpose: Visualize how Decision Trees make decisions from input data.

KNN-NUMPY-ARRAY

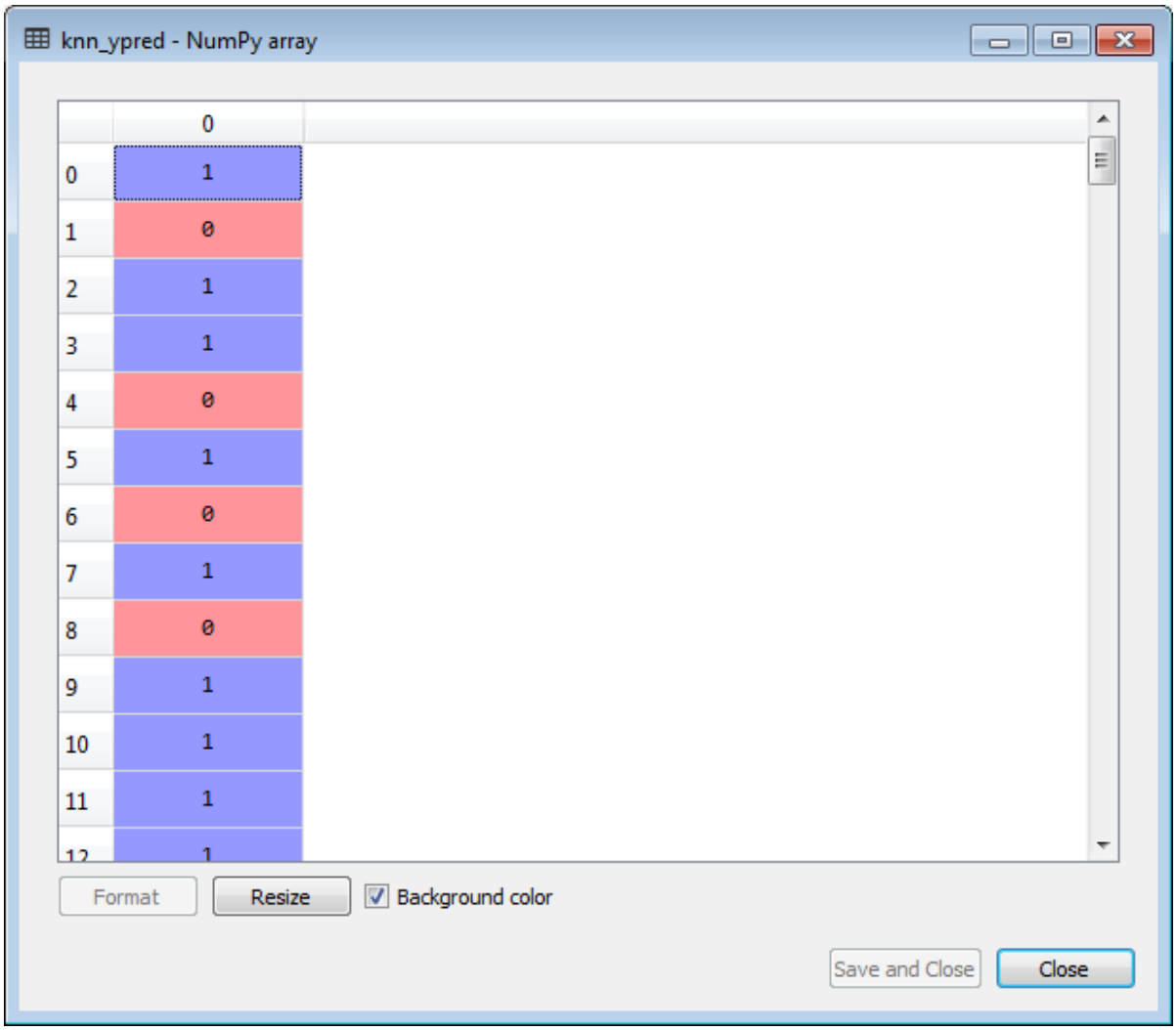


FIGURE.NO 9.2.11 KNN-NUMPY-ARRAY

Description: Predictions using K-Nearest Neighbors algorithm.

Function: Shows accuracy, sensitivity, specificity, and classification report.

Purpose: Compares performance of instance-based learning.

SVM-ARRAY

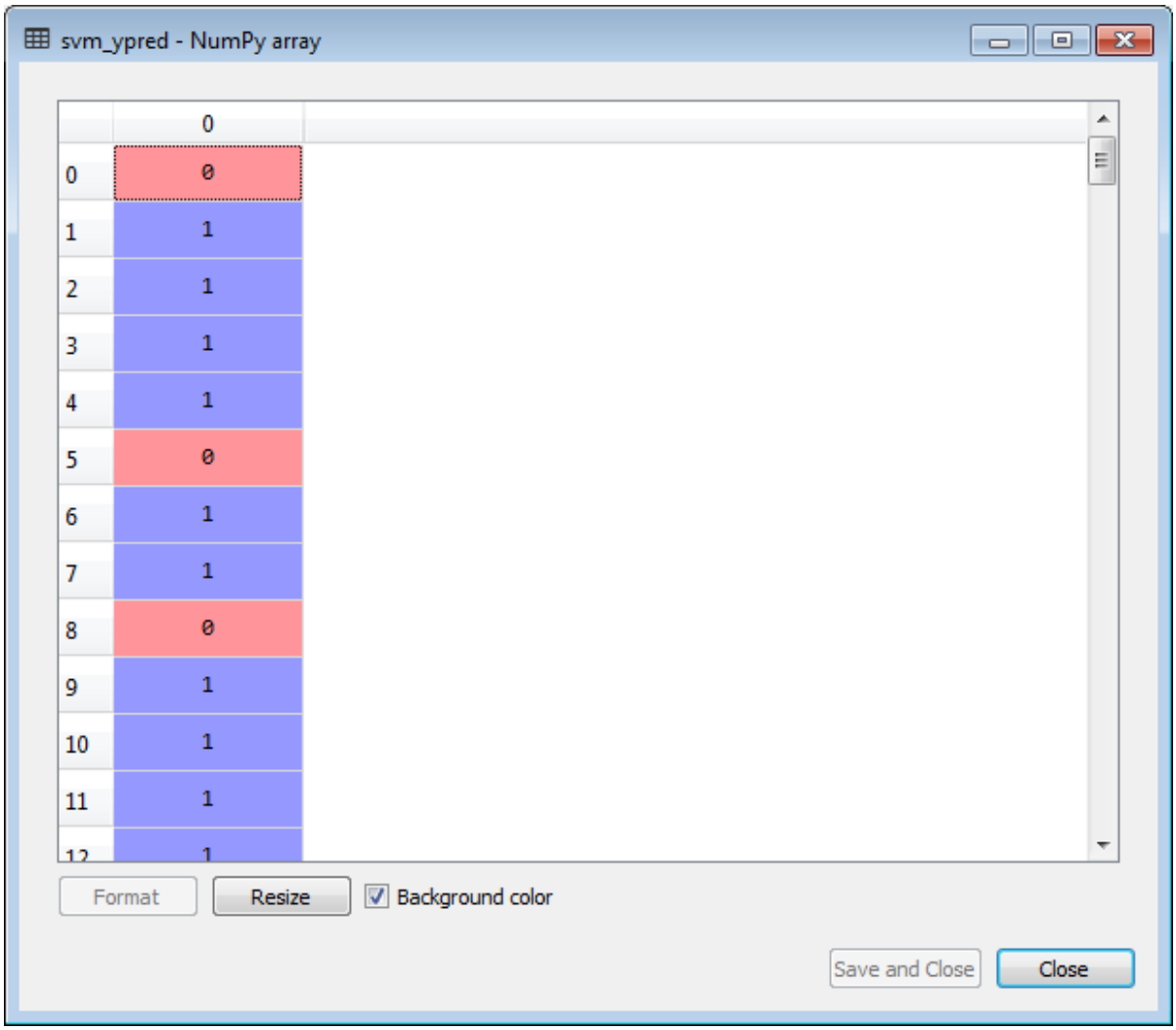


FIGURE.NO 9.2.12 SVM-ARRAY

Description: Results from Support Vector Machine model.

Function: Displays predicted values, confusion matrix, and performance graph.

Purpose: Highlights boundary-based classification performance.

MLP-NUMPY-ARRAY

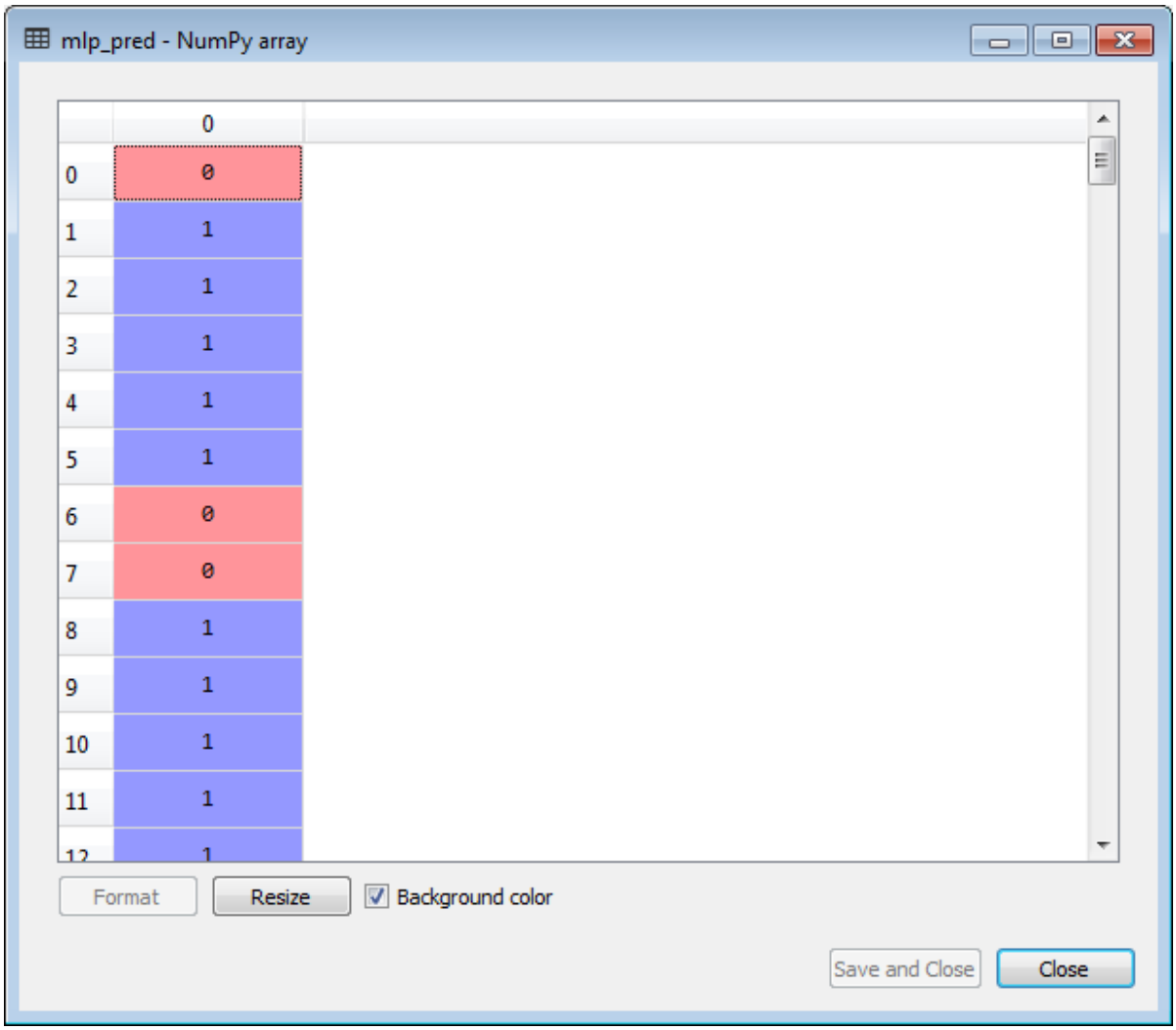


FIGURE.NO 9.2.13 MLP-Numpy-Array

Description: Output from Multilayer Perceptron (Neural Network) model.

Function: Displays model accuracy, sensitivity, specificity, and heatmap.

Purpose: Tests deep learning model accuracy in student success prediction.

DATASET 1

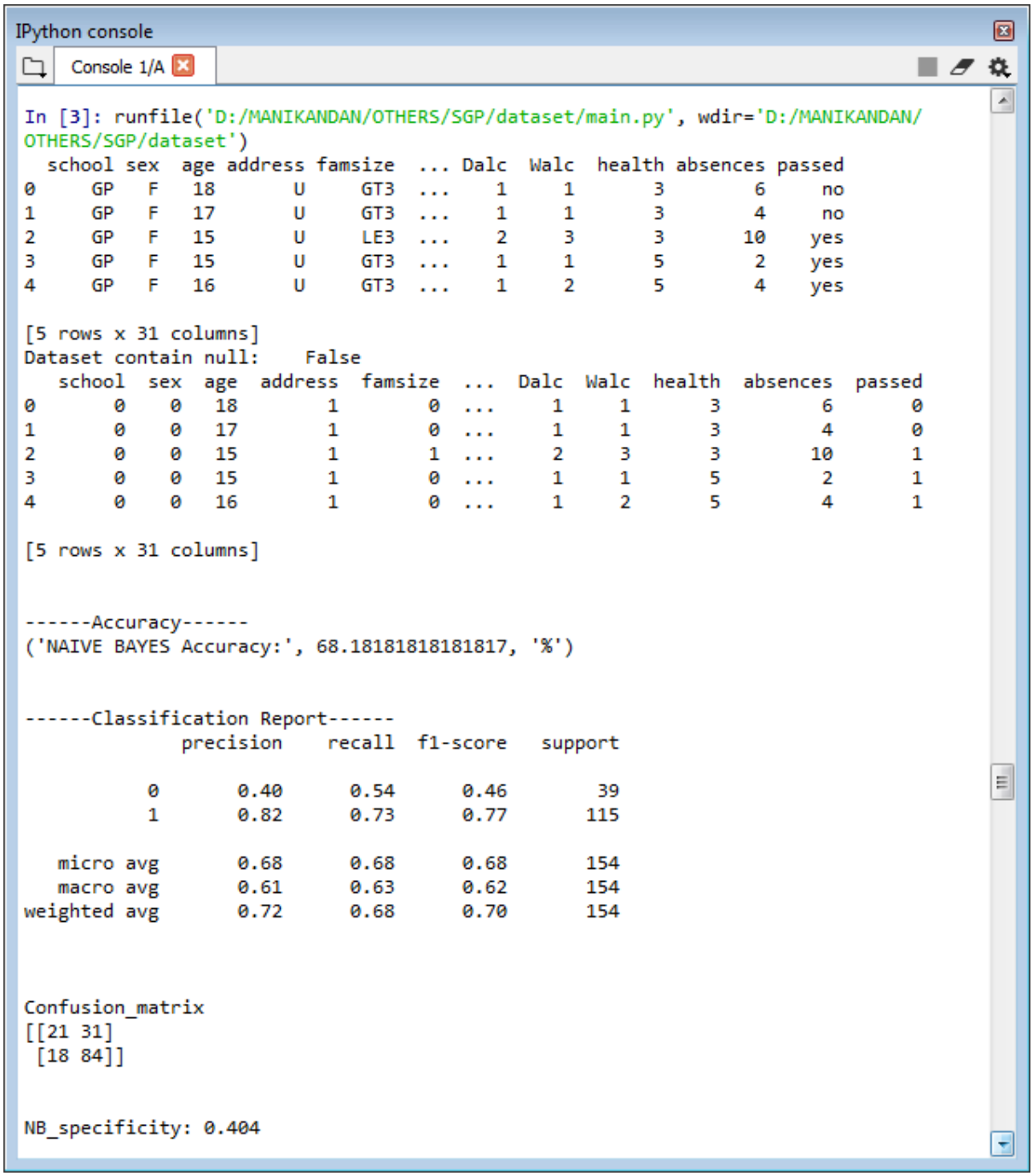


FIGURE.NO 9.2.14 DATASET 1

DATASET 2

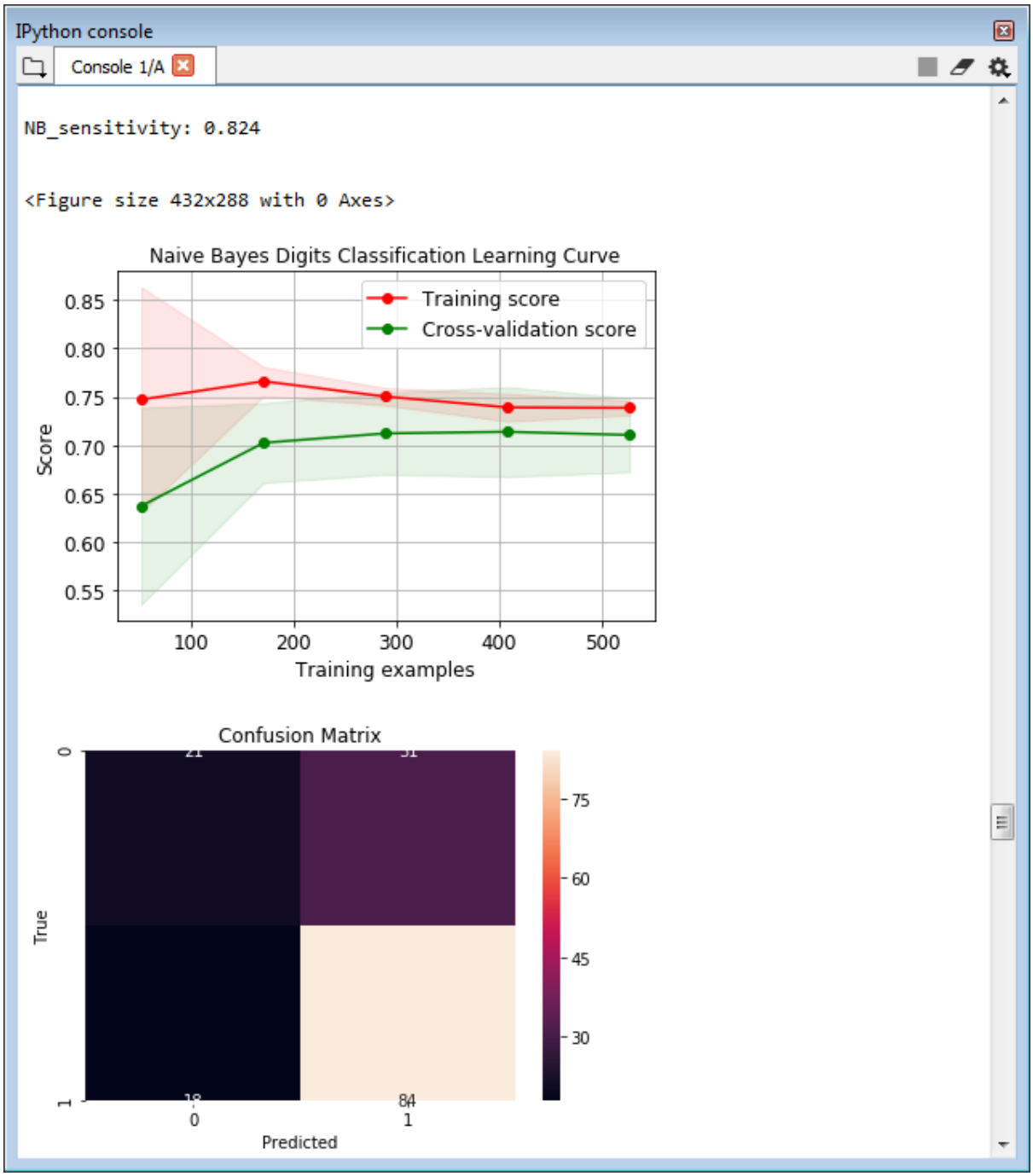


FIGURE.NO 9.2.15 DATASET 2

DATASET 3

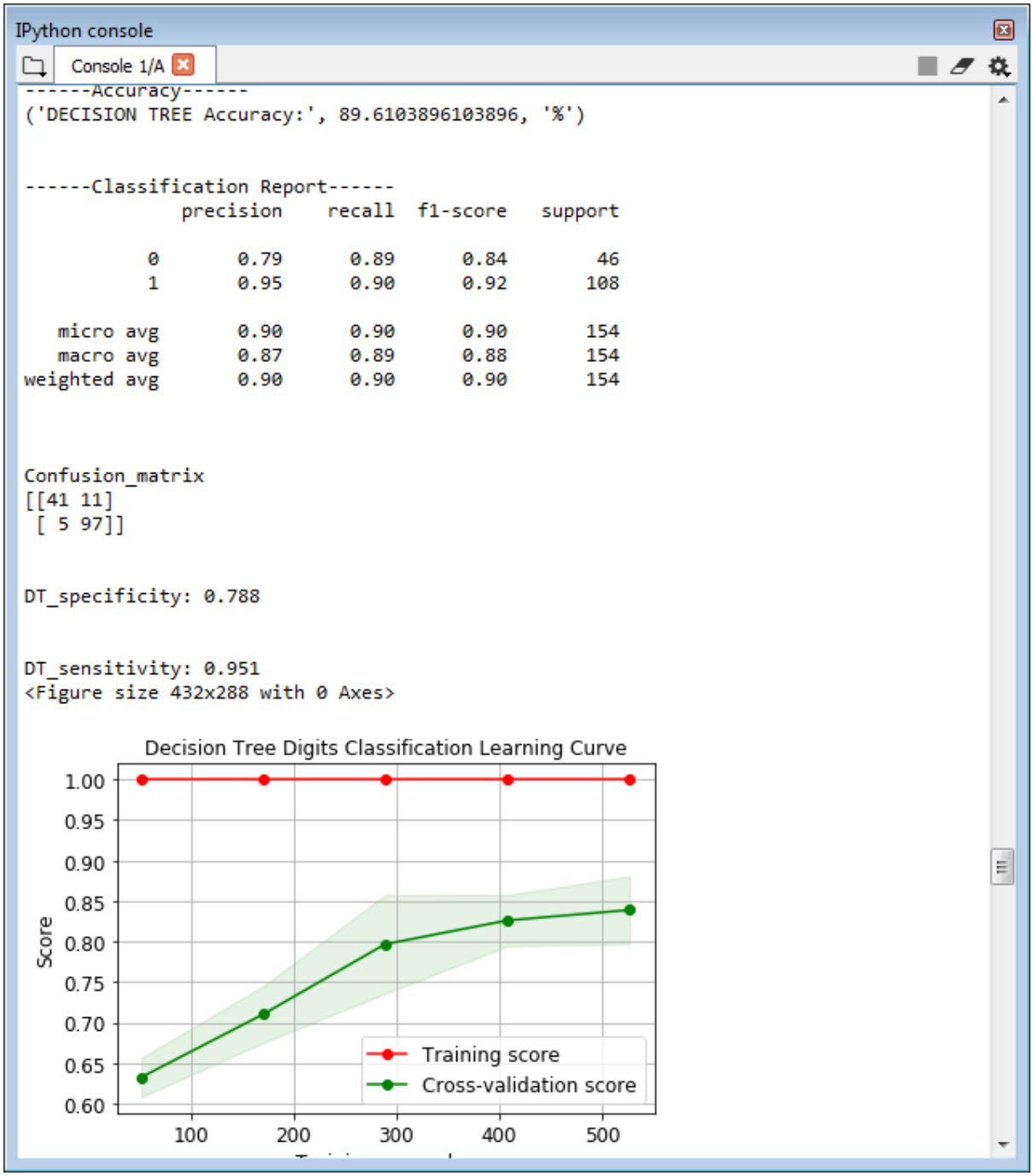


FIGURE.NO 9.2.16 DATASET

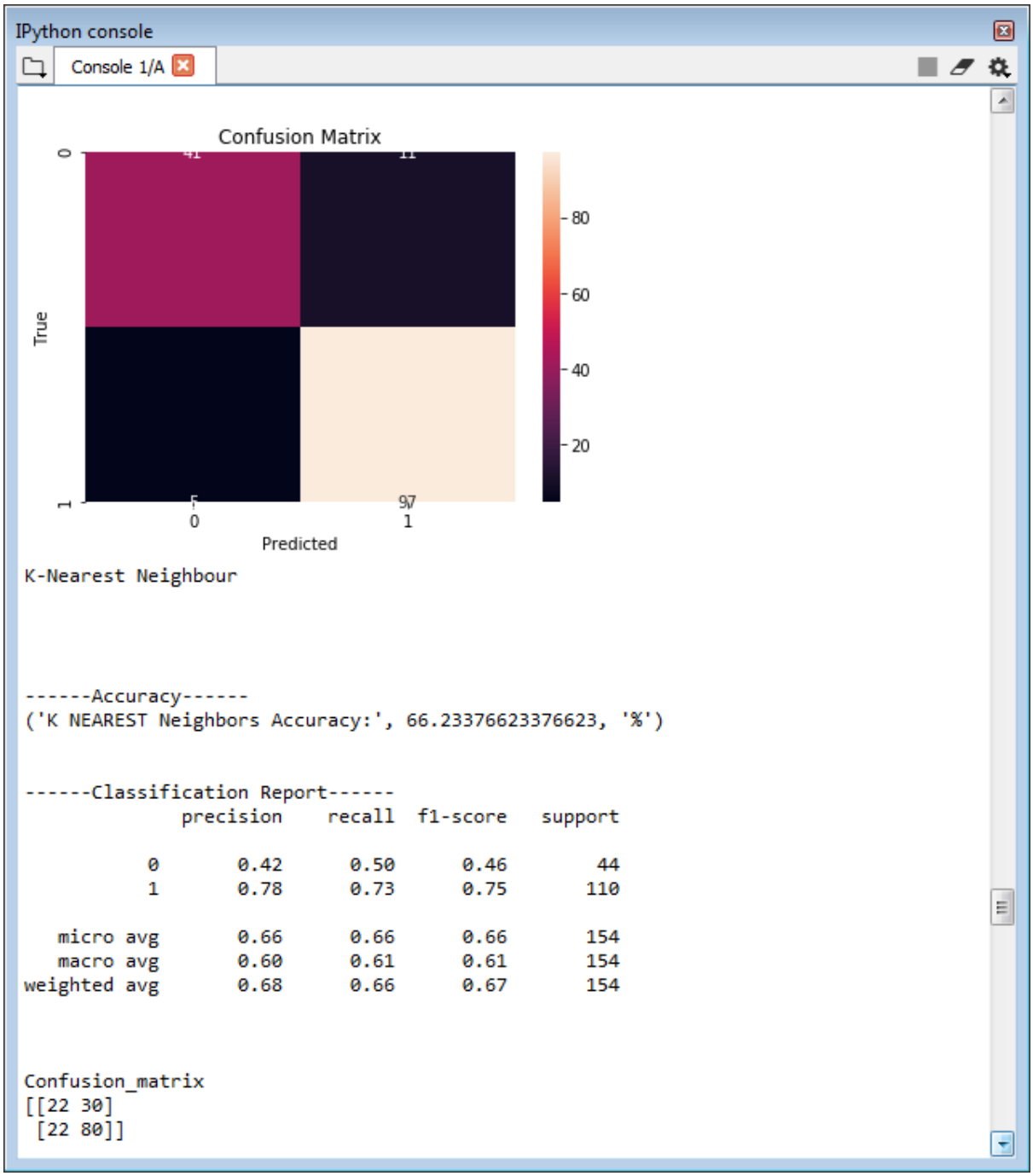


FIGURE.NO 9.2.17 DATASET 3

DATASET 4

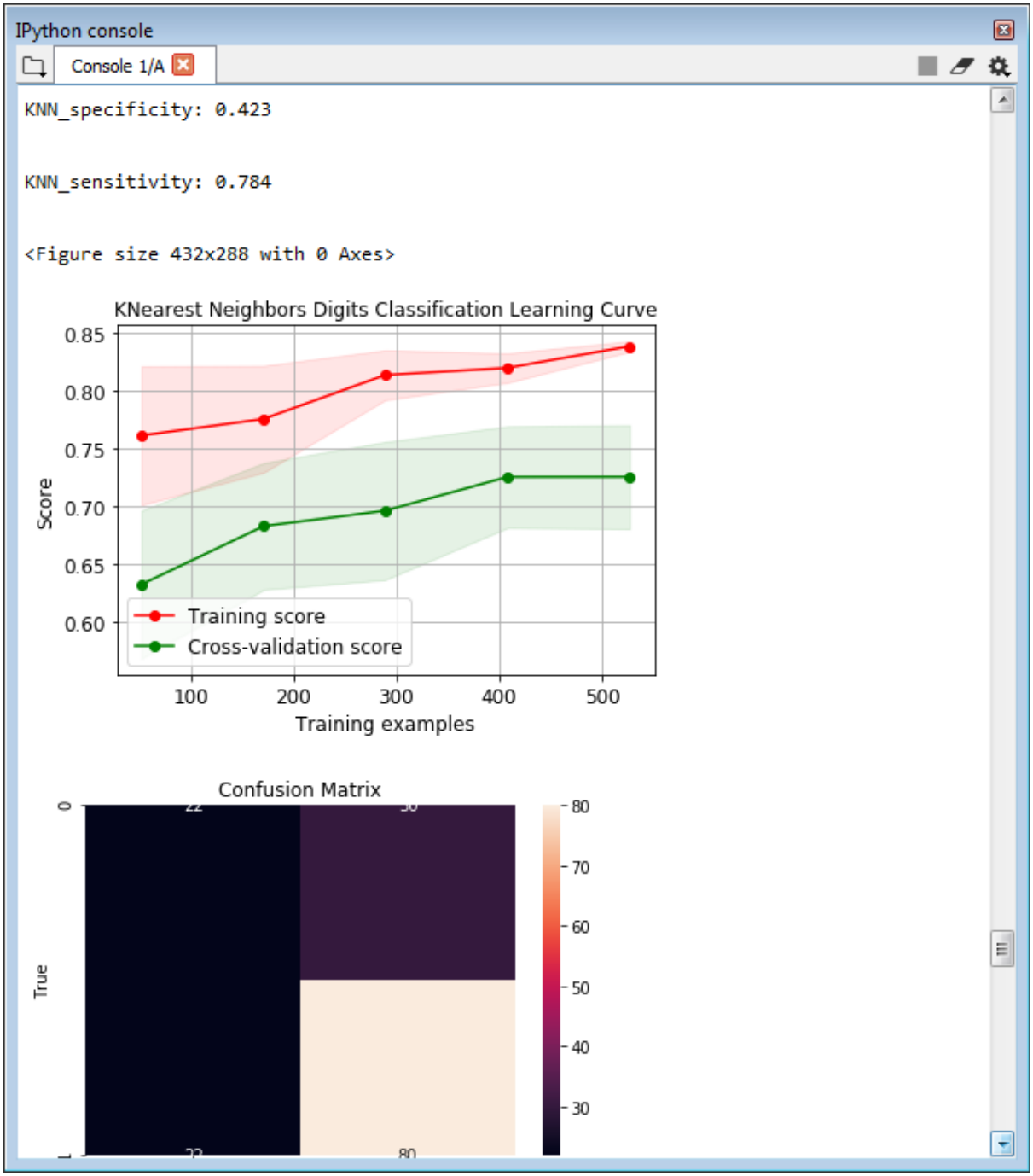


FIGURE.NO 9.2.18 DATASET 4

DATASET 5

IPython console

Console 1/A

```
------Accuracy------
('SVM Accuracy:', 70.77922077922078, '%')


------Classification Report------
              precision    recall  f1-score   support

           0       0.33      0.63      0.43        27
           1       0.90      0.72      0.80       127

   micro avg       0.71      0.71      0.71       154
   macro avg       0.61      0.68      0.62       154
weighted avg       0.80      0.71      0.74       154



Confusion_matrix
[[17 35]
 [10 92]]


SVM_specificity: 0.327


SVM_sensitivity: 0.902
<Figure size 432x288 with 0 Axes>
```

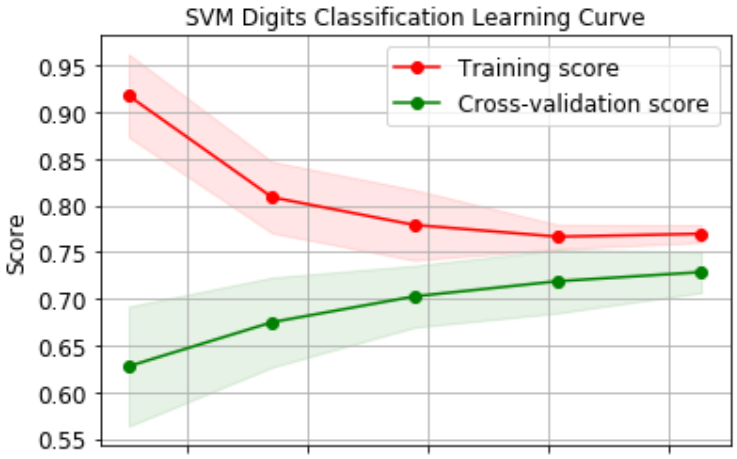


FIGURE.NO 9.2.19 DATASET 5

DATASET 6

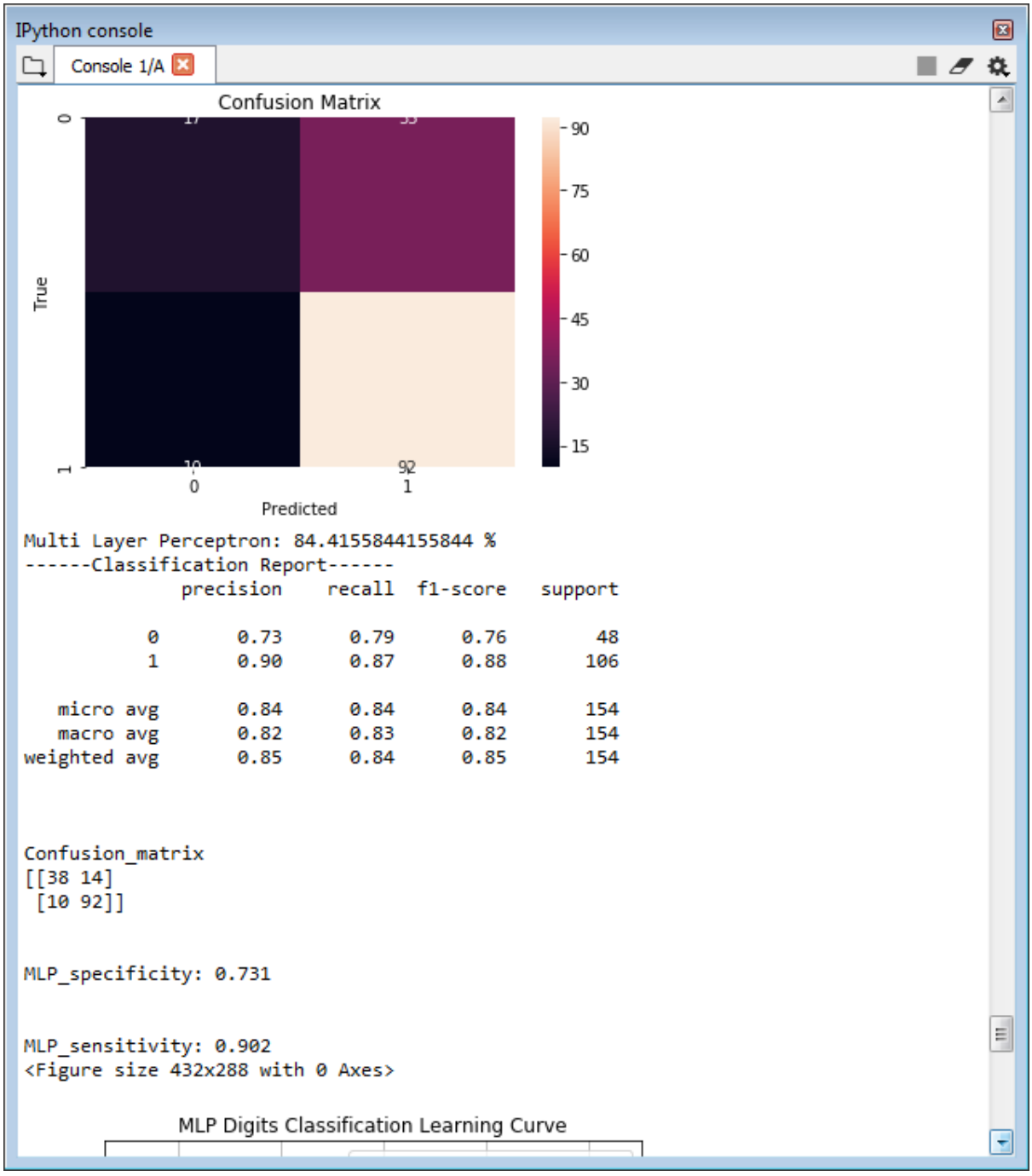


FIGURE.NO 9.2.20 DATASET 6

DATASET 7

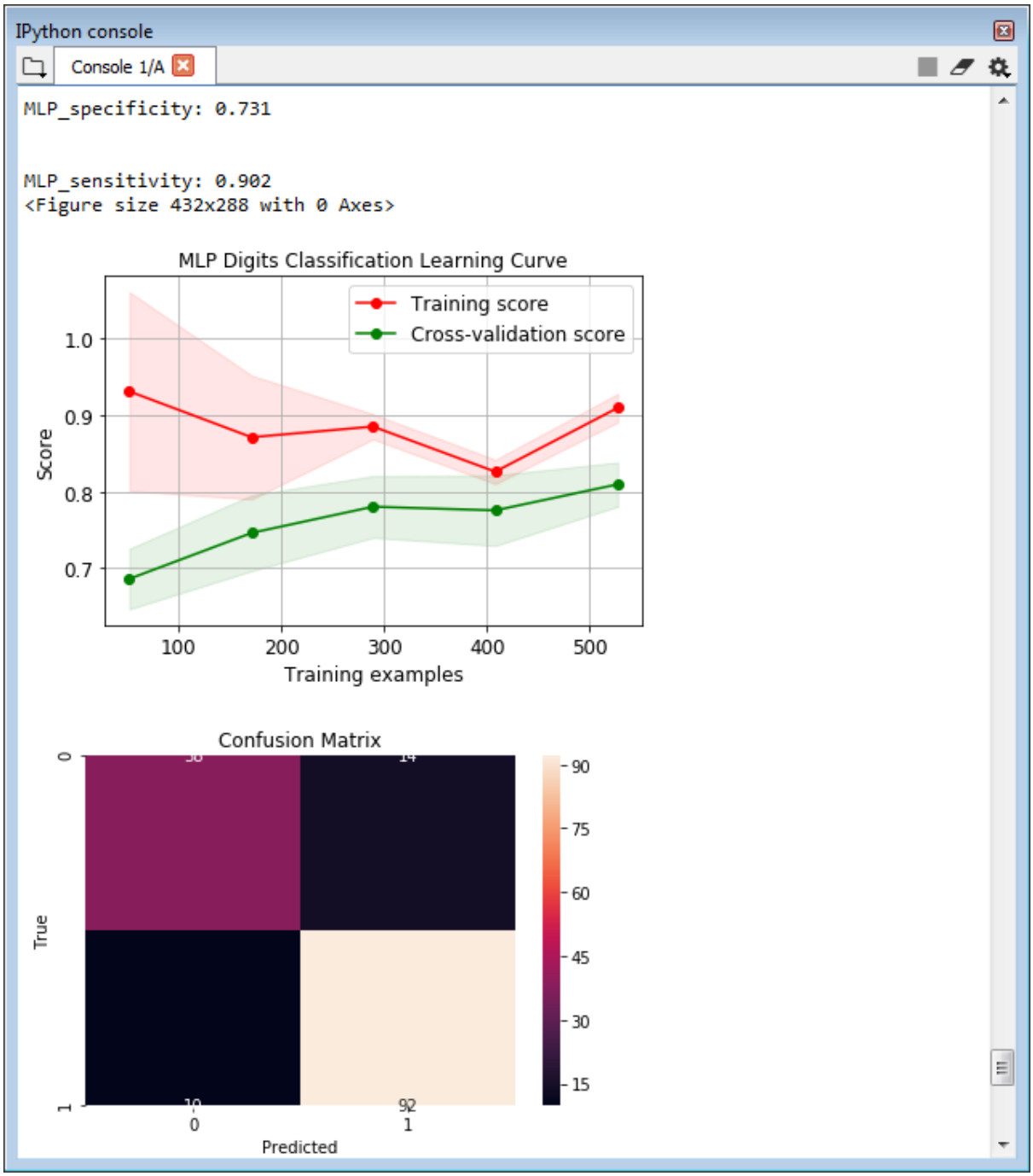


FIGURE.NO 9.2.21 DATASET 7

9.3 GRAPH

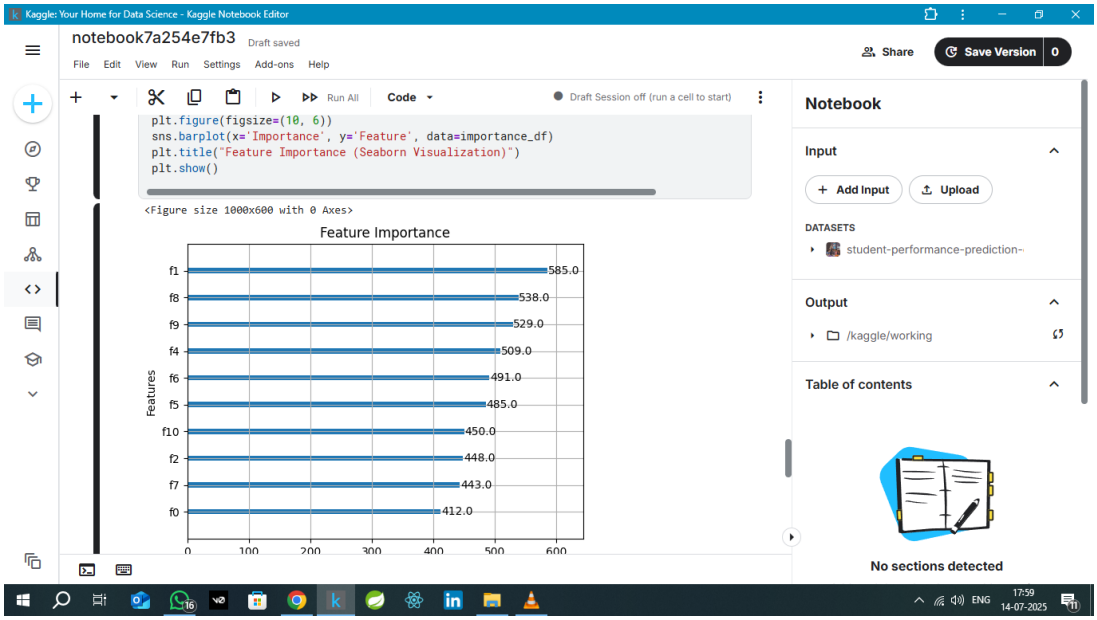


FIGURE.NO 9.3.1 GRAPH

- **Plot Type**: Histogram with KDE (Kernel Density Estimation)
- **Title**: *Distribution of Grade*
- **Purpose**: Shows how student grades are distributed.
- **Observation**:
- Most students scored between **0 and 2**.
- Fewer students got grades higher than 5.

Feature Importance

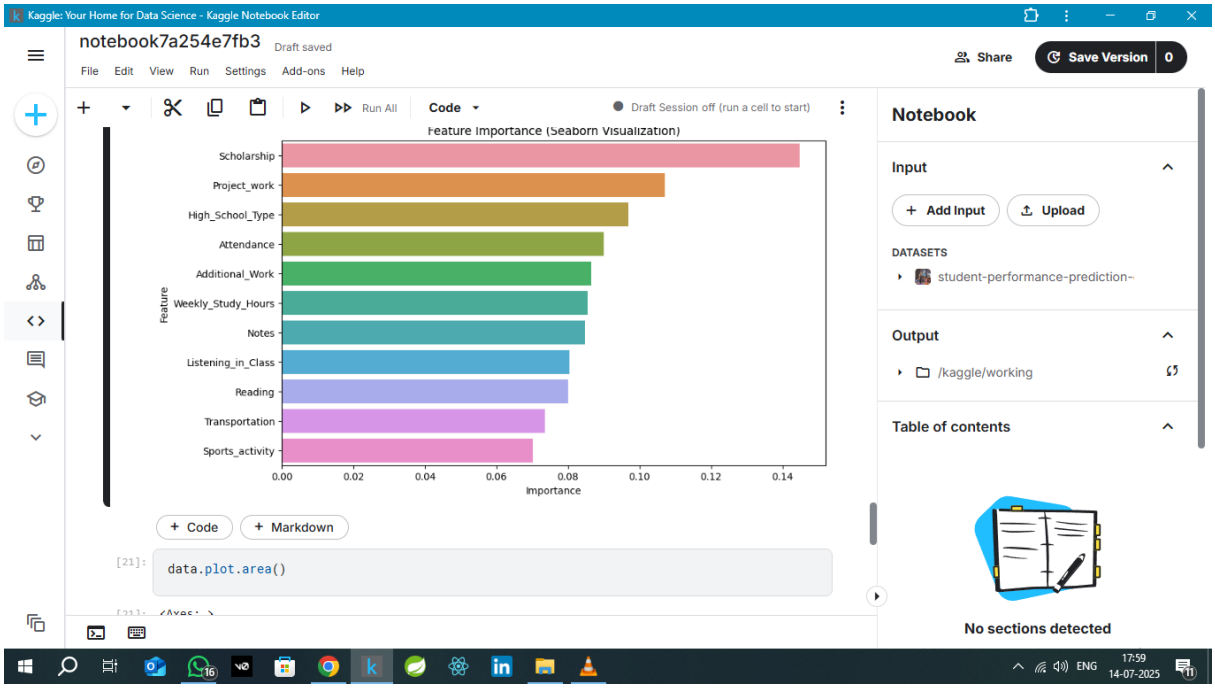


FIGURE.NO 9.3.2 FEATURE IMPORTANCe

- **Plot Type**: Histogram + KDE
- **Title**: *Distribution of High_School_Type*
- **Purpose**: Shows how many students are from each type of high school.
- **Observation**:
- Most students belong to high school type coded as 2.0.
- Fewer students are in types 1.0 and 0.0.

Feature Scaling

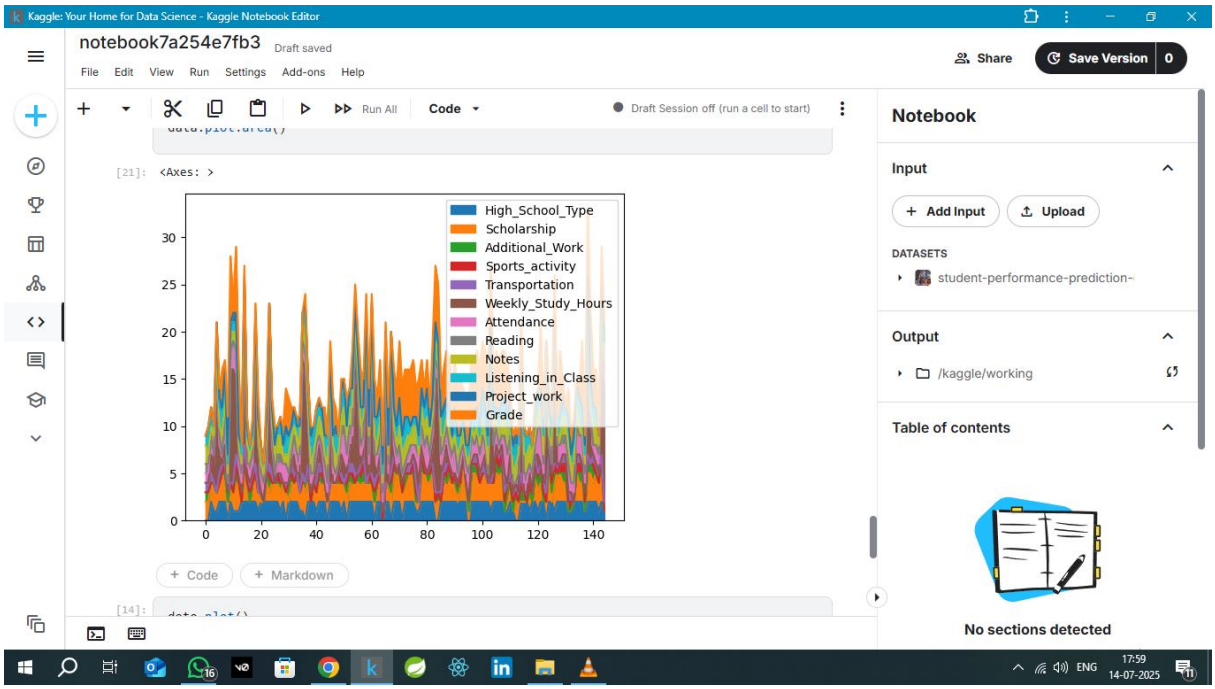


FIGURE.NO 9.3.3 FEATURE SCALING

- **Tool Used**: StandardScaler from sklearn.preprocessing
- **Purpose**: Normalize features for better ML model performance.
- **Explanation**:
  - fit_transform(x_train) is used to scale training data.
  - transform(x_test) is used to scale test data using the same parameters

Feature Importance (Raw)

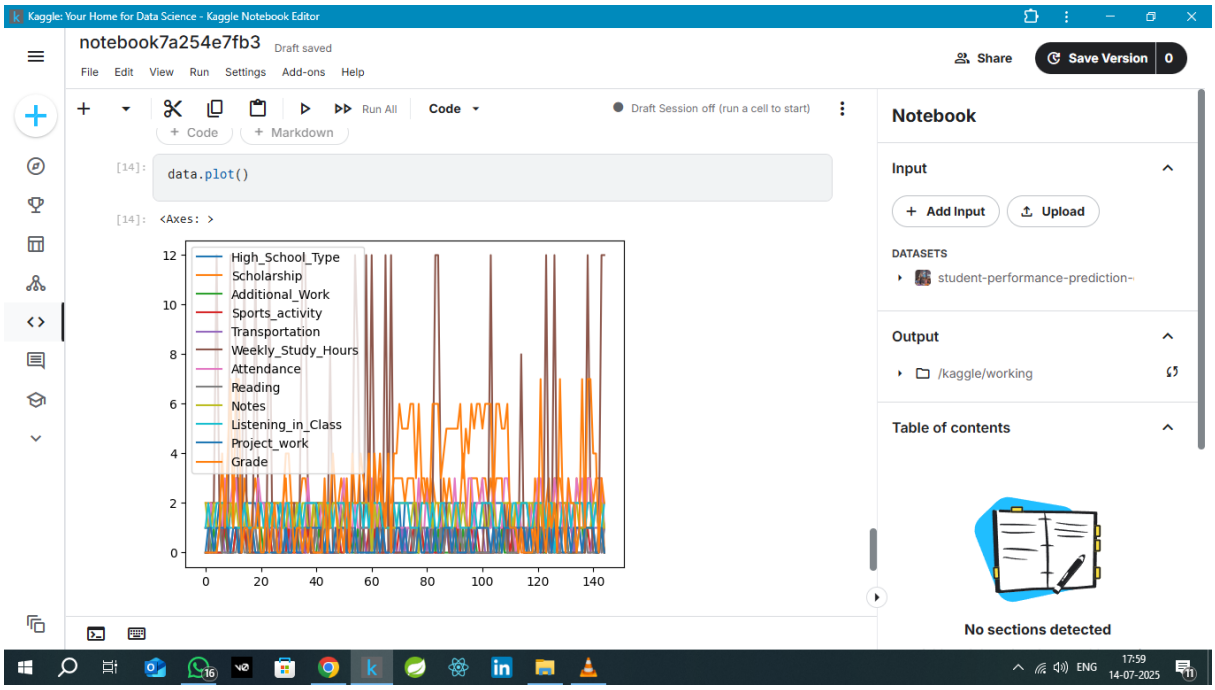


FIGURE.NO 9.3.4 Feature Importance (Raw)

- **Title**: *Feature Importance*
- **Type**: Horizontal bar chart
- **Features**: f0 to f10 (probably encoded features)
- **Observation**:
  - f1, f8, and f9 are the top contributing features to predicting the grade.

Feature Importance

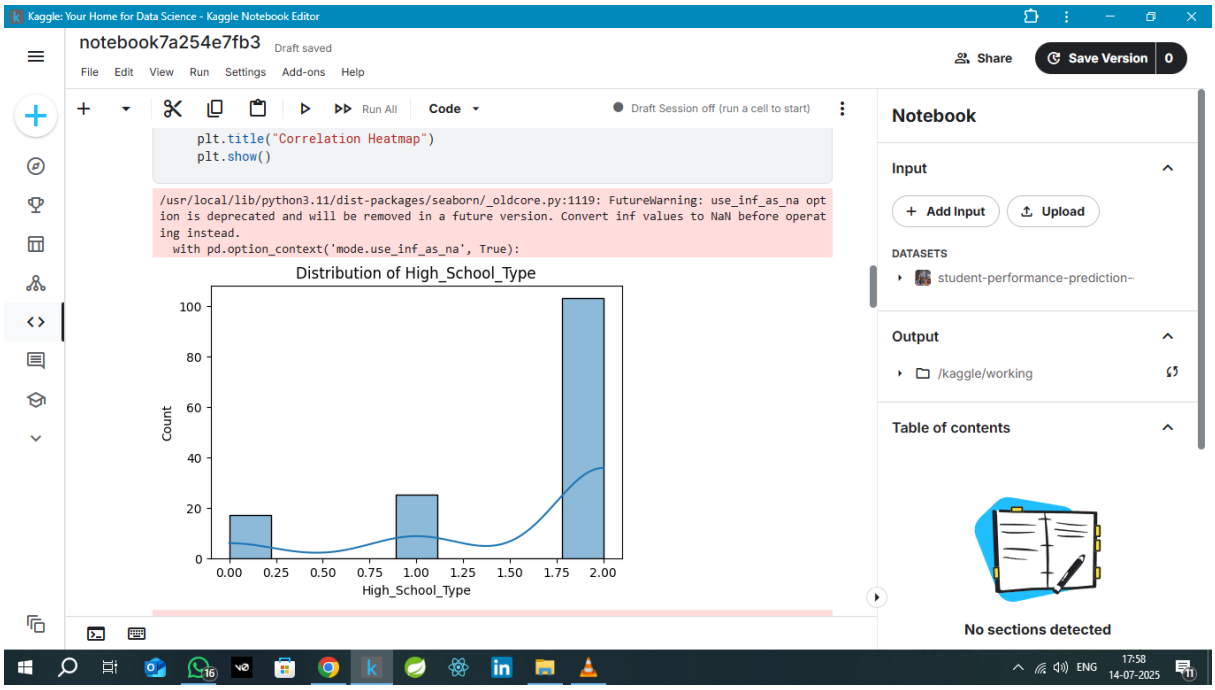


FIGURE.NO 9.3.4 Feature Importance

- **Improved Visualization**:
  - Uses real feature names instead of f0, f1...
- **Top Features**:
  - Scholarship, Project_work, High_School_Type, and Attendance are most important.
- **Use**: Helps identify which student activities contribute most to academic success.

Correlation Heatmap

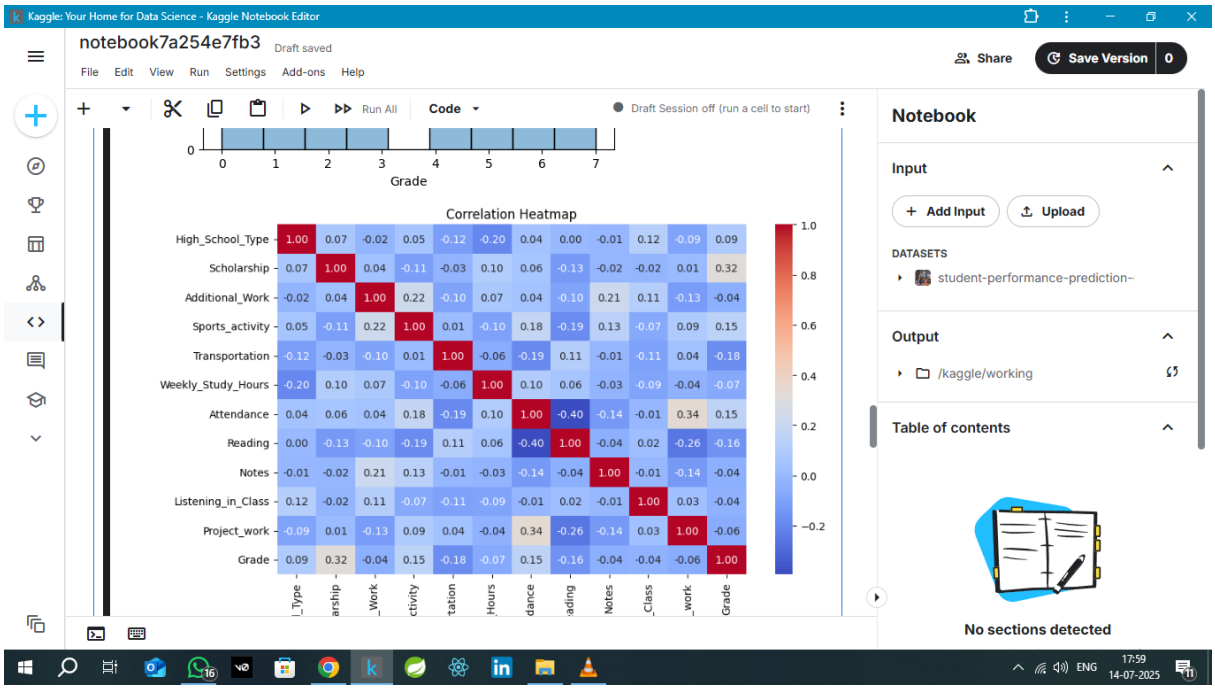


FIGURE.NO 9.3.5 Correlation Heatmap

- **Tool Used**: Seaborn heatmap
- **Purpose**: Show correlation between all features (e.g., Attendance, Reading, Notes) and Grade.
- **Important Insight**:
  - Scholarship (0.32), Attendance (0.15), and Sports_activity (0.15) show relatively higher positive correlation with Grade.
  - Negative or very low correlation for features like Notes and Reading.

CHAPTER X

REFERENCE

- R. L. Ahadi, H. Haapala, and A. Vihavainen, ―Exploring machine learning methods to automatically identify students in need of assistance,‖ in Proc. 11th Annu. Int. Conf. Int. Comput. Educ. Res., 2015, pp. 121–130.
- K. Quille and S. Bergin, ―Programming: Further factors that influence success,‖ in Psychology of Programming Interest Group (PPIG). Cambridge, U.K.: Univ. Cambridge, 2016.
- C. Y. Ko and F. Y. Leu, ―Analyzing attributes of successful learners by using machine learning in an undergraduate computer course,‖ in Proc. 32nd IEEE Int. Conf. Adv. Inf. Netw. Appl. (AINA- 2018), Krakow, Poland, 2018, pp. 801–806.
- S. Kotsiantis and D. Kanellopoulos, ―Association rules mining: A recent overview,‖ Int. Trans. Comput. Sci. Eng., vol. 32, no. 1, pp. 71–82, 2006.
- J.-L. Hung and K. Zhang, ―Revealing online learning behaviors and activity patterns and making predictions with data mining techniques in online teaching,‖ J. Online Learn. Teach., vol. 4, no. 4,
  pp. 426–436, 2008.
- A. Ezen-Can, K. E. Boyer, S. Kellogg, and S. Booth, ―Unsupervised modeling for understanding MOOC discussion forums: A learning analytics approach,‖ in Proc. 5th Int. Conf. Learn. Anal. Knowl., 2015, pp. 146–150.
- J.-L. Hung, M. C. Wang, S. Wang, M. Abdelrasoul, Y. Li, and W. He, ―Identifying at-risk students for early interventions—A time-series clustering approach,‖ IEEE Trans. Emerg. Topics Comput., vol. 5, no. 1, pp. 45–55, Jan.–Mar. 2017.
- C. Romero, M.-I. López, J.-M. Luna, S. Ventura, ―Predicting students‘ final performance from participation in on-line discussion forums,‖ Comput. Educ. vol. 68, pp. 458–472, Oct. 2013.
- R. Asif, A. Merceron, S. A. Ali, and N. G. Haider, ―Analyzing undergraduate students‘ performance using educational data mining,‖ Comput. Educ., vol. 113, pp. 177–194, Oct. 2017.
- S. Amershi and C. Conati, ―Unsupervised and supervised machine learning in user modeling for intelligent learning environments,‖ in Proc. 12th Int. Conf. Intell.

User Interfaces, 2007, pp. 72– 81.

- B. J. Zimmerman, ―Attaining of self-regulation: A social cognitive perspective,‖ in Handbook of Self-Regulation, Research, and Applications, M. Boekaerts, P. Pintrich, and M. Zeidner, Eds. Orlando, FL, USA: Academic, 2000, pp. 13–39.
- B. J. Zimmerman, ―Investigating self-regulation and motivation: Historical background, methodological developments, and future prospects,‖ Amer. Educ. Re
- s. J., vol. 45, no. 1, pp. 166–183, 2008.A. Bandura, Social Learning Theory. Oxford, U.K.: Prentice-Hall, 1977.
- A. Bandura, Self-Efficacy: The Exercise of Control. New York, NY, USA: Freeman, 1997.
- R. Lynch and M. Dembo, ―The relationship between self-regulation and online learning in a blended learning context,‖ Int. Rev. Res. Open Distance Learn., vol. 5, no. 2, pp. 1–16, 2004.
- M. V. J. Veenman, B. H. A. M. Van Hout-Wolters, and P. Afflerbach, ―Metacognition and learning: Conceptual and methodological considerations,‖ Metacogn. Learn., vol. 1, pp. 3–14, Mar. 2006.
- P. R. Pintrich, ―The role of goal orientation in self-regulated learning,‖ in Handbook of Self- Regulation, M. Boekaerts, P. R. Pintrich, and M. Zeidner, Eds. San Diego, CA, USA: Academic, 2000, pp. 451–502.