

Rizvi College of Engineering

Department of Computer Engineering

Machine Learning Mini Project Report

on

Performance-Based Tutoring

Submitted in partial fulfilment of the requirements

of the degree of

Bachelors of Engineering

by

Shaikh Mohd Shoeb – (201P029)

Katheem Kizhar Ahmed – (201P036)

Chaitanya Vijay Parab – (201P022)

Sayed Amman Akhtar – (201P016)

Guide:

Prof. Mohd Ashfaque Shaikh



University of Mumbai

2023 - 2024

**ABSTRACT**

The "Performance-Based Tutoring" initiative is an advanced educational project that harnesses the power of machine learning to comprehensively assess and respond to student performance. Through the meticulous analysis of academic marks, the system identifies specific areas of struggle by establishing predefined thresholds. When a student consistently falls below these benchmarks, the system dynamically generates personalized tutoring recommendations, tailoring interventions to address the precise challenges faced by each student. What sets this project apart is its emphasis on collaboration among students, teachers, and parents. When a student is flagged for additional support, the system initiates a seamless communication loop, involving educators and parents in the process. This collaborative approach ensures that all stakeholders are informed and engaged in addressing the identified academic difficulties. Moreover, the system's adaptability is a key strength, continually monitoring student progress and refining its recommendations to align with evolving performance patterns. By fostering a responsive, dynamic, and collaborative educational environment, the project seeks to enhance academic outcomes by providing targeted support and promoting a holistic approach to student success.

Keywords: machine learning, academic marks, student performance, personalized tutoring recommendations, challenges, collaboration, students, teachers, parents, additional support, academic difficulties, adaptability, monitoring, progress, evolving performance patterns, student success

***Certificate***

This is to certify that the project synopsis entitled “**Performance-Based Tutoring**” has been submitted by **Shaikh Mohd Shoeb, Katheem Kizhar Ahmed, Chaitanya Vijay Parab, Sayed Amman Akhtar** under the guidance of **Prof. Mohd Ashfaque Shaikh** in partial fulfillment of the requirement for the award of the Degree of Bachelor of Engineering in Computer Engineering from Rizvi College of Engineering, University of Mumbai.

Certified By

Prof. \_\_\_\_\_\_\_\_\_\_\_\_ Prof. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Guide Head of Department

Prof. \_\_\_\_\_\_\_\_\_\_\_\_ Prof. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Internal Examiner External Examiner

Dr. Varsha Shah

Principal



Department of Computer Engineering

**Rizvi College of Engineering,**

Off Carter Road, Bandra(W), Mumbai-400050

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

-----------------------------------------

(Signature)

Shaikh Mohd Shoeb - 48

Katheem Kizhar Ahmed -11

Chaitanya Vijay Parab - 30

Sayed Amman Akhtar - 37

Date:

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Title** | **Page No** |
| **1**. | **Introduction** …………..................................................................... |
| **2**. | **Literature Survey**………………………………………………….  **2.1**. …………………………………………………………………  **2.2**. …………………………………………………………………  **2.3.** ……………………………………………………..………….. |
| **3**. | **Proposed System …………………………………………………..**  **3.1. ………………………………………………………………….**  **3.2. ………………………………………………………………….**  **3.3…………………………………………………………………..** |
| 4. | **Implementation Plan ……………………………………………..**  **4.1. ………………………………………………………………….**  **4.2. ………………………………………………………………….**  **4.3…………………………………………………………………..**  **Conclusion**…**………………………………………………………..**  **References**…**………………………………………………………..**  **Apendix – I: Paper Publications** |
|  |

**Chapter 1**

**Introduction**

The "Performance-Based Tutoring" initiative employs advanced machine learning to assess student performance rigorously. It identifies academic struggles by analyzing marks and predefined thresholds, generating tailored tutoring recommendations when students consistently fall below benchmarks. A unique facet is its strong collaboration among students, teachers, and parents, engaging them in addressing academic challenges. The system's adaptability continually refines recommendations as it monitors student progress. Ultimately, this project aims to create a responsive, dynamic, and collaborative educational environment that bolsters academic success by providing targeted support and fostering a holistic approach to student development.

**Chapter 2**

**Review of Literature**

**Paper I**

**Predicting University's Students Performance Based on Machine Learning Techniques**

Machine learning algorithms have been used in many fields, like economics, medicine, etc. Education data mining is one of the areas concerned with exploring patterns of data in an educational environment. One of the most important uses is to predict students' performance to improve the existing educational situation. It can be considered as one of the data mining sciences. The ability to predict in advance in many areas has many benefits. In the case of learning, it enables us to know students' levels in advance and identify students who need special attention. This paper proposes using the algorithm (GBDT) which is a machine learning technology used for regression, classification, and ranking tasks, and is part of the Boosting method family to predict university students' performance in final exams. It compares the proposed system's performance with selected machine learning algorithms (Support vector machine, Logistic Regression, Naive Bayes, Gradient Boosted Trees).

**Paper II**

**Effect of Web-Based Intelligence Tutoring System on Students’ Achievement and Motivation**

This study aims to investigate the effect of Web‐Based Intelligence Tutoring System on Students’ Achievement and Motivation in the computer introduction course. For this purpose, an intelligent tutoring system called Office Master was designed and developed that can be reached on the internet. With this software, subjects are taught to students, are presented in audio, visual, and written form. A specific sequence of subjects was determined in the system. The system includes intelligence features for intelligent tutoring systems. Thanks to these features, when students start the lesson, their preliminary information about the subject is checked, and they can follow the subject from the required unit. In addition, students’ logs are recorded in the system. A quasi‐experimental design with the pretest‐posttest control group was used as a quantitative research design to evaluate the effectiveness of the system. It was observed that the achievement of the students was significantly increased as a result of the application, and it was also concluded that the students were very motivated with the system when the results of the motivation survey were examined. It is recommended that similar systems should be applied in other courses so that the courses taught in universities will be more effective and efficient.

**Chapter 3**

**Proposed System**

**Analysis/Framework/Algorithm:** The proposed system is built on a framework that integrates various components of machine learning, data analysis, and communication systems. It utilizes algorithms for performance assessment, data clustering, and recommendation generation. The key algorithm is a dynamic threshold-based approach that assesses student performance against predefined benchmarks, triggers intervention when necessary, and adapts over time based on evolving performance patterns.

**Details of Hardware & Software:** The hardware infrastructure consists of high-capacity servers and cloud-based resources to handle data storage and processing. The software stack includes machine learning libraries, data analytics tools, and a web-based interface for users, ensuring seamless data management, analysis, and user interaction.

**Design Details:** The system employs a modular design, comprising performance analysis modules, recommendation engines, and communication interfaces. It is designed to be scalable and user-friendly, with an intuitive dashboard for teachers, students, and parents to access and understand student performance data and recommendations.

**Methodology (Approach to Solve the Problem):** The methodology involves a multi-step process:

1. Data Collection: Gathering academic performance data from various sources.
2. Performance Analysis: Using predefined thresholds to assess students' performance.
3. Recommendation Generation: Dynamically creating personalized tutoring recommendations.
4. Communication Loop: Initiating communication among teachers, students, and parents when additional support is required.
5. Continuous Monitoring: Ongoing tracking of student progress to adapt recommendations over time.

**Relevance to PO (Program Outcomes) and PSO (Program Specific Outcomes) of the Department:** The proposed system aligns with the Department's Program Outcomes by promoting data-driven decision-making in education. It fosters collaboration among stakeholders (PO2), utilizes advanced technology (PO7), and enhances student success (PO8). Moreover, it directly supports the Program Specific Outcomes (PSO) by improving the quality of education delivery and addressing individual student needs, ultimately contributing to the Department's mission of academic excellence and student development.

**Chapter 4**

**Implementation/code**

**TRAINING MODEL**

import numpy as np

import pandas as pd

import tensorflow as tf

from tensorflow import keras

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

from sklearn.preprocessing import StandardScaler

# Load your dataset (you should replace 'your\_dataset.csv' with the actual dataset file path)

data = pd.read\_csv('/content/Dataset\_ML\_Project - Sheet1.csv')

min = data.count().min()

min

# data.dropna().sample(min)

data.dropna(inplace=True)

data.count()

# input\_features = ['maths', 'science', 'english']

input\_features = ['total']

output\_label = 'predict'

data\_x = data[input\_features]

data\_y = data[output\_label]

# Normalize the input data (scaling to have mean=0 and variance=1)

scaler = StandardScaler()

data\_x = scaler.fit\_transform(data\_x)

# Split the data into training and testing sets

x\_train, x\_test, y\_train, y\_test = train\_test\_split(data\_x, data\_y, test\_size=0.2, random\_state=42)

len(data\_y)

# Define the neural network model

model = keras.Sequential([

keras.layers.Dense(64, activation='relu', input\_shape=(1,)), # Input layer

keras.layers.Dense(32, activation='relu'), # Hidden layer

keras.layers.Dense(3, activation='softmax') # Output layer (3 classes for 'predict')

])

data\_y.unique()

print(model.input\_shape)

print(X\_train.shape)

# Compile the model

model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

# Train the model

history = model.fit(x\_train, y\_train, epochs=100, batch\_size=32, verbose=2)

from sklearn.metrics import confusion\_matrix, classification\_report, accuracy\_score

import matplotlib.pyplot as plt

# Generate predictions

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

y\_pred\_classes = np.argmax(y\_pred, axis=-1)

# Generate a confusion matrix

confusion = confusion\_matrix(y\_test, y\_pred\_classes)

# Generate a classification report

class\_report = classification\_report(y\_test, y\_pred\_classes)

# Calculate the accuracy score

accuracy = accuracy\_score(y\_test, y\_pred\_classes)

# Print confusion matrix, classification report, and accuracy

print("Confusion Matrix:\n", confusion)

print("\nClassification Report:\n", class\_report)

print("\nAccuracy Score:", accuracy)

# Plot training loss and accuracy over epochs

plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)

plt.plot(history.history['loss'], label='Training Loss')

plt.xlabel('Epochs')

plt.ylabel('Loss')

plt.legend()

plt.subplot(1, 2, 2)

plt.plot(history.history['accuracy'], label='Training Accuracy')

plt.xlabel('Epochs')

plt.ylabel('Accuracy')

plt.legend()

plt.show()

# Machine Learning Algorithm

import pandas as pd

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Load your CSV file into a DataFrame

data = pd.read\_csv('/content/Dataset\_ML\_Project - Sheet1.csv')

data.dropna(inplace=True)

# Define input features and output label

input\_features = ['maths', 'science', 'english']

output\_label = 'predict'

X = data[input\_features]

y = data[output\_label]

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=20)

# Create and train a Random Forest classifier

clf = RandomForestClassifier(n\_estimators=100, random\_state=42)

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

# Make predictions

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

# Evaluate the model

accuracy = accuracy\_score(y\_test, y\_pred)

confusion = confusion\_matrix(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred

# Print evaluation results

print("Accuracy:", accuracy)

print("\nConfusion Matrix:\n", confusion)

print("\nClassification Report:\n", report)

from sklearn.tree import DecisionTreeClassifier

# Create and train a Random Forest classifier

clf = DecisionTreeClassifier(max\_depth=5, random\_state=42)

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

# Make predictions

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

# Evaluate the model

accuracy = accuracy\_score(y\_test, y\_pred)

confusion = confusion\_matrix(y\_test, y\_pred)

report = classification\_report(y\_test, y\_pred)

# Print evaluation results

print("Accuracy:", accuracy)

print("\nConfusion Matrix:\n", confusion)

print("\nClassification Report:\n", report)

import seaborn as sns

# Calculate the correlation matrix

corr\_matrix = data.corr()

# Create a heatmap

plt.figure(figsize=(10, 8))

sns.heatmap(corr\_matrix, annot=True, cmap='coolwarm', linewidths=0.5)

plt.title('Correlation Heatmap')

plt.show()

# import joblib

import pandas as pd

# Collect the input data for testing

new\_data = pd.DataFrame({'maths': [85, 90, 78], 'science': [92, 88, 75], 'english': [88, 80, 85]})

# Use the trained Random Forest classifier to make predictions

new\_predictions = clf.predict(new\_data)

print(prediction)

import pandas as pd

# Collect the input data for testing

new\_data = pd.DataFrame({'maths': [85, 90, 78], 'science': [92, 88, 75], 'english': [88, 80, 85]})

# Use the trained Random Forest classifier to make predictions

new\_predictions = clf.predict(new\_data)

# Interpret the predictions

for i, prediction in enumerate(new\_predictions):

if prediction == 1:

print(f"Data point {i+1}: The model predicts 'Yes'")

else:

print(f"Data point {i+1}: The model predicts 'No'")

import joblib

# Save the trained model to a file

model\_filename = 'random\_forest\_model.pkl'

joblib.dump(clf, model\_filename)

import pandas as pd

import joblib

# Load the saved model

loaded\_model = joblib.load('random\_forest\_model.pkl')

# Collect the input data for testing

new\_data = pd.DataFrame({'maths': [0, 90, 78], 'science': [0, 88, 75], 'english': [0, 80, 85]})

# Use the loaded model to make predictions

new\_predictions = loaded\_model.predict(new\_data)

# Interpret the predictions

for i, prediction in enumerate(new\_predictions):

if prediction == 1:

print(f"Data point {i+1}: The model predicts 'Yes'")

else:

print(f"Data point {i+1}: The model predicts 'No'")

print(new\_predictions)

**GUI**

import tkinter as tk

from tkinter import ttk

import pandas as pd

import joblib

# Load the saved model

import pickle

with open('random\_forest\_model.pkl', 'rb') as model\_file:

loaded\_model = pickle.load(model\_file)

# Define a function to make predictions

def predict():

math\_score = float(math\_entry.get())

science\_score = float(science\_entry.get())

english\_score = float(english\_entry.get())

# Create a DataFrame for the input data

input\_data = pd.DataFrame({'maths': [math\_score], 'science': [science\_score], 'english': [english\_score]})

final = ((math\_score + science\_score + english\_score) / 300) \* 100

# Convert the result to an integer to remove decimal places

final = int(final)

# Use the loaded model to make predictions

prediction = loaded\_model.predict(input\_data

if prediction == 0:

result\_label.config(text="Tutoring Required")

if prediction == 1:

result\_label.config(text="Average")

else:

result\_label.config(text="No Tutoring Required"

# Create the main application window

root = tk.Tk()

root.title("Model Prediction UI")

# Create input fields

math\_label = ttk.Label(root, text="Math Score:")

math\_label.pack()

math\_entry = ttk.Entry(root)

math\_entry.pack()

science\_label = ttk.Label(root, text="Science Score:")

science\_label.pack()

science\_entry = ttk.Entry(root)

science\_entry.pack()

english\_label = ttk.Label(root, text="English Score:")

english\_label.pack()

english\_entry = ttk.Entry(root)

english\_entry.pack()

# Create a prediction button

predict\_button = ttk.Button(root, text="Predict", command=predict)

predict\_button.pack()

# Create a label to display the prediction result

result\_label = ttk.Label(root, text="")

result\_label.pack()

# Start the Tkinter main loop

root.mainloop()

**OUTPUT:**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Chapter 5**

**Conclusions**

The "Performance-Based Tutoring" initiative represents a forward-thinking approach to education, harnessing the power of machine learning to comprehensively assess and support student performance. Through dynamic analysis and personalized recommendations, it addresses the unique challenges faced by students. The emphasis on collaboration among students, teachers, and parents ensures a holistic approach to education, promoting a supportive and engaging environment.

With its adaptability and continuous monitoring, this system is poised to make a lasting impact on academic outcomes. It not only aligns with the core values of our department but also serves as a beacon for data-driven, student-centric education. As we move forward, the "Performance-Based Tutoring" initiative has the potential to transform educational practices and empower students to reach their full potential.

Top of Form

**Chapter 6**

**References**

1. Romero, C.; Ventura, S.; Pechenizkiy, M.; Baker, R.S. Handbook of Educational Data Mining; CRC Press: Boca Raton, FL, USA, 2010.
2. Hernández-Blanco, A.; Herrera-Flores, B.; Tomás, D.; Navarro-Colorado, B. A systematic review of deep learning approaches to educational data mining. Complexity 2019, 2019, 1306039. [CrossRef]
3. Bengio, Y.; Lecun, Y.; Hinton, G. Deep Learning for AI. Commun. ACM 2021, 64, 58–65. [CrossRef]
4. Lykourentzou, I.; Giannoukos, I.; Mpardis, G.; Nikolopoulos, V.; Loumos, V. Early and dynamic student achievement prediction in e-learning courses using neural networks. J. Am. Soc. Inf. Sci. Technol. 2009, 60, 372–380. [CrossRef]
5. Kuzilek, J.; Hlosta, M.; Herrmannova, D.; Zdrahal, Z.; Wolff, A. OU Analyse: Analysing at-risk students at The Open University. Learn. Anal. Rev. 2015, 2015, 1–16.
6. He, J.; Bailey, J.; Rubinstein, B.I.; Zhang, R. Identifying at-risk students in massive open online courses. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015.
7. Kloft, M.; Stiehler, F.; Zheng, Z.; Pinkwart, N. Predicting MOOC dropout over weeks using machine learning methods. In Proceedings of the EMNLP 2014 Workshop on Analysis of Large Scale Social Interaction in MOOCs; Department of Computer Science, Humboldt University of Berlin: Berlin, Germany, 2014; pp. 60–65.
8. Alapont, J.; Bella-Sanjuán, A.; Ferri, C.; Hernández-Orallo, J.; Llopis-Llopis, J.; Ramírez-Quintana, M. Specialised tools for automating data mining for hospital management. In Proceedings of the First East European Conference on Health Care Modelling and Computation, Craiova, Romania, 31 August–2 September 2005; pp. 7–19.
9. Hellas, A.; Ihantola, P.; Petersen, A.; Ajanovski, V.V.; Gutica, M.; Hynninen, T.; Knutas, A.; Leinonen, J.; Messom, C.; Liao, S.N. Predicting academic performance: A systematic literature review. In Proceedings of the Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, Larnaca, Cyprus, 2–4 July 2018; pp. 175–199.
10. Alyahyan, E.; Dü¸stegör, D. Predicting academic success in higher education: Literature review and best practices. Int. J. Educ. Technol. High. Educ. 2020, 17, 1–21. [CrossRef]

**Acknowledgements**

I am profoundly grateful to Prof. MOHD ASHFAQUE SHAIKH for his expert guidance and continuous encouragement throughout to see that this project rights its target.

I would like to express deepest appreciation towards Dr. Varsha Shah, Principal RCOE, Mumbai and Prof. ANUPAM CHOUDHARY HOD of COMPUTER Department whose invaluable guidance supported me in this project.

At last I must express my sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped us directly or indirectly during this course of work.

Shaikh Mohd Shoeb - 48

Katheem Kizhar Ahmed -11

Chaitanya Vijay Parab - 30

Sayed Amman Akhtar - 37