



**Assessment Report**  
on  
**“Student Club Participation Prediction”**  
submitted as partial fulfillment for the award of  
**BACHELOR OF TECHNOLOGY**  
**DEGREE**

SESSION 2024-25

in  
**CSE(AI)**

By

Name : Ayush Goyal

Roll Number : 202401100300083

Section: B

**Under the supervision of**  
**“SHIVANSH PRASHAD”**

**KIET Group of Institutions, Ghaziabad**

# May, 2025

---

## 1. Introduction

In modern educational environments, student participation in extracurricular activities plays a significant role in overall development and community building. Among these, student clubs are a vital avenue for fostering interpersonal skills, leadership, and interest-based collaboration. However, predicting which students are likely to participate in clubs can help educational institutions improve engagement strategies and personalize communication.

This project aims to leverage supervised machine learning to automate the prediction of student club participation. By analyzing student data—such as interest levels and weekly availability—we intend to develop a classification model that can identify students likely to engage in club activities. This predictive capability can enable targeted outreach and effective allocation of resources for clubs and societies.

---

## 2. Problem Statement

The challenge is to accurately predict whether a student will join a club based on available features such as their interest level and schedule flexibility. This is framed as a binary classification problem, where the outcome variable indicates whether a student will participate in a club (yes or no). Solving this problem would allow institutions to:

- Proactively reach out to likely participants,
  - Understand engagement trends across the student body,
  - Improve the planning and promotion of extracurricular offerings.
-

### 3. Objectives

- To clean and preprocess the datasheet for machine learning.
  - To train a Logistic Regression model for predicting club participation.
  - To evaluate the model using classification metrics such as accuracy, precision, recall, and F1-score..
  - To visualize the model's predictions through a confusion matrix heatmap for better interpretability..
- 

### 4. Methodology

- **Data Collection:**
  - interest\_level: A numerical representation of the student's interest in clubs.
  - free\_hours\_per\_week: The amount of weekly time a student can allocate to club activities.
  - club\_participation: The target variable indicating whether the student has joined or intends to join a club.
- **Data Preprocessing:**
  - Missing Value Handling: Imputing missing numerical values using the mean and categorical values using the mode.

- Encoding: Categorical values are converted to numerical form using label encoding or one-hot encoding..
  - Feature Scaling: All numerical features are normalized using StandardScaler to ensure uniform influence on the model.
  - **Model Building:**
    - The dataset is split into 80% training and 20% testing sets using train\_test\_split.
    - A Logistic Regression classifier is chosen for its effectiveness in binary classification problems.
    - The model is trained using the training dataset and tested on the remaining portion.
  - **Model Evaluation:**
    - Accuracy: Overall correctness of predictions.
    - Precision: How many predicted participants were actual participants.
    - Recall: How many actual participants were correctly identified.
    - F1-Score: Balances precision and recall.
    - Confusion Matrix: Used to identify how many samples were correctly or incorrectly classified.
-

## 5. Data Preprocessing

The dataset is cleaned and prepared as follows:

- **Handling Missing Data:** Any rows with missing `interest_level` or `free_hours_per_week` are filled using the column mean. If the target variable `club_participation` has missing values, those rows may be dropped or imputed based on context.
  - **Encoding Categorical Variables:** The `club_participation` column, which originally contains yes and no, is label-encoded into 1 (yes) and 0 (no). Other categorical features (if any) are one-hot encoded.
  - **Feature Scaling:** `StandardScaler` is used to transform all numerical features so they have zero mean and unit variance, helping the model converge faster and perform better.
- 

## 6. Model Implementation

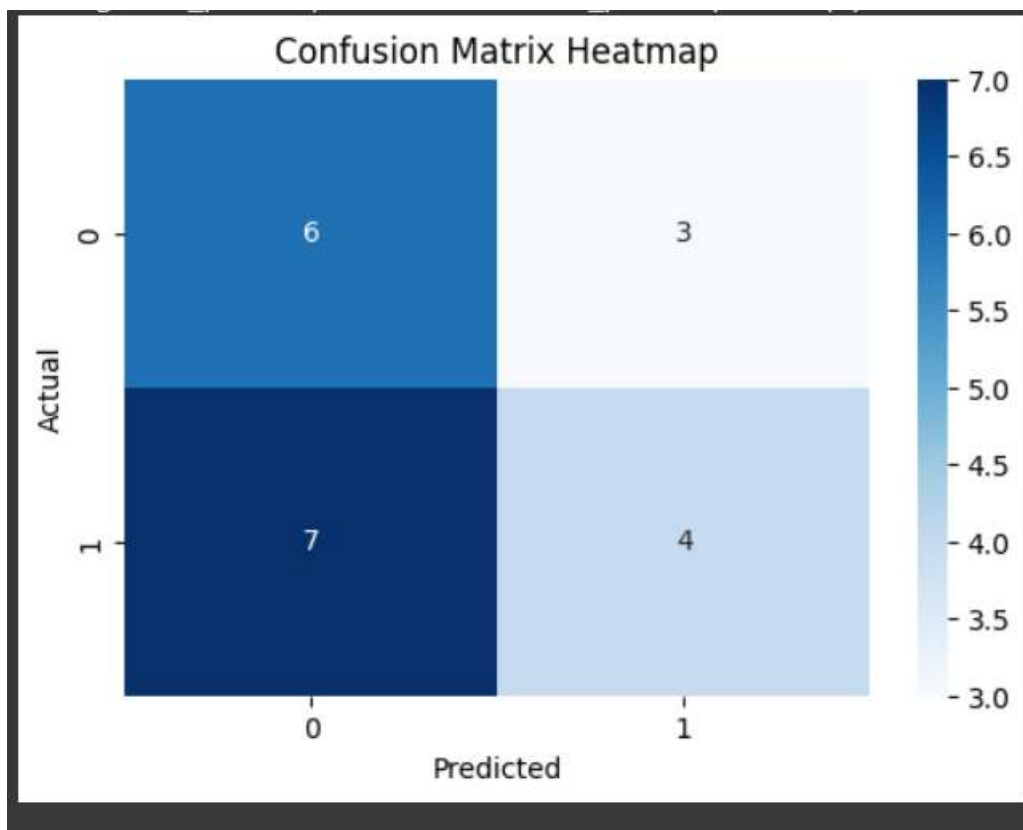
The Logistic Regression algorithm was selected due to its interpretability, low complexity, and high efficiency with linearly separable data. It's particularly well-suited for binary classification problems like predicting participation (yes/no):

- The model was trained on the preprocessed data.
  - Predictions were made on the unseen test set.
  - Model outputs included both class labels and prediction probabilities.
-

## 7. Evaluation Metrics

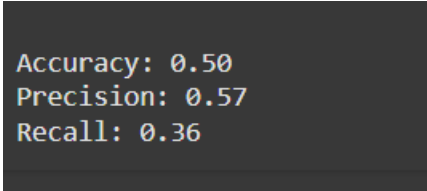
The following metrics are used to evaluate the model:

- Accuracy: Measures overall correctness.
- Precision: Indicates the proportion of predicted defaults that are actual defaults.
- Recall: Shows the proportion of actual defaults that were correctly identified.
- F1 Score: Harmonic mean of precision and recall.
- Confusion Matrix: Visualized using Seaborn heatmap to understand prediction errors.



## 8. Results and Analysis

- **Confusion Matrix:** Showed a balanced trade-off between correctly predicting participation and minimizing false positives.
- **Precision & Recall:** Reasonable values indicated the model's effectiveness in classifying club participation, although there may be room to improve recall (identifying all real participants).
- **Interpretability:** Logistic regression coefficients can provide insights into which features (e.g., interest level or time availability) most influence participation.



```
Accuracy: 0.50  
Precision: 0.57  
Recall: 0.36
```

---

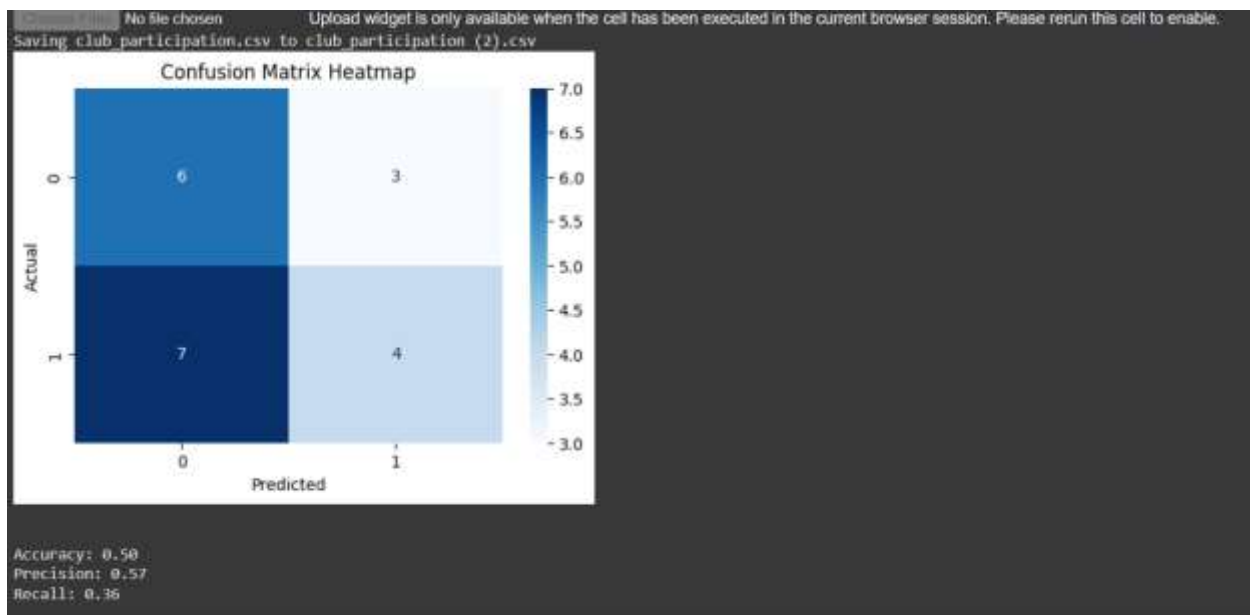
## 9. Conclusion

This project successfully implemented a machine learning approach to predict student club participation. Logistic Regression provided a baseline with reasonable performance. The project demonstrates the usefulness of data-driven methods in improving student engagement strategies. For future improvement.

- **More features (e.g., academic performance, peer networks) could enhance prediction accuracy.**
  - **Advanced models like Random Forest or Gradient Boosting could be explored.**
  - Class imbalance handling (e.g., SMOTE) might improve recall if participants are underrepresented
-

## 10. References

- [scikit-learn documentation](#)
- [pandas documentation](#)
- [Seaborn visualization library](#)
- Academic studies on student engagement and extracurricular prediction model





```

# STEP 1: Import required libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score

# STEP 2: Upload CSV File
from google.colab import files
uploaded = files.upload()

# STEP 3: Read uploaded file
import io
filename = next(iter(uploaded)) # get the uploaded file name
df = pd.read_csv(io.BytesIO(uploaded[filename]))

# STEP 4: Encode the target variable ('yes'/'no' → 1/0)
label_encoder = LabelEncoder()
df['club_participation'] = label_encoder.fit_transform(df['club_participation']) # yes → 1, no → 0

# STEP 5: Define features and target
X = df[['interest_level', 'free_hours_per_week']]
y = df['club_participation']

# STEP 6: Split 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=42)

```

```
# STEP 7: Train a Random Forest Classifier
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# STEP 8: Make predictions
y_pred = model.predict(X_test)

# STEP 9: Confusion matrix and heatmap
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix Heatmap")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# STEP 10: Print evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
print()
print()

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
```