Introduction to AI: Predicting Club Participation Using Machine Learning

Title Page

Title: Predicting Club Participation Using Machine Learning
Problem Statement: The goal is to develop a machine learning model that predicts a student's club participation based on various attributes.
Name: Harsh Kumar Mishra
Roll No.: 202401100300114
Date: 22-04-2025

Introduction

In educational institutions, understanding student participation in clubs can help enhance extracurricular planning and student engagement. This project uses machine learning to predict a student's club participation based on various input features such as student's interest and number of hours he get free in a week . A Random Forest Classifier is used for classification.

Methodology

1. Dataset Acquisition: The dataset was downloaded from Google Drive using the gdown library.
2. Preprocessing:
   - Column names were cleaned.
   - Categorical columns were encoded using LabelEncoder.
3. Modeling:
   - The target variable is club_participation.
   - Data was split into 80% training and 20% testing.
   - A RandomForestClassifier was trained on the training set.
4. Evaluation: Model performance was assessed using accuracy score on the test data.
5. Prediction: The model takes user input to predict the likely club a student would join.

Code

```
# Importing necessary libraries

import pandas as pd

import gdown

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score


# URL of the dataset to download

url = "https://drive.google.com/uc?id=1d_stwG7PnM509FwEjDcFcvZml4RdYzYN"

output = "club_participation.csv"

# Download the dataset

gdown.download(url, output, quiet=False)


# Load the dataset into a DataFrame

df = pd.read_csv("club_participation.csv")


# Strip any leading/trailing spaces from the column names

df.columns = df.columns.str.strip()


# Print the available column names in the dataset

print("Available columns:", df.columns.tolist())
```

```
# Define the target column for prediction
target_column = 'club_participation'

# Check if the target column exists in the dataset, raise an error if not
if target_column not in df.columns:
    raise ValueError(f"Target column '{target_column}' not found in dataset.")

# Create a dictionary to store label encoders for categorical columns
label_encoders = {}
for column in df.columns:
    # Check if the column has categorical data (object type)
    if df[column].dtype == 'object':
        # Initialize LabelEncoder for this column
        le = LabelEncoder()
        # Convert the categorical values to numeric values
        df[column] = le.fit_transform(df[column])
        # Store the encoder in the dictionary
        label_encoders[column] = le

# Separate the features (X) and target variable (y)
X = df.drop(target_column, axis=1)
y = df[target_column]

# Split the dataset into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Initialize and train a RandomForestClassifier model
model = RandomForestClassifier()
model.fit(X_train, y_train)

# Make predictions using the trained model on the test data
y_pred = model.predict(X_test)

# Calculate and print the model's accuracy on the test data
print("Model Accuracy:", round(accuracy_score(y_test, y_pred) * 100, 2), "%")

# Request user input to predict the club participation of a student
print("\nEnter student details for prediction")
user_data = {}

# Loop through all columns in the feature set (X)
for col in X.columns:
    # If the column is categorical, ask the user to choose from available options
    if col in label_encoders:
        options = list(label_encoders[col].classes_)
        print(f"\nOptions for {col}: {options}")
        val = input(f"Enter {col}: ")
        # Ensure the input is valid, prompt again if not
        while val not in options:
            print("Invalid input! Please choose from:", options)
            val = input(f"Enter {col}: ")
```

```
        # Encode the input value into a numeric format
        encoded_val = label_encoders[col].transform([val])[0]
        user_data[col] = encoded_val
    else:
        # For numeric columns, directly ask for the value
        val = float(input(f"Enter {col} (numeric): "))
        user_data[col] = val

# Create a DataFrame with the user's input data
input_df = pd.DataFrame([user_data])

# Make a prediction based on the user's input
prediction = model.predict(input_df)[0]

# If the target column is categorical, convert the prediction back to its original label
if target_column in label_encoders:
    result = label_encoders[target_column].inverse_transform([prediction])[0]
else:
    result = prediction

# Output the predicted result
print(f"\nPredicted Club Participation: {result}")
```

Output/Result

Model Accuracy: 40.0%

Sample Screenshot of the Output:

```
Downloading...
From: https://drive.google.com/uc?id=1d_stwG7PnM509FwEjDcFcvZml4RdYzYN
To: /content/club_participation.csv
100%|██████████| 859/859 [00:00<00:00, 2.05MB/s]
Available columns: ['interest_level', 'free_hours_per_week', 'club_participation']
Model Accuracy: 40.0 %

Enter student details for prediction
Enter interest_level (numeric): 7
Enter free_hours_per_week (numeric): 22

Predicted Club Participation: yes
```

```
Downloading...
From: https://drive.google.com/uc?id=1d_stwG7PnM509FwEjDcFcvZml4RdYzYN
To: /content/club_participation.csv
100%|██████████| 859/859 [00:00<00:00, 1.86MB/s]
Available columns: ['interest_level', 'free_hours_per_week', 'club_participation']
Model Accuracy: 35.0 %

Enter student details for prediction
Enter interest_level (numeric): 4
Enter free_hours_per_week (numeric): 10

Predicted Club Participation: no
```

References/Credits

Dataset Source: https://drive.google.com/uc?id=1d_stwG7PnM509FwEjDcFcvZml4RdYzYN
Python Libraries: pandas, sklearn, gdown