

<<<<

COURSE RECOMMENDATION SYSTEM

AI MINI PROJECT

By:

Ethan Saju
Ruben Santhosh
Soundarya Jois
Vignesh R Nair

ENG22AM0175
ENG22AM0162
ENG22AM0134
ENG22AM0142

<<<<

01.

PROBLEM DEFINITION

/ / / / / / / / / /

PROBLEM DEFINITION

The need for a robust student course recommendation system integrating AI arises from the challenge of helping students navigate the increasingly complex and diverse array of courses available in educational institutions. With a vast selection of subjects, electives, and specializations, students often face difficulty in identifying courses that align with their academic interests, career goals, and skill levels. An AI-powered recommendation system addresses this issue by analyzing a wealth of data, including student preferences, as well as course content and outcomes. By leveraging machine learning algorithms, the system can deliver personalized, data-driven course suggestions.





02.

IMPORTANCE OF THE PROJECT



IMPORTANCE OF THE PROJECT

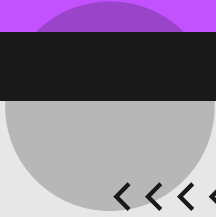
- **AI-Powered Decision Tree Classifier:** The Student Course Recommendation System employs an advanced Decision Tree Classifier algorithm, driven by artificial intelligence, to revolutionize the field of education.
- **Personalized Recommendations:** Tailoring course suggestions to individual student profiles, including education level, year of study, main field, and interests, addresses the challenge of course selection in traditional educational settings.
- **Individualized Education Pathways:** The system's emphasis on personalization diverges from conventional one-size-fits-all approaches, ensuring that students engage with coursework aligned with their aspirations and academic strengths.
- **Comprehensive Analysis:** By analyzing various factors, the system recommends the most suitable courses, enhancing learning outcomes and simplifying the course selection process.
- **Reduction of Decision Fatigue:** The AI-driven recommendations streamline the course selection process, alleviating anxiety and decision fatigue among students.
- **Data-Driven Guidance:** The system's recommendations go beyond traditional advisement, offering scalable, data-driven, and highly individualized academic guidance.



IMPORTANCE OF THE PROJECT

- **Adaptive and Student-Centered:** This innovation signifies a crucial shift towards a more adaptive and student-centered educational landscape, reflecting the evolving needs of modern learners.





03.

SOLUTION





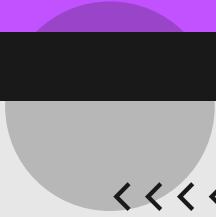
SOLUTION



To forge meaningful connections between students and their ideal courses, the algorithm engages in a meticulous process that begins with a meticulously curated CSV dataset. This dataset serves as a rich repository of knowledge, containing a comprehensive inventory of courses alongside their essential attributes. The decision tree classifier meticulously analyzes this data to uncover hidden patterns and relationships, ultimately constructing a model that expertly aligns student profiles with potential course selections. This model construction involves a two-pronged approach:

First, the model undergoes a rigorous training process, diligently learning from a carefully selected subset of the dataset. This training phase empowers the algorithm to discern the defining characteristics that differentiate various courses and establish the intricate connections between student profiles and optimal course choices. Second, the model's accuracy and reliability undergo a validation process, ensuring its consistency and trustworthiness. This involves evaluating its performance on a separate portion of the dataset, rigorously testing its ability to generalize its knowledge and make accurate recommendations across diverse student profiles.

AL
EN
ND



04.

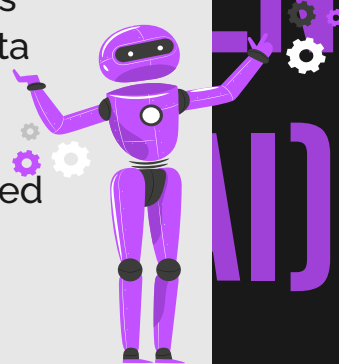
ALGORITHM



ALGORITHM



At the core of this project lies the strategic deployment of a decision tree classifier. This fundamental machine learning algorithm offers unique suitability for the task at hand. Its proficiency in constructing intricate decision paths based on a diverse array of input features makes it an ideal choice. In the context of our course recommendation system, these features encompass the student's level of education, year of study, primary field of interest, and specific learning pursuits. The decision tree classifier's remarkable versatility extends to its adept handling of both categorical and numerical data, rendering it well-equipped to process the rich variety of inputs necessary for effective course recommendations. This inherent strength in data diversity sets it apart from other algorithms and solidifies its position as the cornerstone of our system's personalized learning pathways. By meticulously analyzing each student's unique profile, the decision tree acts as a sophisticated architect, constructing bespoke academic journeys that seamlessly align with individual aspirations.



<<<<

05.

CODE EXPLANATION

/ / / / / / / / /

ARTIFICIAL INTELLIGENCE (AI)

CODE

Main.py :

```
import tkinter as tk
from tkinter import ttk
from course_rec import predictions

# Create the main application window
app = tk.Tk()
app.title("Course Recommendation System")
app.configure(bg='#2C3E50') # Set background color to a dark color

# Function to generate course recommendation

def generate_course():
    user_name = name_var.get()
    user_education = education_var.get()
    user_year = year_var.get()
    selected_field = field_var.get()
    selected_interest = interest_var.get()

    # Call the predictions function from the model file
    recommended_course = predictions(selected_field,
selected_interest)
```

<<<<

ARTIFICIAL INTELLIGENCE (AI)

CODE

```
# Display the recommended course along with user information
result_label.config(
    text=f" Recommended Course: {recommended_course}")

# Define fields, interests, education levels, and years
fields = ['Data Science', 'Internet of Things',
          'Artificial Intelligence', 'Electronics', 'Cyber
Security']
interests = ['Art', 'Music', 'Biology', 'Graphic Design',
             'Marketing']
education_levels = ['Undergraduate', 'Postgraduate']

# Create and place widgets with updated colors
name_label = ttk.Label(app, text="Your Name:",
                       background='#2C3E50', foreground='white')
name_var = ttk.Entry(app)
```

<<<<

ARTIFICIAL INTELLIGENCE (AI)

CODE

```
education_label = ttk.Label(
    app, text="Level of Education:", background='#2C3E50',
    foreground='white')
education_var = ttk.Combobox(app, values=education_levels)

year_label = ttk.Label(
    app, text="Year:", background='#2C3E50', foreground='white')
year_var = ttk.Combobox(app, values=[])

field_label = ttk.Label(app, text="Select Field:",
    background='#2C3E50', foreground='white')
field_var = ttk.Combobox(app, values=fields)

interest_label = ttk.Label(
    app, text="Select Interest:", background='#2C3E50',
    foreground='white')
interest_var = ttk.Combobox(app, values=interests)
# Simplified button style
generate_button = ttk.Button(
    app, text="Generate Course", command=generate_course,
    style='TButton')

result_label = ttk.Label(app, text="Recommended Course: ",
    background='#2C3E50', foreground='white')
```

<<<<

ARTIFICIAL INTELLIGENCE (AI)

CODE

```
# Grid layout
name_label.grid(row=0, column=0, padx=10, pady=10, sticky="w")
name_var.grid(row=0, column=1, padx=10, pady=10, sticky="ew")

education_label.grid(row=1, column=0, padx=10, pady=10, sticky="w")
education_var.grid(row=1, column=1, padx=10, pady=10, sticky="ew")

year_label.grid(row=2, column=0, padx=10, pady=10, sticky="w")
year_var.grid(row=2, column=1, padx=10, pady=10, sticky="ew")

field_label.grid(row=3, column=0, padx=10, pady=10, sticky="w")
field_var.grid(row=3, column=1, padx=10, pady=10, sticky="ew")

interest_label.grid(row=4, column=0, padx=10, pady=10, sticky="w")
interest_var.grid(row=4, column=1, padx=10, pady=10, sticky="ew")

generate_button.grid(row=5, column=0, columnspan=2, pady=10)

result_label.grid(row=6, column=0, columnspan=2, pady=10)
```

<<<<

ARTIFICIAL INTELLIGENCE (AI)

CODE

```
# Function to update year options based on education level

def update_years(*args):
    selected_education = education_var.get()
    if selected_education == 'Undergraduate':
        year_var['values'] = ['1', '2', '3', '4']
    elif selected_education == 'Postgraduate':
        year_var['values'] = ['1', '2']

# Bind the function to the education level combobox
education_var.bind("<<ComboboxSelected>>", update_years)

# Run the application
app.mainloop()
```

<<<<

ARTIFICIAL INTELLIGENCE (AI)

CODE

Course_rec.py:

```
import warnings
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
import pandas as pd

df = pd.read_csv('Couse Recommendation
System\course_recommendation.csv')

X = df.drop(columns=['Course'])

y = df['Course']

X_train, X_test, y_train, y_test = train_test_split(X, y,
train_size=0.8)

model = DecisionTreeClassifier()

model.fit(X_train, y_train)
```

<<<<

CODE

```
def predictions(f, i):
    fields = {'Data Science': 1, 'Internet of Things': 2,
              'Artificial Intelligence': 3, 'Electronics': 4, 'Cyber
Security': 5}
    interests = {'Art': 500, 'Music': 1000, 'Biology': 1500,
                 'Graphic Design': 2000, 'Marketing': 2500}

    field = fields[f]
    interest = interests[i]


    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        return model.predict([[field, interest]])[0]
```

<<<<

ARTIFICIAL
INTELLIGENCE
(AI)

OUTPUT

ART
INT
[AI]

 Course Recommendation System

Your Name:


Level of Education:

Year:

Select Field:

Select Interest:

Recommended Course: Churn Analysis and Retention Strategies

 Course Recommendation System

Your Name:

Level of Education:

Year:

Select Field:

Select Interest:

Recommended Course: AI-assisted branding and graphic design

CODE EXPLANATION

Main.py :

This Python code utilizes the Tkinter library to build a graphical user interface (GUI) for a Course Recommendation System. The application prompts users to input their name, education level, year, field, and interest. Upon clicking the "Generate Course" button, the code calls a prediction function from an external module, determining a recommended course based on the user's inputs. The outcome is then displayed in a labeled section of the GUI.

The GUI features labels, entry fields, comboboxes, and buttons for a user-friendly interface. Users can select their education level, year, field of interest, and area of interest through dropdown menus. Additionally, the code dynamically updates the available years based on the selected education level, enhancing the user experience.

In summary, the application assists users in obtaining personalized course recommendations by employing a straightforward and interactive Tkinter-based interface.

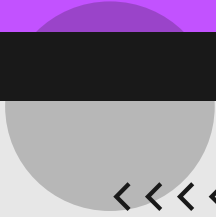


CODE EXPLANATION



Course_rec.py :

This code begins by importing necessary libraries, including scikit-learn for machine learning functionalities and pandas for data manipulation. It reads a CSV file containing course recommendation data, separates features (X) and target labels (y), and splits the dataset into training and testing sets. Subsequently, a Decision Tree Classifier is instantiated, trained on the training data, and tested for accuracy. The predictions function takes user-selected field and interest inputs, maps them to numerical values, and utilizes the trained model to predict a recommended course. The mappings for fields and interests are defined within the function. Notably, the code handles warnings during the prediction process. Overall, this script demonstrates a machine learning approach to course recommendation based on given user preferences.



06.

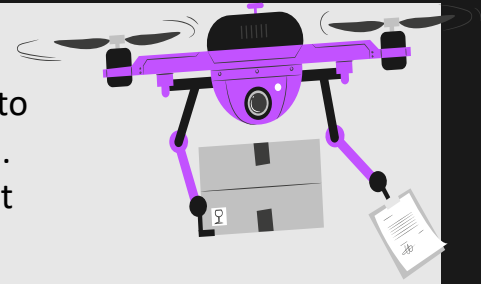
CONCLUSION



CONCLUSION

The Course Recommendation System is a valuable tool for students looking to find the best course to take based on their interests and preferred main field. The program's graphical user interface is user-friendly and intuitive, making it easy for users to input their information and receive a course recommendation. The implementation of different fields to input users interest, level of education, year of study and the type of domain they like to work on ensures that the recommended course is suitable for the user.

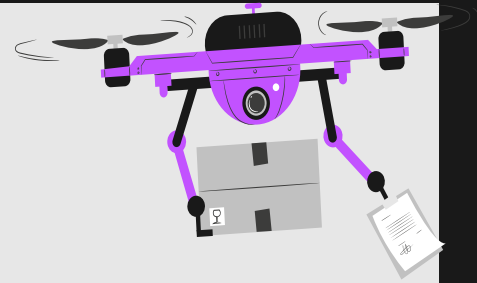
Furthermore, the Course Recommendation System employs advanced algorithms that analyze the user's input data, taking into account not only their stated interests but also their educational background, current year of study, and preferred domain of work. This multi-faceted approach enhances the accuracy of the course recommendations, providing students with tailored suggestions that align with their academic and professional aspirations.



CONCLUSION

The system's versatility extends beyond merely suggesting courses; it also considers the dynamic nature of educational programs and industry trends. Regular updates and integration with relevant databases ensure that the recommendations remain current and reflect the latest advancements in various fields of study.

In summary, the Course Recommendation System stands as an indispensable resource for students, offering a comprehensive and sophisticated approach to course selection. Its user-friendly interface, algorithmic precision, adaptability to evolving educational landscapes, and commitment to transparency collectively make it an invaluable tool for guiding students towards courses that align perfectly with their interests and academic goals.





ARTI

ARTI

THANK YOU

(AI)

(AI)