SmartSage – AI that teaches like a wise mentor

By:- Vansh Gupta

03/30/25

Summary

Our project focuses on AI-driven personalized learning for K-12 students. It predicts assessment scores, determines promotion eligibility, and customizes study material based on student levels. The system leverages data analysis, machine learning, and NLP to curate content dynamically for effective learning.

PROBLEM STATEMENTS

- Predict assessment score & promotion decision → Need a model to train on assessment data.
- Decide what to keep/skip at different levels → Need clear rules based on grade levels.
- Predict study material based on student level → Need a model to map materials to student levels.
- Generate adaptive teaching material dynamically → Requires NLP model (Gemini) to simplify/modify content dynamically.

SOLUTIONS

01

- Develop an AI model to analyze past performance, study time, and IQ to predict assessment scores and determine promotion eligibility.

02

- Use AI-driven recommendations to match students with the right learning resources, ensuring personalized and effective learning experiences.

03

- Identify key topics for each grade level, ensuring an optimized learning path by removing redundant or complex topics based on student understanding.

04

- Implement NLP-based models to simplify, restructure, and generate adaptive learning materials in real time based on student needs.

STEPS TO ACHIEVE

- Step 1: Collect student data (IQ, scores, study time, etc.).
- Step 2: Perform Exploratory Data Analysis (EDA) to find patterns.
- Step 3: Train AI/ML models for prediction and recommendations.
- Step 4: Implement an NLP model to simplify teaching materials.

Roles And Responsibility

1. Vansh Gupta (Co-Founder & Project Lead)

- Project Management & Leadership – Overseeing the project's overall vision, execution, and progress.
- Technical Development – Handling coding, software architecture, and integrations.
- Innovation & Problem-Solving – Ensuring new features and optimizations align with project goals.

2. Dipaya Das (Documentation Specialist)

- Report Writing & Documentation – Creating and maintaining structured project reports, user manuals, and technical documentation.
- Requirement Gathering – Documenting project requirements, use cases, and functional specifications.
- Presentation & Reports – Preparing PPTs, PDFs, and Word reports for stakeholders.

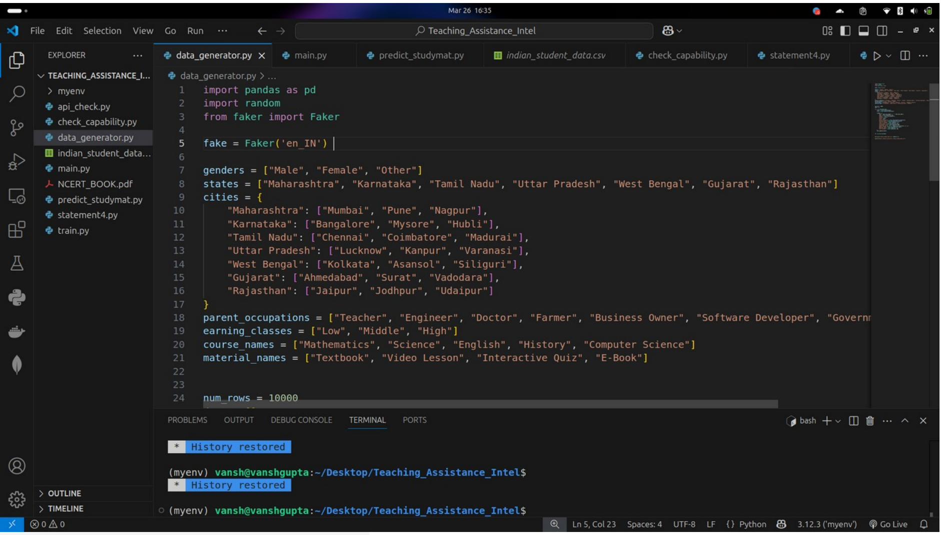
MODEL USED

- Regression Models – Predict student performance.

- Classification Models – Identify students at risk.
- NLP Models (Gemini/GPT) – Simplify learning content.

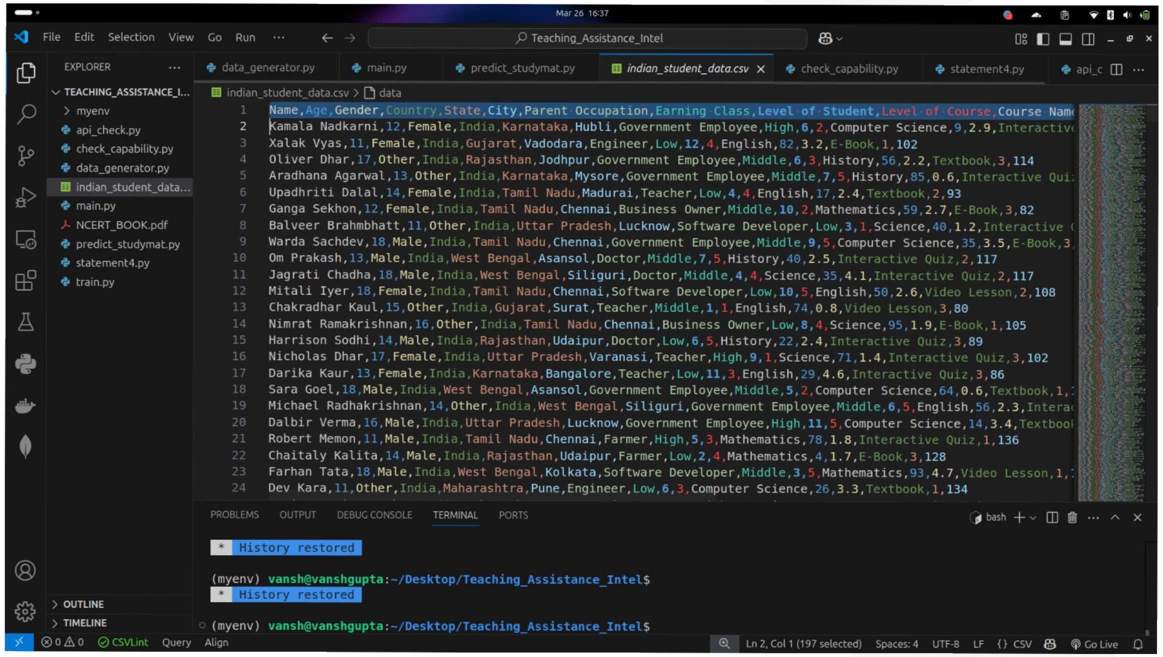
- Recommendation Systems – Suggest study material.

Data



```
import pandas as pd
import random
from faker import Faker

fake = Faker('en_IN')

genders = ["Male", "Female", "Other"]
states = ["Maharashtra", "Karnataka", "Tamil Nadu", "Uttar Pradesh", "West Bengal", "Gujarat", "Rajasthan"]
cities = {
    "Maharashtra": ["Mumbai", "Pune", "Nagpur"],
    "Karnataka": ["Bangalore", "Mysore", "Hubli"],
    "Tamil Nadu": ["Chennai", "Coimbatore", "Madurai"],
    "Uttar Pradesh": ["Lucknow", "Kanpur", "Varanasi"],
    "West Bengal": ["Kolkata", "Asansol", "Siliguri"],
    "Gujarat": ["Ahmedabad", "Surat", "Vadodara"],
    "Rajasthan": ["Jaipur", "Jodhpur", "Udaipur"]
}
parent_occupations = ["Teacher", "Engineer", "Doctor", "Farmer", "Business Owner", "Software Developer", "Governn
earning_classes = ["Low", "Middle", "High"]
course_names = ["Mathematics", "Science", "English", "History", "Computer Science"]
material_names = ["Textbook", "Video Lesson", "Interactive Quiz", "E-Book"]


num_rows = 10000
```

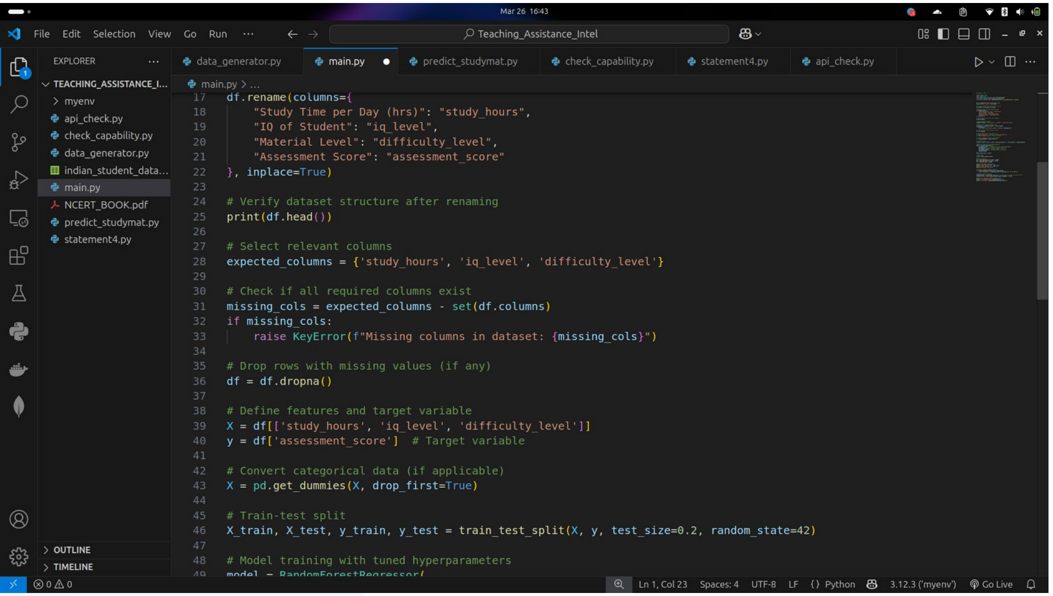
```
* History restored

(myenv) vanshgupta:~/Desktop/Teaching_Assistance_Intel$
* History restored

(myenv) vanshgupta:~/Desktop/Teaching_Assistance_Intel$
```



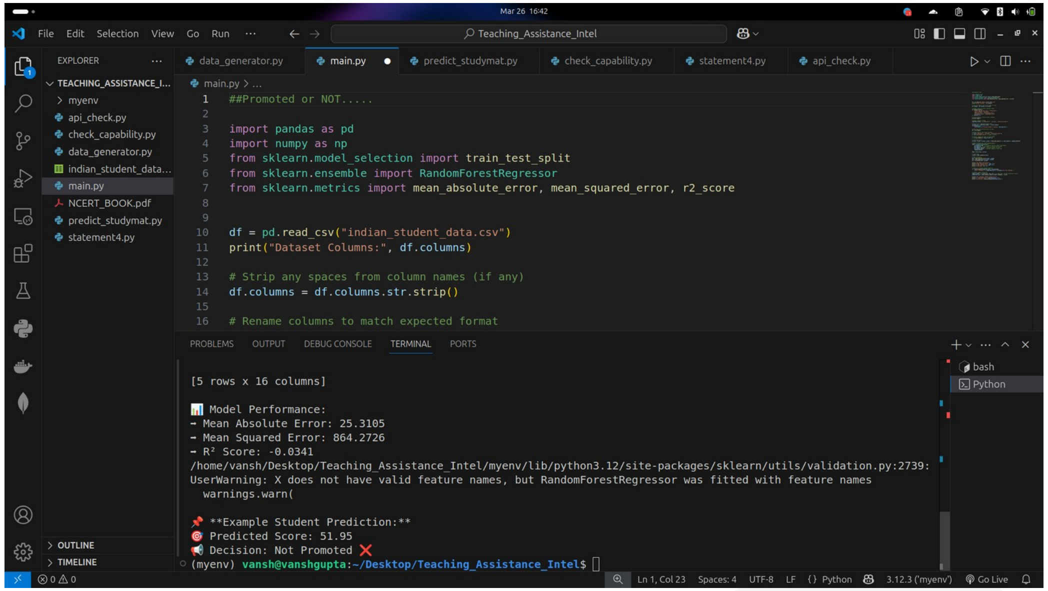
```
Name,Age,Gender,Country,State,City,Parent Occupation,Earning Class,Level of Student,Level of Course,Course Name
Kamala Nadkarni,12,Female,India,Karnataka,Hubli,Government Employee,High,6,2,Computer Science,9,2.9,Interactive
Xalak Vyas,11,Female,India,Gujarat,Vadodara,Engineer,Low,12,4,English,82,3.2,E-Book,1,102
Oliver Dhar,17,Other,India,Rajasthan,Jodhpur,Government Employee,Middle,6,3,History,56,2.2,Textbook,3,114
Aradhana Agarwal,13,Other,India,Karnataka,Mysore,Government Employee,Middle,7,5,History,85,0.6,Interactive Qui
Upadhriti Dalal,14,Female,India,Tamil Nadu,Madurai,Teacher,Low,4,4,English,17,2.4,Textbook,2,93
Ganga Sekhon,12,Female,India,Tamil Nadu,Chennai,Business Owner,Middle,10,2,Mathematics,59,2.7,E-Book,3,82
Balveer Brahmbhatt,11,Other,India,Uttar Pradesh,Lucknow,Software Developer,Low,3,1,Science,40,1.2,Interactive
Warda Sachdev,18,Male,India,Tamil Nadu,Chennai,Government Employee,Middle,9,5,Computer Science,35,3.5,E-Book,3,
Om Prakash,13,Male,India,West Bengal,Asansol,Doctor,Middle,7,5,History,40,2.5,Interactive Quiz,2,117
Jagrati Chadha,18,Male,India,West Bengal,Siliguri,Doctor,Middle,4,4,Science,35,4.1,Interactive Quiz,2,117
Mitali Iyer,18,Female,India,Tamil Nadu,Chennai,Software Developer,Low,10,5,English,50,2.6,Video Lesson,2,108
Chakradhar Kaul,15,Other,India,Gujarat,Surat,Teacher,Middle,1,1,English,74,0.8,Video Lesson,3,80
Nimrat Ramakrishnan,16,Other,India,Tamil Nadu,Chennai,Business Owner,Low,8,4,Science,95,1.9,E-Book,1,105
Harrison Sodhi,14,Male,India,Rajasthan,Udaipur,Doctor,Low,6,5,History,22,2.4,Interactive Quiz,3,89
Nicholas Dhar,17,Female,India,Uttar Pradesh,Varanasi,Teacher,High,9,1,Science,71,1.4,Interactive Quiz,3,102
Darika Kaur,13,Female,India,Karnataka,Bangalore,Teacher,Low,11,3,English,29,4.6,Interactive Quiz,3,86
Sara Goel,18,Male,India,West Bengal,Asansol,Government Employee,Middle,5,2,Computer Science,64,0.6,Textbook,1,
Michael Radhakrishnan,14,Other,India,West Bengal,Siliguri,Government Employee,Middle,6,5,English,56,2.3,Interac
Dalbir Verma,16,Male,India,Uttar Pradesh,Lucknow,Government Employee,High,11,5,Computer Science,14,3.4,Textbook
Robert Memon,11,Male,India,Tamil Nadu,Chennai,Farmer,High,5,3,Mathematics,78,1.8,Interactive Quiz,1,136
Chaitaly Kalita,14,Male,India,Rajasthan,Udaipur,Farmer,Low,2,4,Mathematics,4,1.7,E-Book,3,128
Farhan Tata,18,Male,India,West Bengal,Kolkata,Software Developer,Middle,3,5,Mathematics,93,4.7,Video Lesson,1,
Dev Kara,11,Other,India,Maharashtra,Pune,Engineer,Low,6,3,Computer Science,26,3.3,Textbook,1,134
```

```
* History restored

(myenv) vanshgupta:~/Desktop/Teaching_Assistance_Intel$
* History restored

(myenv) vanshgupta:~/Desktop/Teaching_Assistance_Intel$
```

PROMOTED OR NOT



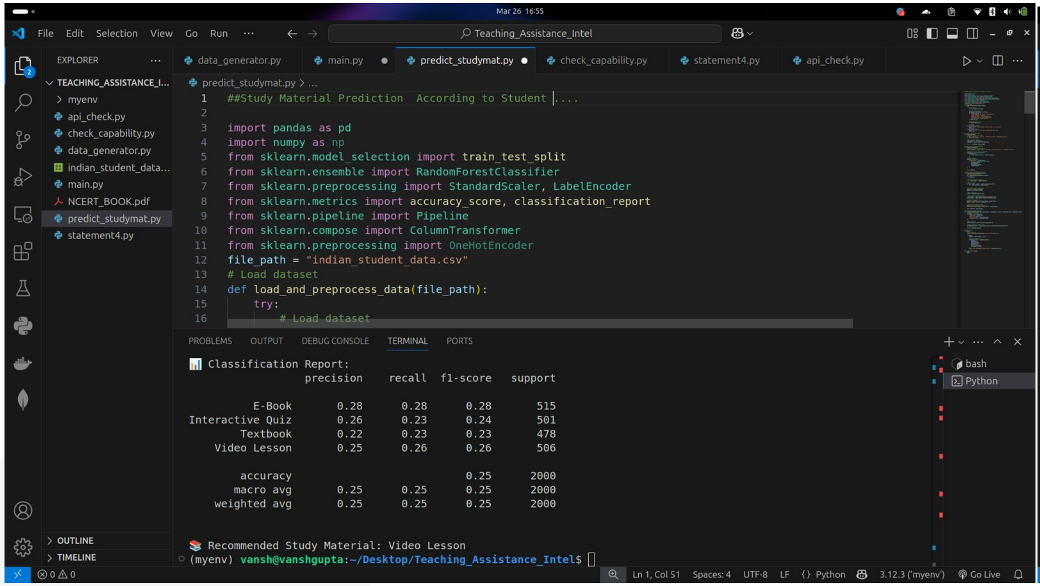
```python
df.rename(columns={
    "Study Time per Day (hrs)": "study_hours",
    "IQ of Student": "iq_level",
    "Material Level": "difficulty_level",
    "Assessment Score": "assessment_score"
}, inplace=True)

# Verify dataset structure after renaming
print(df.head())

# Select relevant columns
expected_columns = {'study_hours', 'iq_level', 'difficulty_level'}

# Check if all required columns exist
missing_cols = expected_columns - set(df.columns)
if missing_cols:
    raise KeyError(f"Missing columns in dataset: {missing_cols}")

# Drop rows with missing values (if any)
df = df.dropna()

# Define features and target variable
X = df[['study_hours', 'iq_level', 'difficulty_level']]
y = df['assessment_score']  # Target variable

# Convert categorical data (if applicable)
X = pd.get_dummies(X, drop_first=True)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model training with tuned hyperparameters
model = RandomForestRegressor(
```



```python
##Promoted or NOT.....

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score


df = pd.read_csv("indian_student_data.csv")
print("Dataset Columns:", df.columns)

# Strip any spaces from column names (if any)
df.columns = df.columns.str.strip()

# Rename columns to match expected format
```

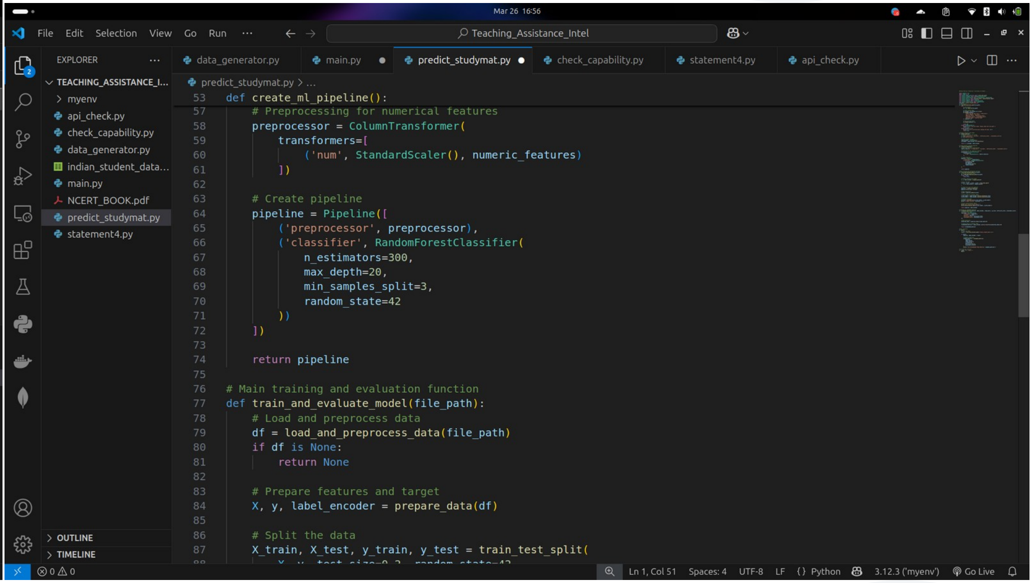
```
[5 rows x 16 columns]

📊 Model Performance:
➡ Mean Absolute Error: 25.3105
➡ Mean Squared Error: 864.2726
➡ R² Score: -0.0341
/home/vansh/Desktop/Teaching_Assistance_Intel/myenv/lib/python3.12/site-packages/sklearn/utils/validation.py:2739:
UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(

📌 **Example Student Prediction:**
🎯 Predicted Score: 51.95
📢 Decision: Not Promoted ❌
(myenv) vansh@vanshgupta:~/Desktop/Teaching_Assistance_Intel$
```

STUDY MATERIAL PREDICTION



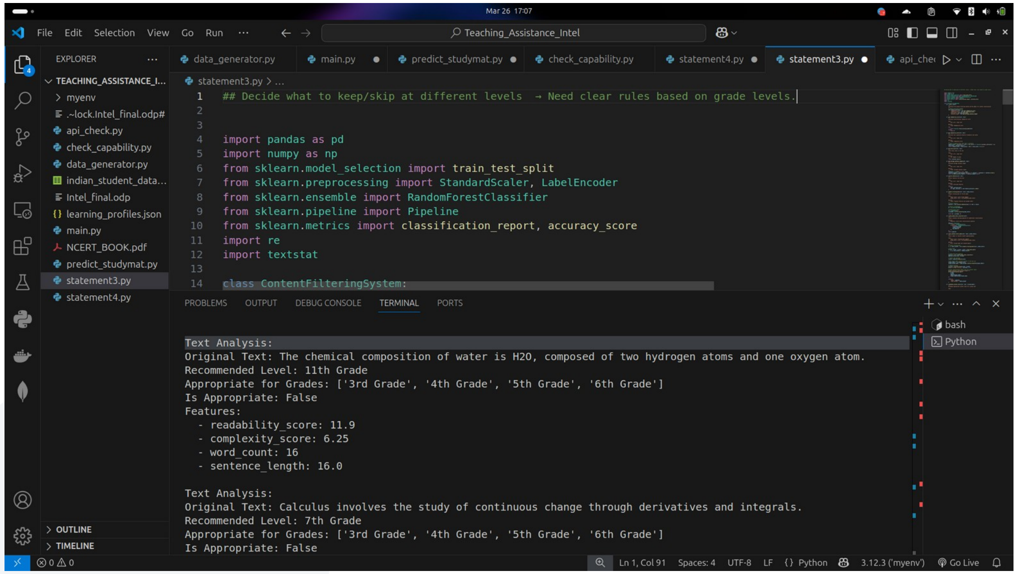
```python
##Study Material Prediction  According to Student ....

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import accuracy_score, classification_report
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
file_path = "indian_student_data.csv"
# Load dataset
def load_and_preprocess_data(file_path):
    try:
        # Load dataset
```

```
📊 Classification Report:
                  precision    recall  f1-score   support

          E-Book       0.28      0.28      0.28       515
Interactive Quiz       0.26      0.23      0.24       501
        Textbook       0.22      0.23      0.23       478
    Video Lesson       0.25      0.26      0.26       506

        accuracy                           0.25      2000
       macro avg       0.25      0.25      0.25      2000
    weighted avg       0.25      0.25      0.25      2000

📚 Recommended Study Material: Video Lesson
(myenv) vansh@vanshgupta:~/Desktop/Teaching_Assistance_Intel$
```



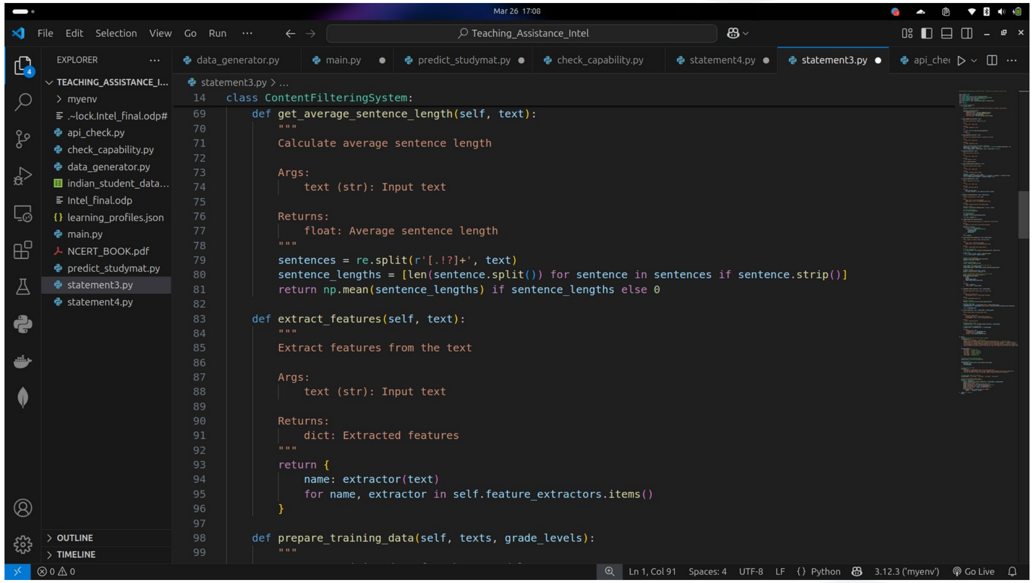
```python
def create_ml_pipeline():
    # Preprocessing for numerical features
    preprocessor = ColumnTransformer(
        transformers=[
            ('num', StandardScaler(), numeric_features)
        ])

    # Create pipeline
    pipeline = Pipeline([
        ('preprocessor', preprocessor),
        ('classifier', RandomForestClassifier(
            n_estimators=300,
            max_depth=20,
            min_samples_split=3,
            random_state=42
        ))
    ])

    return pipeline

# Main training and evaluation function
def train_and_evaluate_model(file_path):
    # Load and preprocess data
    df = load_and_preprocess_data(file_path)
    if df is None:
        return None

    # Prepare features and target
    X, y, label_encoder = prepare_data(df)

    # Split the data
    X_train, X_test, y_train, y_test = train_test_split(
```

KEEP/SKIP DIFFERNT LEVEL



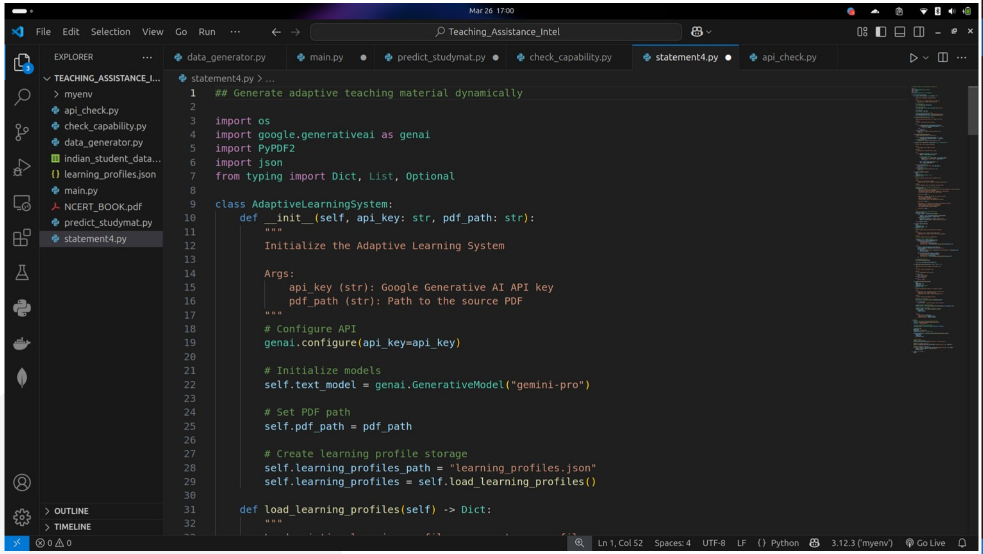
```python
## Decide what to keep/skip at different levels  → Need clear rules based on grade levels.


import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, accuracy_score
import re
import textstat

class ContentFilteringSystem:
```

```
Text Analysis:
Original Text: The chemical composition of water is H2O, composed of two hydrogen atoms and one oxygen atom.
Recommended Level: 11th Grade
Appropriate for Grades: ['3rd Grade', '4th Grade', '5th Grade', '6th Grade']
Is Appropriate: False
Features:
  - readability_score: 11.9
  - complexity_score: 6.25
  - word_count: 16
  - sentence_length: 16.0

Text Analysis:
Original Text: Calculus involves the study of continuous change through derivatives and integrals.
Recommended Level: 7th Grade
Appropriate for Grades: ['3rd Grade', '4th Grade', '5th Grade', '6th Grade']
Is Appropriate: False
```

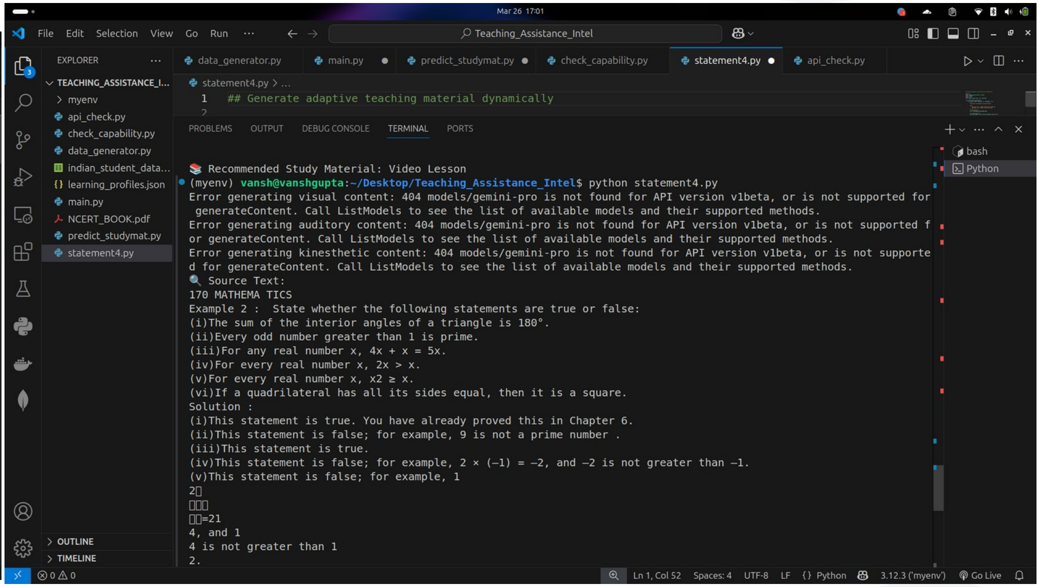


```python
class ContentFilteringSystem:
    def get_average_sentence_length(self, text):
        """
        Calculate average sentence length

        Args:
            text (str): Input text

        Returns:
            float: Average sentence length
        """
        sentences = re.split(r'[.!?]+', text)
        sentence_lengths = [len(sentence.split()) for sentence in sentences if sentence.strip()]
        return np.mean(sentence_lengths) if sentence_lengths else 0

    def extract_features(self, text):
        """
        Extract features from the text

        Args:
            text (str): Input text

        Returns:
            dict: Extracted features
        """
        return {
            name: extractor(text)
            for name, extractor in self.feature_extractors.items()
        }

    def prepare_training_data(self, texts, grade_levels):
        """
```

GENERATE TEACHING MATERIAL



```python
## Generate adaptive teaching material dynamically

import os
import google.generativeai as genai
import PyPDF2
import json
from typing import Dict, List, Optional

class AdaptiveLearningSystem:
    def __init__(self, api_key: str, pdf_path: str):
        """
        Initialize the Adaptive Learning System

        Args:
            api_key (str): Google Generative AI API key
            pdf_path (str): Path to the source PDF
        """
        # Configure API
        genai.configure(api_key=api_key)

        # Initialize models
        self.text_model = genai.GenerativeModel("gemini-pro")

        # Set PDF path
        self.pdf_path = pdf_path

        # Create learning profile storage
        self.learning_profiles_path = "learning_profiles.json"
        self.learning_profiles = self.load_learning_profiles()

    def load_learning_profiles(self) -> Dict:
        """
```



```
📚 Recommended Study Material: Video Lesson
(myenv) vansh@vanshgupta:~/Desktop/Teaching_Assistance_Intel$ python statement4.py
Error generating visual content: 404 models/gemini-pro is not found for API version v1beta, or is not supported for
 generateContent. Call ListModels to see the list of available models and their supported methods.
Error generating auditory content: 404 models/gemini-pro is not found for API version v1beta, or is not supported f
or generateContent. Call ListModels to see the list of available models and their supported methods.
Error generating kinesthetic content: 404 models/gemini-pro is not found for API version v1beta, or is not supporte
d for generateContent. Call ListModels to see the list of available models and their supported methods.
🔍 Source Text:
170 MATHEMA TICS
Example 2 :  State whether the following statements are true or false:
(i)The sum of the interior angles of a triangle is 180°.
(ii)Every odd number greater than 1 is prime.
(iii)For any real number x, 4x + x = 5x.
(iv)For every real number x, 2x > x.
(v)For every real number x, x2 ≥ x.
(vi)If a quadrilateral has all its sides equal, then it is a square.
Solution :
(i)This statement is true. You have already proved this in Chapter 6.
(ii)This statement is false; for example, 9 is not a prime number .
(iii)This statement is true.
(iv)This statement is false; for example, 2 × (–1) = –2, and –2 is not greater than –1.
(v)This statement is false; for example, 1
2□
□□□
□□=21
4, and 1
4 is not greater than 1
2.
```

CONCLUSION

- In this project, we successfully executed our objectives by efficiently dividing tasks and responsibilities. As the project lead, I, Vansh Gupta, managed the overall development, technical implementation, and project execution. My role involved ensuring smooth coordination, decision-making, and driving the project forward with innovative solutions.
- Meanwhile, Dipaya Das focused on documentation, ensuring that all project reports, technical manuals, and presentations were well-structured and comprehensive. The documentation played a crucial role in maintaining clarity, tracking progress, and presenting our work effectively.
- Through our collaborative efforts, we achieved the desired outcomes, streamlined the workflow, and delivered a well-documented and technically sound project. This structured approach ensured that all aspects of the project were handled efficiently, leading to a successful completion.

THANK YOU