Department of Computer Science Engineering (AI)

ARTIFICIAL INTELLIGENCE

Project Report

On

Classify Students Based on Study Methods

By

ANSHUL ANAND

202401100300056

DataSet Used:student_methods.csv

Date: 22/04/2025

TABLE OF CONTENTS

CHAPTER 1: - INTRODUCTION

1.1. INTRODUCTION

1.2. **Methodology**

1.3.code

1.4.**Results**

1.5.**Credits**

1.1 INTRODUCTION

Understanding individual learning styles—visual, auditory, and kinesthetic—can help improve how students absorb and retain information. Each student has a unique preference for learning, and tailoring teaching methods to these styles can enhance learning outcomes. In this project, we aim to predict a student's learning style using their visual, auditory, and kinesthetic scores.

We use a supervised machine learning approach to classify students based on their sensory scores. The dataset includes labeled data for each student’s learning style along with their corresponding scores. A Random Forest Classifier is trained to make predictions, and its performance is evaluated using accuracy and confusion matrix metrics. This project shows how machine learning can support more personalized education by identifying learning preferences automatically.

NATIONAL INSTITUTIONAL RANKING FRAMEWORK
NIRF-2023
Engineering Rank Band (151-200)
Pharmacy Rank - 88
Innovation Rank Band (51-100)
www.kiet.edu
Delhi-NCR, Ghaziabad
KIET GROUP OF INSTITUTIONS
Connecting Life with Learning
CELEBRATING 25 YEARS OF EXCELLENCE
A+ NAAC
Accredited by
NAAC with Grade "A+"
IQAC

1.2 Methodology

The following approach was used to solve the problem:

Data Preprocessing:

- **Loaded the dataset containing `visual_score`, `auditory_score`, and `kinesthetic_score` along with the target `learning_style`.**
- **Checked for missing values to ensure the data quality.**
- **Encoded the `learning_style` labels into numerical values using LabelEncoder.**

Model Selection:

- **Selected Random Forest Classifier due to its strong performance in multi-class classification problems and robustness against overfitting.**

Data Splitting:

- **Split the data into 80% training and 20% testing sets using `train_test_split()` to evaluate model performance reliably.**

Training:

- **Trained the Random Forest model on the training dataset using default hyperparameters.**

Evaluation:

- **Predicted learning styles on the test dataset.**

- **Evaluated model performance using Accuracy Score, Confusion Matrix, and Classification Report.**

- **Visualized the Confusion Matrix using a heatmap for better interpretation.**
-

CODE

```python
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Machine learning libraries
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

# Load the dataset
df = pd.read_csv('/content/student_methods.csv')  # Adjust the path if needed

# Display the first few rows
print("First 5 rows of the dataset:")
print(df.head())

# Check for missing values
print("\nChecking for missing values:")
print(df.isnull().sum())

# Encode the target variable (learning_style)
le = LabelEncoder()
df['learning_style_encoded'] = le.fit_transform(df['learning_style'])

# Define features and target
X = df[['visual_score', 'auditory_score', 'kinesthetic_score']]
y = df['learning_style_encoded']

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

```
# Initialize and train the Random Forest Classifier
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

print("Accuracy Score:", accuracy_score(y_test, y_pred))

# Plot Confusion Matrix
conf_mat = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_mat, annot=True, fmt='d', cmap='Blues',
            xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```

OUTPUT

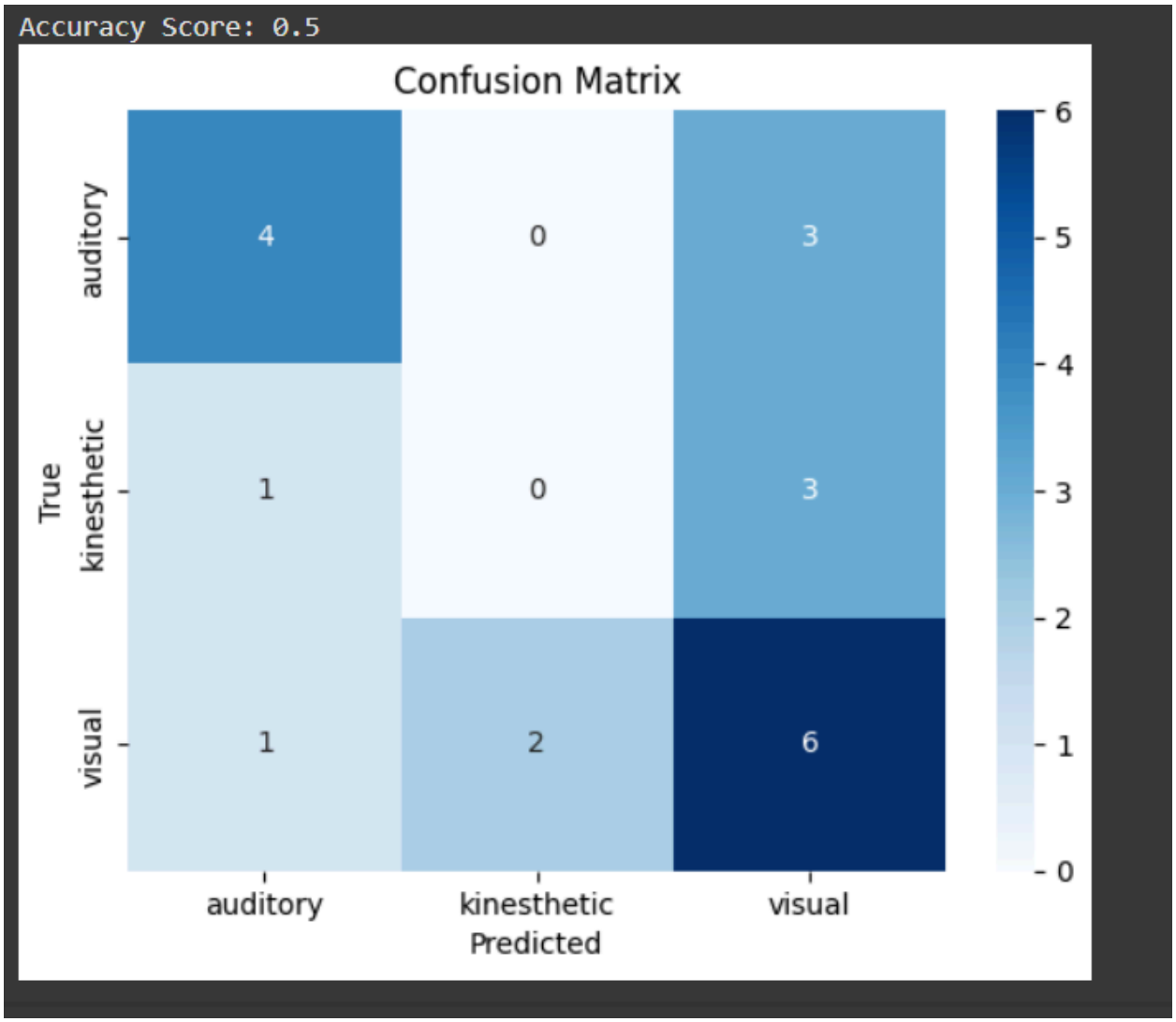
```
First 5 rows of the dataset:
   visual_score  auditory_score  kinesthetic_score learning_style
0      8.000301        1.389837           9.686887         visual
1      8.401052        7.294055           4.853655         visual
2      9.124874        3.975049           6.688173       auditory
3      5.724100        7.702631           7.535001       auditory
4      5.060739        4.711628           4.302653    kinesthetic

Checking for missing values:
visual_score         0
auditory_score       0
kinesthetic_score    0
learning_style       0
dtype: int64

Classification Report:
              precision    recall  f1-score   support

           0       0.67      0.57      0.62         7
           1       0.00      0.00      0.00         4
           2       0.50      0.67      0.57         9

    accuracy                           0.50        20
   macro avg       0.39      0.41      0.40        20
weighted avg       0.46      0.50      0.47        20
```



References/Credits:

Dataset: Provided internally for academic purposes.

Libraries Used:

- Pandas (for data handling)
- NumPy (for numerical operations)
- Scikit-learn (for machine learning models and evaluation)
- Matplotlib and Seaborn (for data visualization)

Model Used: Random Forest Classifier from Scikit-learn.

Tools Used: Google Colab for coding and model training.

Special Thanks: To course instructors and open-source community documentation.