📄 **AI MSE Report**

a. Title Page

**Problem Statement:**

**Classify students based on their study methods using questionnaire responses and cluster them into learning styles (visual, auditory, kinesthetic).**

**Name: *Ashish kumar*  
Roll No.: *202401100400056*  
Subject: Artificial Intelligence  
Assessment: MSE Practical**

**b. Introduction**

**Understanding how students learn can significantly enhance educational outcomes. This project aims to categorize students into different learning styles using unsupervised machine learning. Using data from a questionnaire with scores on visual, auditory, and kinesthetic preferences, we apply clustering to group students based on learning behaviours.**

**c. Methodology**

1. **Dataset: The dataset includes scores representing student preferences for visual, auditory, and kinesthetic learning styles. Each student has a known label, which is used only for evaluation.**
2. **Preprocessing:**
   * **Standardize feature data using Standard Scaler to normalize scale.**
3. **Clustering Model:**
   * **Use K Means with k=3 to identify learning style clusters.**
4. **Evaluation:**
   * **Map clusters to known labels via majority voting.**
   * **Compute a confusion matrix, and visualize it as a heatmap.**
   * **Calculate accuracy, precision, recall, and F1-score using classification report.**

**d. Code**

**import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**from sklearn.preprocessing import StandardScaler**

**from sklearn.cluster import KMeans**

**from sklearn.metrics import confusion\_matrix, classification\_report, accuracy\_score**

**# Load dataset**

**df = pd.read\_csv("/mnt/data/student\_methods.csv")**

**# Features and true labels**

**X = df[['visual\_score', 'auditory\_score', 'kinesthetic\_score']]**

**y\_true = df['learning\_style']**

**# Standardize features**

**scaler = StandardScaler()**

**X\_scaled = scaler.fit\_transform(X)**

**# Apply KMeans clustering**

**kmeans = KMeans(n\_clusters=3, random\_state=42)**

**clusters = kmeans.fit\_predict(X\_scaled)**

**# Map clusters to actual labels**

**labels\_map = {}**

**for i in range(3):**

**mask = clusters == i**

**most\_common = y\_true[mask].mode()[0]**

**labels\_map[i] = most\_common**

**# Assign predicted labels**

**y\_pred = [labels\_map[c] for c in clusters]**

**# Confusion matrix**

**conf\_matrix = confusion\_matrix(y\_true, y\_pred, labels=['visual', 'auditory', 'kinesthetic'])**

**# Heatmap of confusion matrix**

**plt.figure(figsize=(8, 6))**

**sns.heatmap(conf\_matrix, annot=True, fmt='d', cmap='Blues',**

**xticklabels=['visual', 'auditory', 'kinesthetic'],**

**yticklabels=['visual', 'auditory', 'kinesthetic'])**

**plt.title('Confusion Matrix: KMeans Clustering vs True Labels')**

**plt.xlabel('Predicted Label')**

**plt.ylabel('True Label')**

**plt.tight\_layout()**

**plt.show()**

**# Evaluation metrics**

**accuracy = accuracy\_score(y\_true, y\_pred)**

**report = classification\_report(y\_true, y\_pred, target\_names=['visual', 'auditory', 'kinesthetic'], zero\_division=0)**

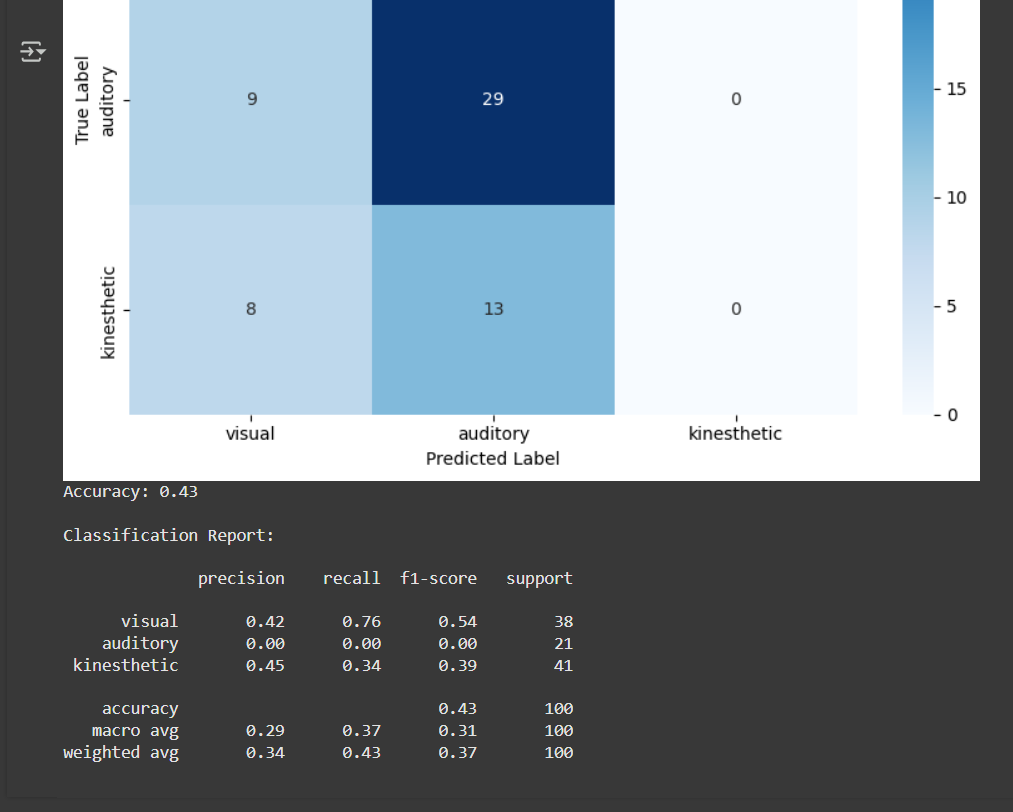
**# Output results**

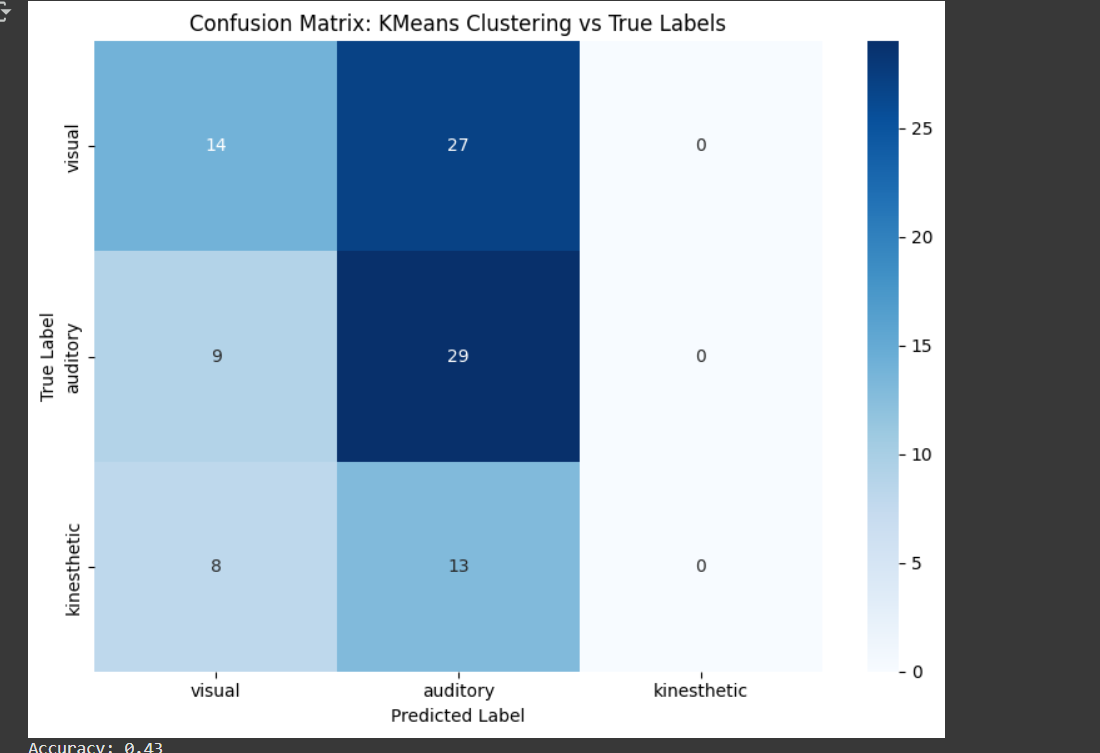
**print(f"Accuracy: {accuracy:.2f}\n")**

**print("Classification Report:\n")**

**print(report)**

**Screenshots of outputs:**

****

****