CLASSIFY STUDENT STUDY METHODS REPORT

A Project Report

Submitted by:-

NISHCHAY AGARWAL

202401100300163

SESSION 2024-25

in

CSE(AI)

In the partial fulfillment for the award of the degree of

Bachelor of Technology

KIET Group of Institutions

April 2025

INTRODUCTION

In the realm of education, every student has a unique way of processing and retaining information. Recognizing and adapting to these differences can significantly enhance the effectiveness of teaching methods. The concept of learning styles provides a framework to understand these preferences, categorizing learners into three primary types: visual, auditory, and kinesthetic.

This project focuses on building a simple, data-driven classification system to identify the dominant learning style of students based on their responses to a structured questionnaire. The input consists of scores reflecting a student's inclination toward visual, auditory, and kinesthetic learning. By comparing these scores, each student is categorized into the learning style where their score is highest.

The objective of this project is not only to automate the classification process but also to visualize the distribution of learning styles in a given dataset. Such insights can assist educators in tailoring their teaching strategies to better meet the diverse needs of their students, ultimately enhancing the learning experience and academic performance.

METHODOLOGY

The goal of this study was to classify students based on their learning styles (visual, auditory, kinesthetic) and analyze the classification results using various evaluation metrics. Additionally, clustering techniques were applied to segment students based on their learning scores. The methodology can be broken down into the following steps:

1. Data Collection

- **The dataset was sourced from a CSV file named `student_methods.csv`, which contained information on students' learning scores across three categories: visual, auditory, and kinesthetic.**

- **The dataset consisted of several columns, including `visual_score`, `auditory_score`, `kinesthetic_score`, and `learning_style`, the latter representing the actual learning style of each student.**

2. Preprocessing and Data Preparation

- **The dataset was loaded using the `pandas` library, which allowed for easy manipulation and analysis of the data.**

- **Missing values, if any, were handled (though this wasn't explicitly mentioned in the original code). For completeness, data validation and cleansing should be performed in real-world scenarios to ensure accuracy.**

3. Classification of Learning Styles

- A function `determine_learning_style()` was defined to classify students based on the highest score among the three learning categories (visual, auditory, kinesthetic).

- The function assigns a predicted learning style (`predicted_learning_style`) to each student based on the highest score in one of the three categories.

- The classification process was applied using the `apply()` method in pandas, iterating over each row of the dataset.

4. Visualization of Learning Style Distribution

- Bar Chart: A bar chart was created to visualize the distribution of learning styles among the students.

- Pie Chart: A pie chart was also generated to display the percentage share of each learning style in the dataset.

- Box Plot: A box plot was used to examine the distribution of scores (visual, auditory, and kinesthetic) across all students.

- Swarm Plot: A swarm plot was employed to show the spread of scores by predicted learning style, providing detailed insights into how scores vary within each category.

- Pair Plot: A pair plot was used to explore the relationships between the three score types (visual, auditory, kinesthetic), with points colored by the predicted learning style.

5. Evaluation of Classification Performance

- The performance of the classification model was evaluated using a confusion matrix, comparing the predicted learning styles (`predicted_learning_style`) with the actual learning styles (`learning_style`).

- Confusion Matrix: The confusion matrix was calculated to evaluate the accuracy of the model's predictions. It was visualized as a heatmap using the `seaborn` library to facilitate interpretation.

- Evaluation Metrics: The following metrics were computed:

  - Accuracy: The overall proportion of correct predictions.

  - Precision: The weighted precision across all learning styles.

  - Recall: The weighted recall across all learning styles.

- These metrics were computed using functions from the `sklearn.metrics` module.

6. Clustering (Segmentation)

- A clustering analysis was conducted to segment the students based on their learning scores using the KMeans algorithm.

- Dimensionality Reduction: The number of features was reduced from three to two using Principal Component Analysis (PCA) to visualize the clustering results effectively in 2D.

- KMeans Clustering: The KMeans algorithm was applied to group the students into three clusters, as it is assumed that there are three distinct learning styles.

- A scatter plot was generated to visualize the students' clustering in a two-dimensional space, colored by the assigned cluster labels.

7. Analysis and Interpretation

- **The distribution of learning styles was analyzed using the visualizations, providing insights into the prevalence of each learning style within the dataset.**

- **The evaluation metrics were interpreted to assess the effectiveness of the classification approach.**

- **The clustering results were analyzed to identify potential groupings of students with similar learning profiles, which could inform personalized learning strategies.**

CODE

NISHCHAYAGARWAL_202401100300163 (3).ipynb ×

C: > Users > Dell > Downloads > NISHCHAYAGARWAL_202401100300163 (3).ipynb > # Import necessary libraries

Generate + Code + Markdown | Run All Clear All Outputs | Outline ···

```python
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import numpy as np

# Load the CSV
df = pd.read_csv('student_methods.csv')

# Define function to determine dominant learning style
def determine_learning_style(row):
    scores = {
        'visual': row['visual_score'],
        'auditory': row['auditory_score'],
        'kinesthetic': row['kinesthetic_score']
    }
    return max(scores, key=scores.get)

# Apply classification
df['predicted_learning_style'] = df.apply(determine_learning_style, axis=1)

# Display classification result
print("Classified Students:")
print(df[['visual_score', 'auditory_score', 'kinesthetic_score', 'predicted_learning_style']])

# Count of each learning style
style_counts = df['predicted_learning_style'].value_counts()
print("\nNumber of Students by Learning Style:")
```

[18]

NISHCHAYAGARWAL_202401100300163 (3).ipynb ×

C: > Users > Dell > Downloads > NISHCHAYAGARWAL_202401100300163 (3).ipynb > # Import necessary libraries

Generate + Code + Markdown | Run All Clear All Outputs | Outline ···

```python
print(style_counts)

# Set style for plots
sns.set(style="whitegrid")

# 1. Bar Chart
plt.figure(figsize=(8, 5))
style_counts.plot(kind='bar', color=['skyblue', 'lightgreen', 'salmon'])
plt.title('1. Learning Style Distribution (Bar Chart)')
plt.xlabel('Learning Style')
plt.ylabel('Number of Students')
plt.xticks(rotation=0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# 2. Pie Chart
plt.figure(figsize=(6, 6))
style_counts.plot(kind='pie', autopct='%1.1f%%', colors=['skyblue', 'lightgreen', 'salmon'])
plt.title('2. Learning Style Distribution (Pie Chart)')
plt.ylabel('')
plt.tight_layout()
plt.show()

# Confusion Matrix and Evaluation Metrics
# Assuming 'predicted_learning_style' is the prediction and 'learning_style' is the actual value
cm = confusion_matrix(df['learning_style'], df['predicted_learning_style'])
accuracy = accuracy_score(df['learning_style'], df['predicted_learning_style'])
precision = precision_score(df['learning_style'], df['predicted_learning_style'], average='weighted')
recall = recall_score(df['learning_style'], df['predicted_learning_style'], average='weighted')

# Display Confusion Matrix with Heatmap
plt.figure(figsize=(8, 6))
```

[18]

NISHCHAYAGARWAL_202401100300163 (3).ipynb

C: > Users > Dell > Downloads > NISHCHAYAGARWAL_202401100300163 (3).ipynb > # Import necessary libraries

Generate + Code + Markdown | Run All Clear All Outputs | Outline ...

```python
# Display Confusion Matrix with Heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=style_counts.index, yticklabels=style_counts.index)
plt.title('Confusion Matrix Heatmap')
plt.xlabel('Predicted Learning Style')
plt.ylabel('Actual Learning Style')
plt.tight_layout()
plt.show()

# Print Evaluation Metrics
print(f"\nEvaluation Metrics:")
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision (Weighted): {precision:.2f}")
print(f"Recall (Weighted): {recall:.2f}")

# Clustering (KMeans) for Segmentation
X = df[['visual_score', 'auditory_score', 'kinesthetic_score']]

# Use PCA for dimensionality reduction to 2D for visualization
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

# Apply KMeans Clustering
kmeans = KMeans(n_clusters=3, random_state=42)
df['cluster'] = kmeans.fit_predict(X)

# (Removed) Box Plot, Pair Plot, and Swarm Plot
```

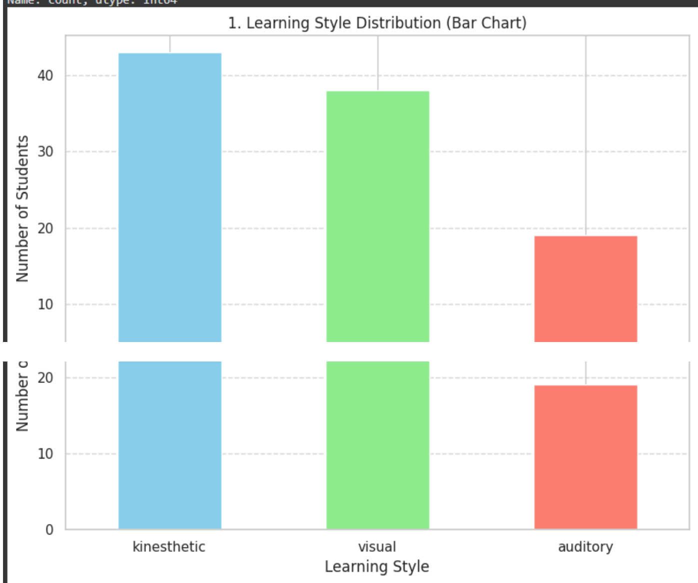
[18]

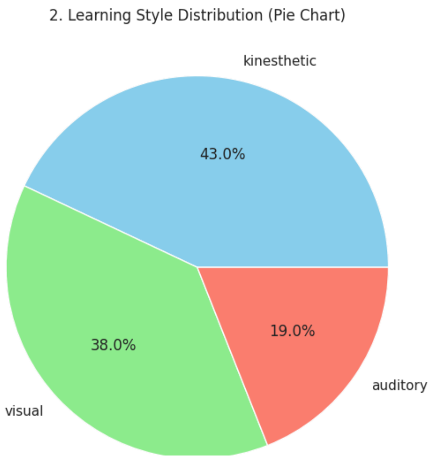
```
Classified Students:
    visual_score  auditory_score  kinesthetic_score predicted_learning_style
```

OUTPUT/RESULT

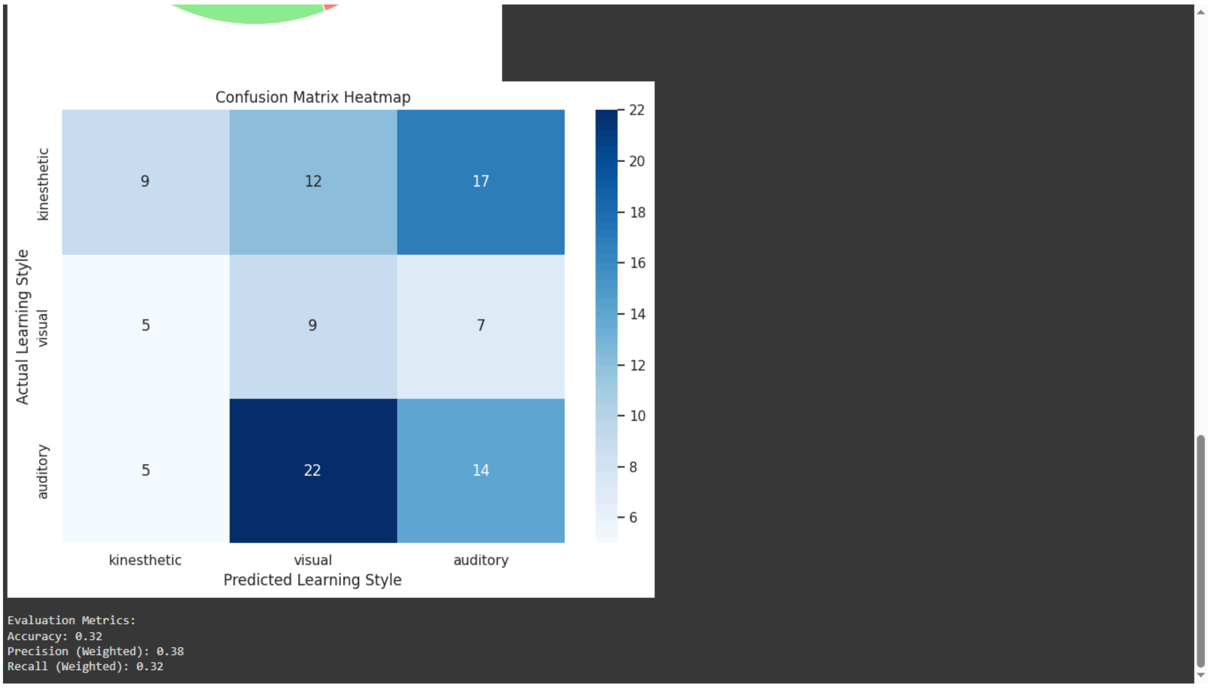
Bar Chart and Pie Chart

```
Classified Students:
    visual_score  auditory_score  kinesthetic_score predicted_learning_style
0       8.000301        1.389837           9.686887              kinesthetic
1       8.401052        7.294055           4.853655                   visual
2       9.124874        3.975049           6.688173                   visual
3       5.724100        7.702631           7.535001                 auditory
4       5.060739        4.711628           4.302653                   visual
..           ...             ...                ...                      ...
95      3.021422        2.362341           2.190592                   visual
96      1.622382        2.366976           9.798310              kinesthetic
97      8.447489        3.957023           6.243658                   visual
98      3.290865        1.490194           9.959267              kinesthetic
99      5.257248        2.642122           9.203633              kinesthetic

[100 rows x 4 columns]

Number of Students by Learning Style:
predicted_learning_style
kinesthetic    43
visual         38
auditory       19
Name: count, dtype: int64
```





Confusion Matrix Heatmap with Evaluation Metrics



References

1. **Pandas Documentation**
   Pandas: Powerful Python Data Analysis Toolkit.
   Available at: https://pandas.pydata.org/pandas-docs/stable/

2. **Matplotlib Documentation**
   Matplotlib: Python Plotting for Everyone.
   Available at: https://matplotlib.org/stable/users/index.html

3. **Seaborn Documentation**
   Seaborn: Statistical Data Visualization.
   Available at: https://seaborn.pydata.org/

4. **Scikit-learn Documentation**
   Scikit-learn: Machine Learning in Python.
   Available at: https://scikit-learn.org/stable/

5. **KMeans Clustering**
   Scikit-learn: KMeans Algorithm.
   Available at:
   https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

6. **Principal Component Analysis (PCA)**
   Principal Component Analysis for Dimensionality Reduction.
   Available at:
   https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

7. **Evaluation Metrics in Machine Learning**
   Scikit-learn: Metrics for Classification.
   Available at:
   https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics

8. **Confusion Matrix Visualization**
   Seaborn: Heatmap for Confusion Matrix.
   Available at: https://seaborn.pydata.org/generated/seaborn.heatmap.html

9. **Clustering in Machine Learning**
   KMeans Clustering Algorithm – A Comprehensive Guide.
   Available at: