**Team Project IV: Winner-Take-All Learning**

1. **Introduction**

This report explores the application of the Winner-Take-All (WTA) learning algorithm to the unsupervised clustering of the Iris dataset. The WTA algorithm, insipred by competitive learning in biological neural networks, updates only the “winning” unit for each input based on similarity, making it suitable for clustering tasks without the use of labels.

The Iris dataset consists of 150 flower samples from three species (Iris-setosa, Iris-versicolor, Iris-virginica), each represented by four numerical features. The goal of this project is to cluster the data into three groups using WTA and evaluate how well the learned clusters align with the actual species categories.

The implementation was done in Python, incorporating feature normalization, principal component analysis (PCA) for visualization, and evaluation metrics including clustering accuracy and per-class precision. The performance of WTA is analyzed using confusion matrices and plotted results to understand the separation between classes.

1. **Programming Language and Environment**

The implementation of this project was carried out in **Python**, utilizing a **Jupyter Notebook** environment through **Visual Studio Code** with the Jupyter extension. This environment provided an efficient workflow for combining code development, debugging, and result visualization.

The following Python libraries were used throughout the project:

* **NumPy** – for numerical operations, data normalization, and matrix computations.
* **Matplotlib** – for visualizing the clustering results in 2D space using PCA projections.
* **scikit-learn (sklearn)** – for applying Principal Component Analysis (PCA) for dimensionality reduction.
* **Pandas** – for organizing predicted and true labels into confusion matrices and tabular evaluation.
* **SciPy** – for solving the optimal label mapping problem using the Hungarian algorithm in clustering evaluation.

This setup enabled clear, modular implementation of the Winner-Take-All learning process, as well as comprehensive performance analysis of the clustering results.

1. **Task Description**

The main objective of this project is to implement and analyze the Winner-Take-All (WTA) competitive learning algorithm for unsupervised clustering. The algorithm is applied to two datasets:

* **Example 2 (Sample Dataset):**

The provided 2D example dataset is used to verify the correctness of the WTA implementation. The clustering behavior is observed visually to ensure that the winner neuron updates its weights appropriately based on the input pattern distribution.

* **Iris Dataset (UCI Machine Learning Repository):**

The classic Iris dataset is used to test the WTA algorithm in a real-world clustering scenario. Only the four numeric features (sepal length, sepal width, petal length, and petal width) are used for clustering, while the class labels (Setosa, Versicolor, Virginica) are reserved for evaluation.

Three clusters are used to match the natural grouping of the dataset. After training, the predicted cluster assignments are compared with the true labels using confusion matrices, cluster-to-class alignment, overall accuracy, and per-class precision.

This project aims to explore the effectiveness of unsupervised learning using a simple WTA model and assess its ability to discover underlying class structure without using label information during training.

1. **Code Structure and Functions**

Key functions and logic in the implementation:

* **load\_iris\_data(path):**

Loads the Iris dataset from a file, separates the feature vectors and class labels, and applies Z-score normalization to the features. This ensures that all input features have mean 0 and standard deviation 1, improving the stability of training.

* **label\_to\_numeric(labels):**

Converts string class labels (‘Iris-setosa’, ‘Iris-versicolor’, ‘Iris-virginica’) into numerical values (0, 1, 2), allowing easier evaluation of clustering results against ground truth labels.

* **winner\_take\_all\_learning(x, clus, alpha, n\_update):**

Implements the core Winner-Take-All (WTA) learning algorithm:

* + Initializes the weights randomly by choosing a few input vectors.
  + Iteratively presents each training pattern, selects the “winner” neuron with the highest dot product, and updates its weights toward the input pattern.
  + Normalizes the weights to prevent divergence.
  + Repeats this process for a fixed number of iterations (n\_update).
* **classify(x, w):**

Uses the trained weight vectors to assign each input to a cluster. The input is compared to each cluster center using dot product, and the closest (winner) is chosen.

* **plot\_clusters(x, labels, title):**

Projects the input features into 2D using PCA for visualization and plots the data points colored by their assigned cluster (or true label), providing visual insight into clustering performance.

* **evaluate\_clustering(true\_labels, predicted\_clusters):**

Aligns predicted clusters with true labels using majority voting, and computes:

* + Overall clustering accuracy
  + Precision for each true class
  + Cluster confusion table
  + Optionally visualizes the confusion matrix for easier interpretation
* **main program logic:**
  + Loads and normalizes the Iris dataset
  + Trains the WTA network with 3 clusters
  + Assigns cluster labels and aligns them with the true classes
  + Visualizes the clustering result and ground truth
  + Prints confusion matrix, accuracy, and per-class presicion

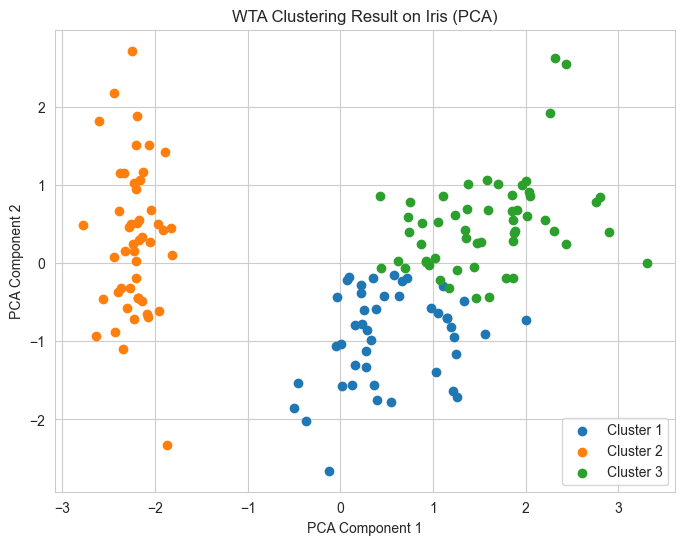
This structure supports modular analysis and easy replacement of the dataset or learning parameters for further experimentation.

1. **Visual Analysis**

To better understand the clustering behavior of the WTA algorithm on the Iris dataset, three visualizations were generated:

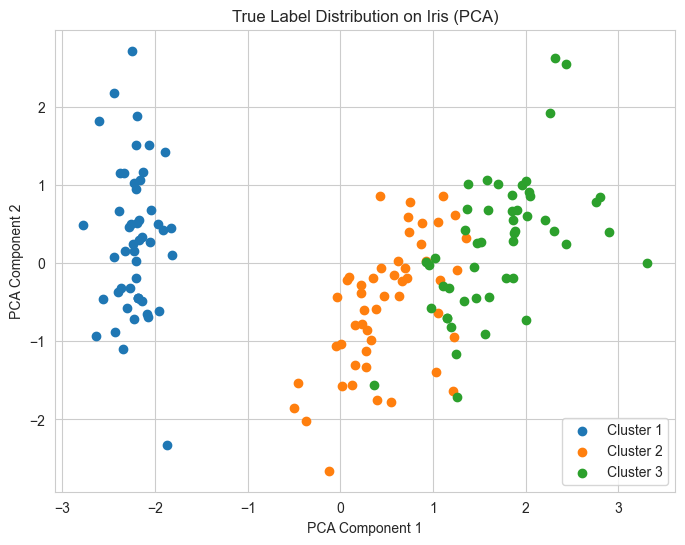
* **WTA Clustering Result on Iris (PCA):**

This plot shows the cluster assignments produced by the trained WTA network. Principal Component Analysis (PCA) was applied to project the original 4-dimensional input features onto a 2D plane for visualization. Data points are colored based on the cluster they were assigned to. The figure demonstrates how the unsupervised WTA algorithm separated the data into three distinct clusters, with varying degrees of overlap.



* **True Label Distribution on Iris (PCA):**

For comparison, this plot shows the true class labels of the Iris dataset (Setosa, Versicolor, Virginica) projected onto the same 2D PCA space. The Setosa class is linearly separable from the other two, which supports the observation that WTA often correctly clusters this class. However, the Versicolor and Virginica classes are less separable, leading to higher confusion between them.

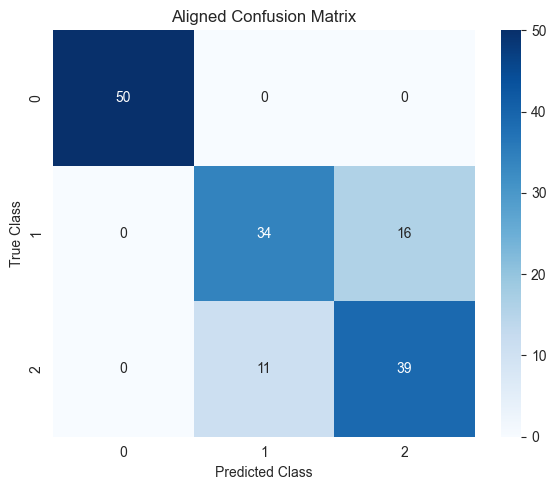


* **Aligned Confusion Matrix:**

To quantitatively assess clustering quality, a confusion matrix was constructed by aligning predicted clusters with true class labels using majority voting. The confusion matrix shows that:

* + Setosa was perfectly clustered.
  + Most misclassifications occured between Versicolor and Virginica, which is expected due to their overlapping feature distributions.

The matrix visualization provides a clear summary of classification alignment and misalignment, and supports the numerical evaluation results such as overall accuracy and per-class precision.

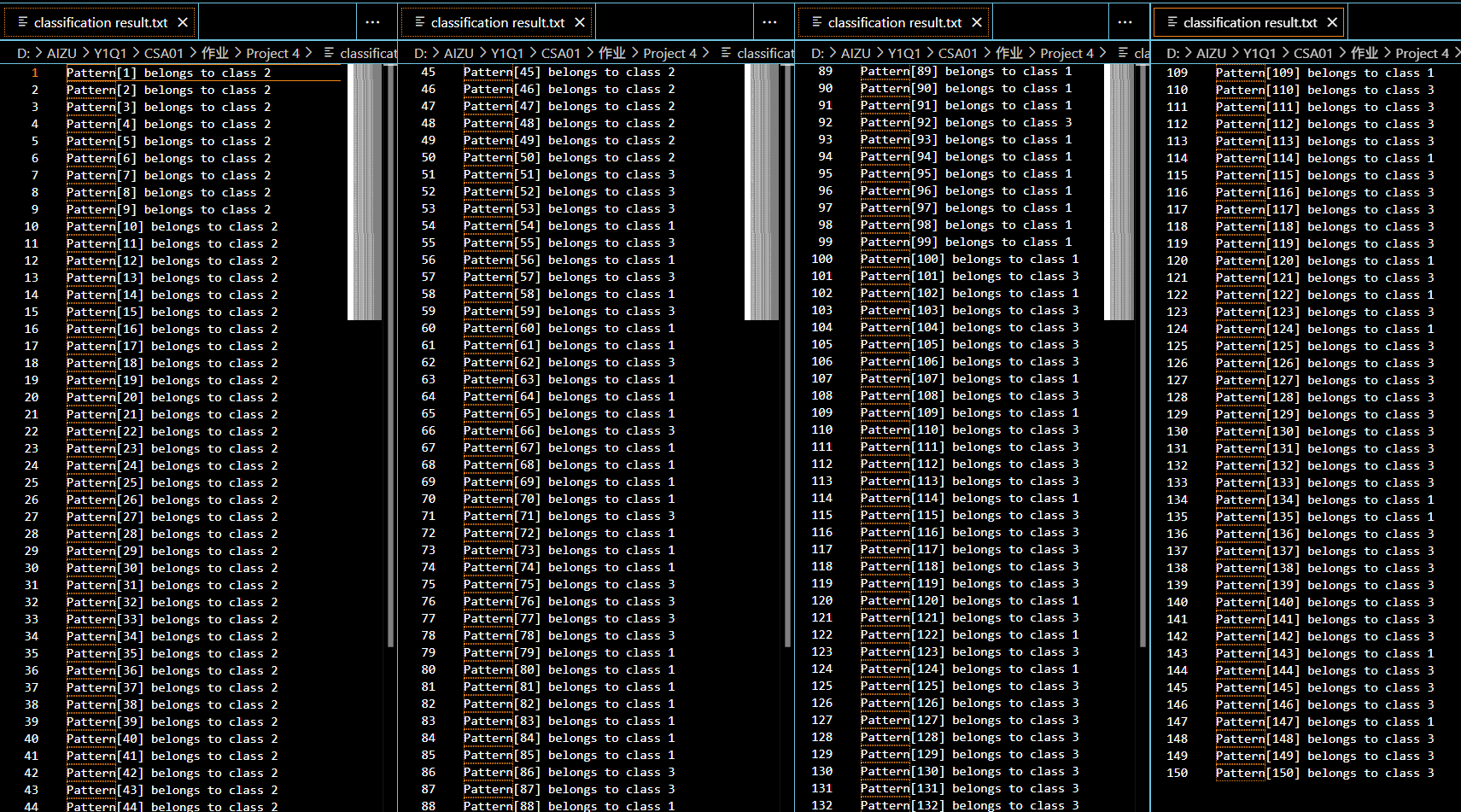


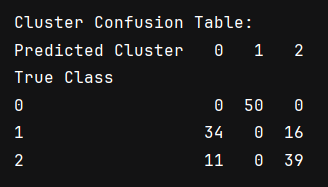
These visual tools complement the numerical metrics and help in interpreting how well the WTA clustering approximates the underlying class structure in the Iris dataset.

1. **Results and Discussion**

The WTA (Winner-Take-All) neural network was applied to the Iris dataset with the number of clusters set to 3, corresponding to the three species: Setosa, Versicolor, and Virginica. After 100 training iterations with a learning rate of 0.1, the model assigned each input pattern to one of the three output neurons based on the highest activation.

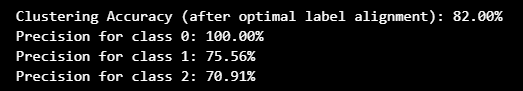
After clustering, predicted labels were aligned to the true class labels using the Hungarian algorithm to maximize overall accuracy. The final cluster confusion table is shown below:





**From this table, several key insights can be drawn:**

* **Setosa (class 0)** was perfectly clustered, with all 50 samples correctly grouped into a single cluster. This demonstrates that Setosa’s features are well-separated from the other two classes, which is consistent with PCA visualization.
* **Versicolor (class 1)** and **Virginica (class 2)** exhibited significant overlap. Many Versicolor samples were incorrectly clustered as Virginica and vice versa, suggesting difficulty in separating these two classes using unsupervised WTA learning.
* The **overall clustering accuracy** after label alignment was 82%, which is relatively high for an unsupervised approach.
* **Per-class precision** was:
  + **Class 0 (Setosa): 100%**
  + **Class 1 (Versicolor): 75.56%**
  + **Class 2 (Virginica): 70.91%**



These results confirm that while WTA clustering performs excellently for linearly separable data (like Setosa), it struggles with classes that are more similar in feature space. The overlap between Versicolor and Virginica, often observed in unsupervised tasks, results in reduced clustering precision for those classes.

1. **Conclusion**

This project investigated the application of a Winner-Take-All (WTA) neural network for unsupervised clustering on the Iris dataset. The implementation successfully clustered the data into three groups corresponding to the three flower species. Visual and quantitative evaluations showed that the WTA network effectively captured class separability in cases where the data distribution was distinct.

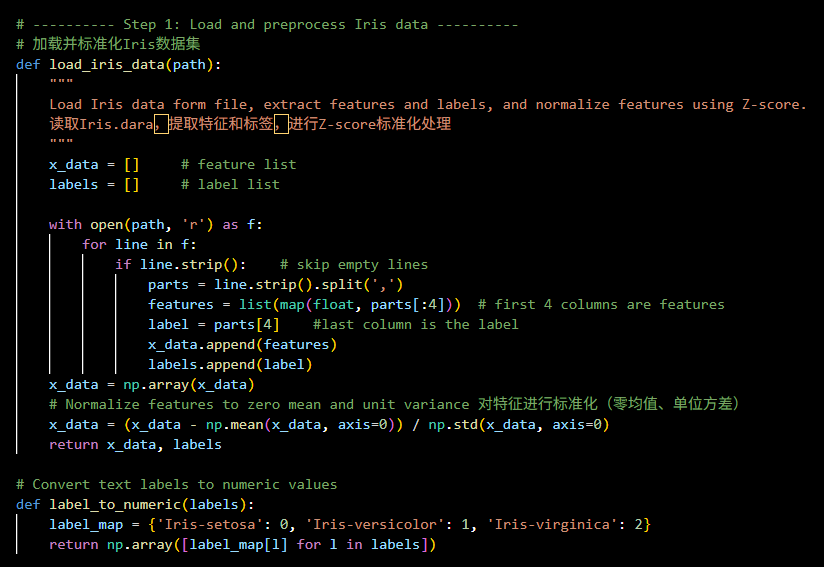
The clustering results demonstrated that Setosa, being linearly separable from the other two classes, was identified with perfect precision. However, Versicolor and Virginica whose features overlap significantly were more difficult to distinguish without supervision, leading to moderate precision scores for those classes.

With overall clustering accuracy of 82% after optimal label alignment, the experiment confirmed that WTA networks are suitable for discovering structure in relatively clean datasets, though their performance is sensitive to class similarity and feature representation.

1. **Appendix: Code Screenshots**

**Figure A1. Data Loading and Normalization**

This section includes load\_iris\_data() and label\_to\_numeric() functions. It reads the iris.data file, extracts the first four columns as features and the last as labels, then standardizes the features using Z-score normalization. Labels are converted to numeric form (0, 1, 2).



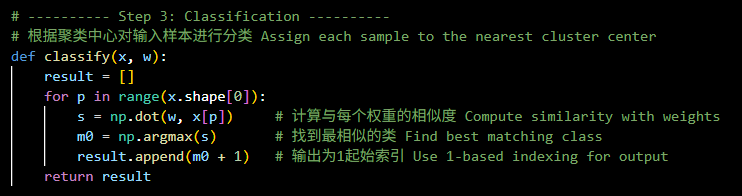
**Figure A2. Winner-Take-All Learning Algorithm**

The winner\_take\_all\_learning() function performs unsupervised clustering using the WTA rule. It randomly initializes cluster weights, iteratively updates the closest (winning) weight vector for each input, and normalizes weights after each update.



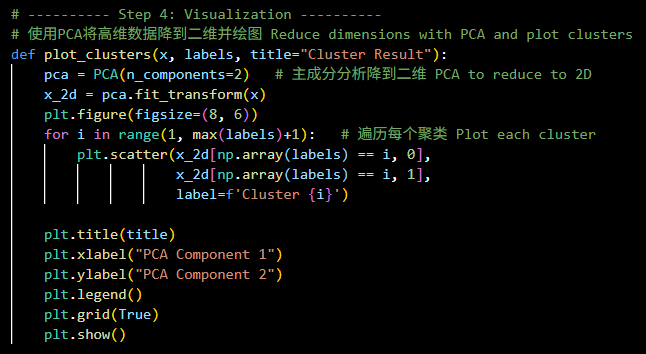
**Figure A3. Pattern Classification by Cluster Centers**

The classify() function assigns each input sample to its closest cluster center (highest dot product similarity). Cluster labels are returned using 1-based indexing for clarity.



**Figure A4. PCA-Based Cluster Visualization**

The plot\_clusters() function uses Principal Component Analysis (PCA) to reduce the 4D input features to 2D for visualization. It then displays the predicted clusters or true labels in a scatter plot.



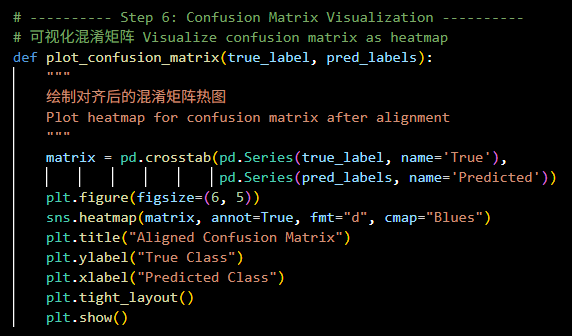
**Figure A5. Evaluation via Confusion Matrix and Metrics**

The evaluate\_clustering() function computes a confusion matrix between predicted and true labels. It applies the Hungarian algorithm (via linear\_sum\_assignment) to align predicted clusters with actual classes optimally. Outputs include overall clustering accuracy and per-class precision.



**Figure A6. Aligned Confusion Matrix Visualization**

The plot\_confusion\_matrix() function visualizes the aligned confusion matrix as a heatmap using Seaborn. This helps illustrate the correnpondence between predicted and actual class distributions.



**Figure A7. Main Execution**

The main code section loads the data, trains the WTA network, classifies the input patterns, displays predictions, plots cluster and label distributions, evaluates accuracy and precision, and plots the final aligned confusion matrix.

