Pattern Recognition Homework Report

Name: **Vahid Maleki**
Student ID: **40313004**

This report presents the analysis performed for the pattern recognition homework. Parts 1-3 analyze the 'seeds.xlsx' dataset, and Part 4 demonstrates the Self-Organizing Map (SOM) technique using the 'iris' dataset.

Part 1: Dataset Loading and Feature-Analysis

1. Dataset Dimensions

The seeds.xlsx dataset was loaded. Its dimensions are **(210, 8)**, representing 210 samples and 8 columns (7 features + 1 class label).

2. Features

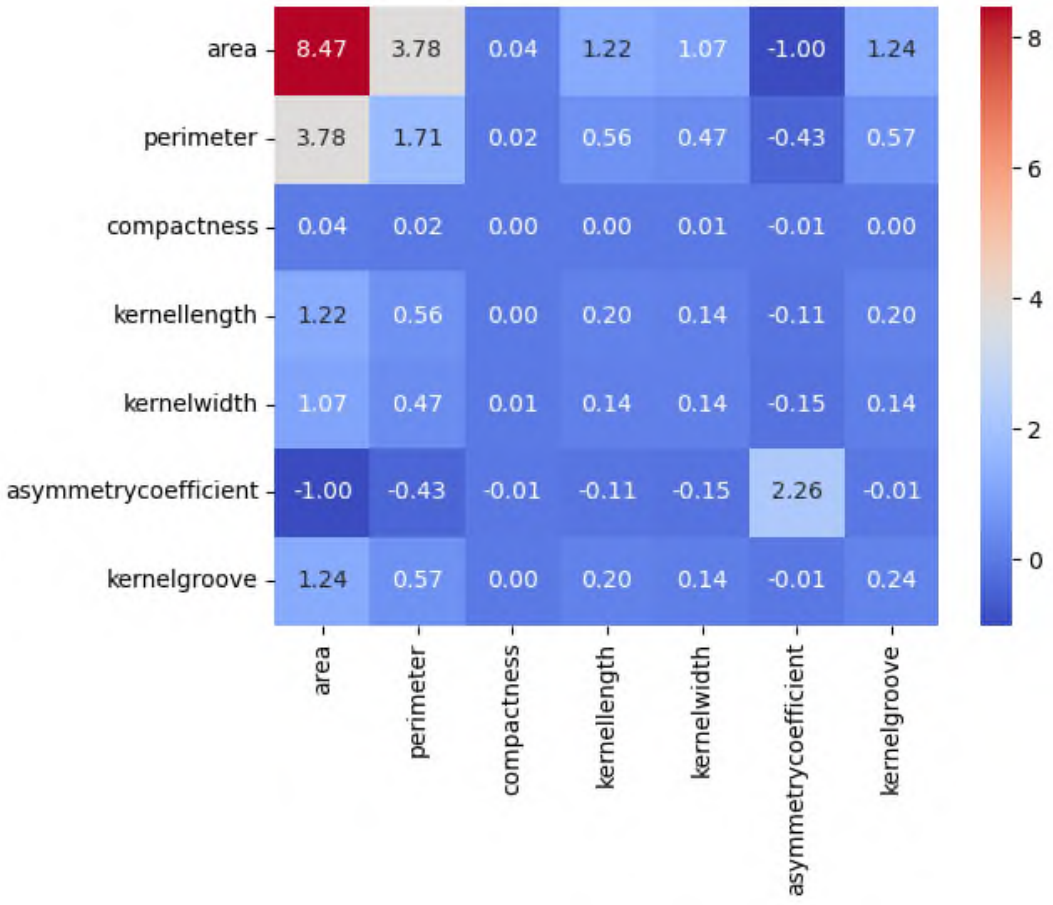
The class column was separated from the data matrix. The dataset contains **7 features**. The names of these features are:

- area
- perimeter
- compactness
- kernellength
- kernelwidth
- asymmetrycoefficient
- kernelgroove

Part 2: Principal Component Analysis (PCA)

3. Covariance Matrix

The covariance matrix for the 7 features was computed and visualized as a heatmap.



4. Eigenvalues and Eigenvectors

The eigenvalues and eigenvectors of the covariance matrix were computed:

Eigenvalues:

[10.7933269 , 2.12945512, 0.07363003, 0.01288749, 0.00274823,
0.00157045, 0.00002966]

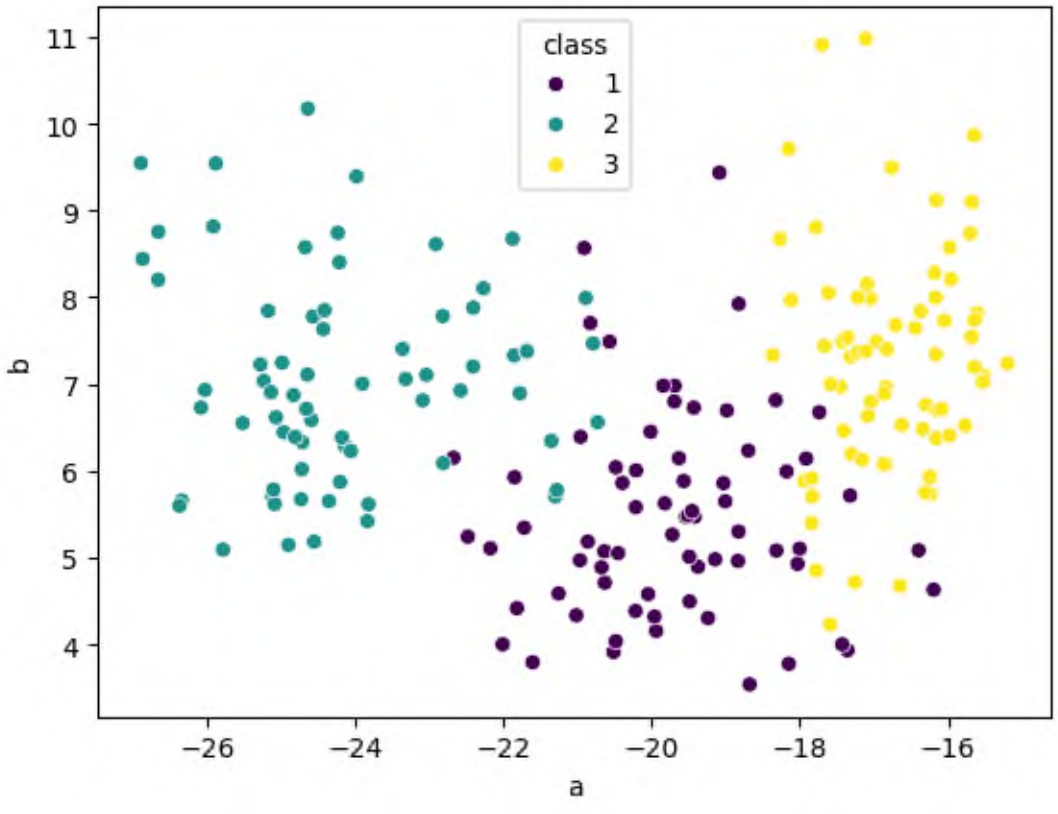
Eigenvectors:

[[-0.8842285 , 0.10080577, -0.26453354, -0.19944949, -0.13717297, 0.28063956, -0.02539824],

[-0.39540542, 0.05648963, 0.28251995, 0.57881686, 0.57475603, -0.30155864, 0.0658399],
[-0.00431132, -0.00289474, -0.05903584, -0.05776023, -0.05310454, -0.04522905, 0.99412565],
[-0.12854448, 0.03062173, 0.40014946, 0.43610024, -0.78699776, -0.11343761, 0.00143143],
[-0.11105914, 0.00237229, -0.31923869, -0.23416358, -0.1448029 , -0.89626785, -0.0815499],
[0.12761562, 0.98941048, -0.06429754, 0.02514736, -0.00157564, 0.003288 , 0.00114269],
[-0.1289665 , 0.08223339, 0.76193973, -0.61335659, 0.08765361, -0.10992364, 0.00897193]]

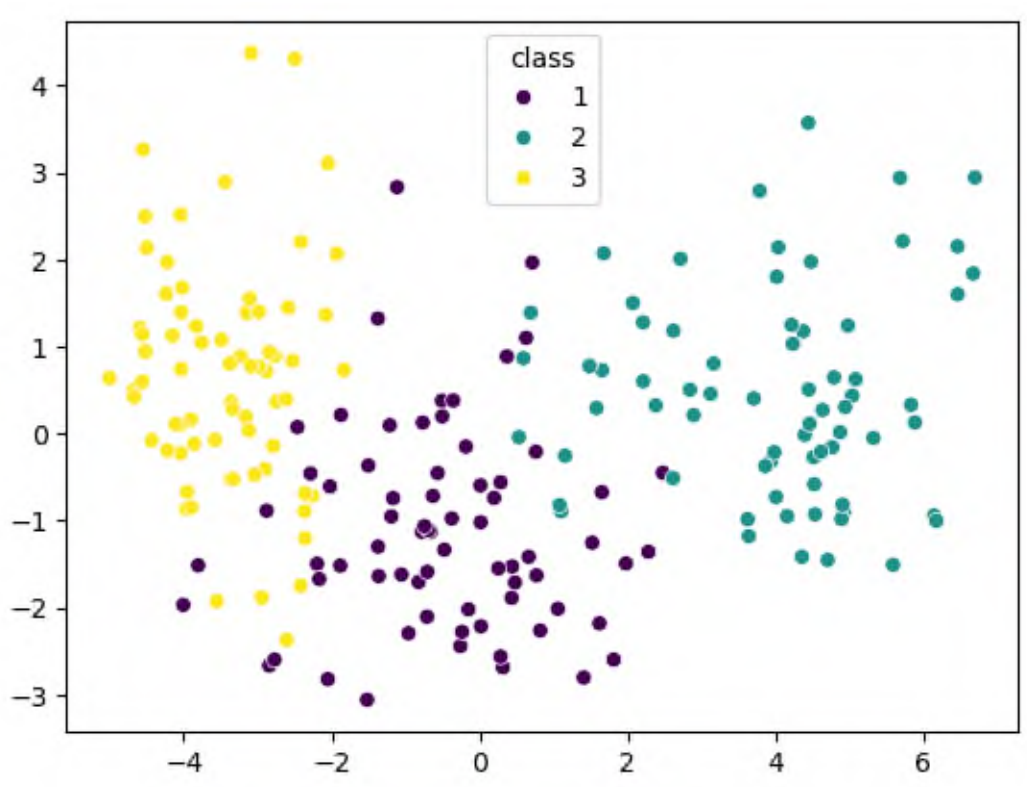
5. Manual PCA

PCA was performed manually by selecting the two eigenvectors corresponding to the two largest eigenvalues (10.793 and 2.129). The original 7-dimensional data was projected onto this new 2D space.



6. Manual PCA Plot

The resulting 2D data from the manual PCA is plotted below. Different colors represent the three different classes (1, 2, 3).



7. Scikit-learn PCA and Comparison

PCA was performed again using the scikit-learn library, reducing the dimensionality to 2.

Comparison: The scikit-learn PCA plot (pca_sklearn.png) shows the data centered around (0,0). The manual PCA plot (pca_manual.png) shows a similar relative distribution of the classes, but the entire plot is shifted. This difference is because the scikit-learn PCA function automatically centers the data (subtracts the mean) before projection, which is a standard step. The manual implementation performed did not include this data-centering step.

8. Error Calculation

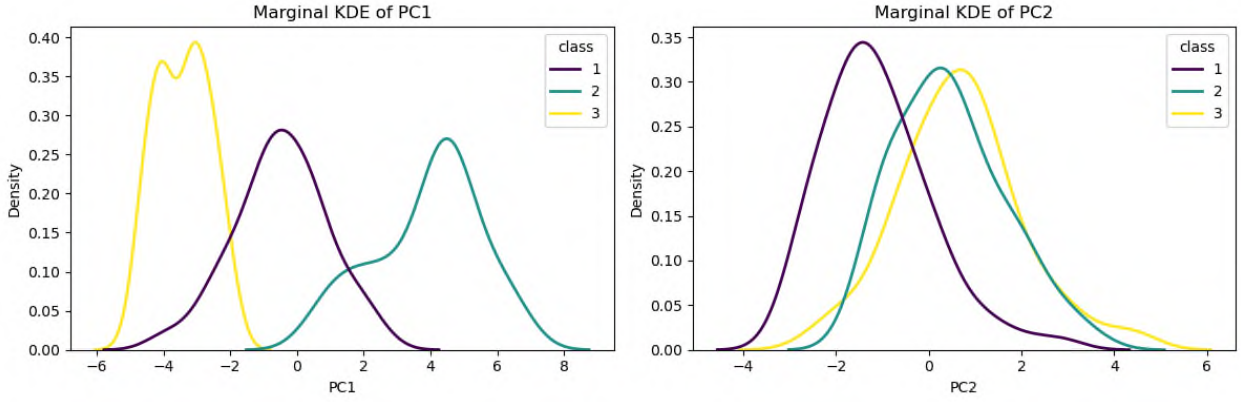
The sum of all eigenvalues and the sum of the two retained (largest) eigenvalues were computed to find the relative mean squared error.

- **Sum of all eigenvalues:** 13.01364789558829
- **Sum of selected eigenvalues (top 2):** 12.922782035961518
- **Error:** $1 - \frac{12.922782035961518}{13.01364789558829} = \mathbf{0.006982351171309631}$

This very low error (approx 0.7%) indicates that the first two principal components capture over 99% of the variance in the data.

9. Marginal Densities

The marginal density distributions (Kernel Density Estimation) for each class along the two principal components (PC1 and PC2) from the scikit-learn PCA are plotted.



Part 3: Gaussian Distribution Analysis in 2D PCA Space

10. Parameter Estimation

The mean vector and covariance matrix were estimated for all samples combined and for each class individually within the 2D PCA space (using the scikit-learn PCA result).

Total Samples:

- **Mean:** [-5.41365894e-15 -7.10542736e-16]
- **Covariance Matrix:**
  [[10.7933269 , 1.58383753e-14],
  [1.58383753e-14, 2.12945512]]

Class 1:

- **Mean:** [-0.48505414, -1.11868774]
- **Covariance Matrix:**
  [[1.91090387, -0.01015799],
  [-0.01015799, 1.36098387]]

Class 2:

- **Mean:** [3.90583319, 0.45295876]
- **Covariance Matrix:**
  [[2.54870834, 0.07262228],
  [0.07262228, 1.38758624]]

Class 3:

- **Mean:** [-3.42077905, 0.66572899]
- **Covariance Matrix:**
  [[0.64658404, -0.09745791],
  [-0.09745791, 1.77415583]]

11. Inverse Covariance Matrices

The inverse of each class's covariance matrix was computed:

- **Class 1 Inverse Covariance:**
  [[0.52333333, 0.00390601],
  [0.00390601, 0.73479172]]

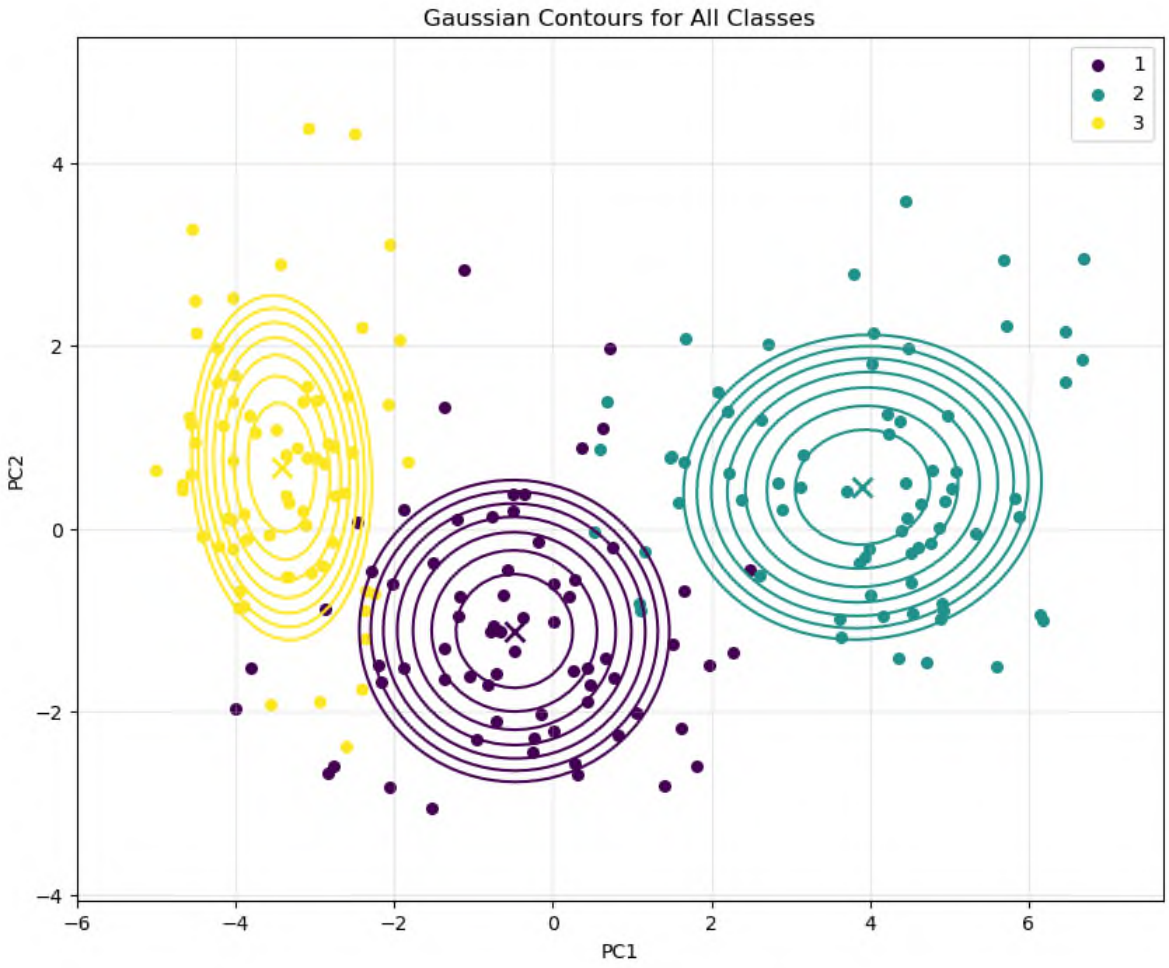
- **Class 2 Inverse Covariance:**

[[0.39294159, -0.02056543],
 [-0.02056543, 0.72175226]]

- **Class 3 Inverse Covariance:**
  [[1.55950166, 0.08566653],
   [0.08566653, 0.56835418]]

12. Gaussian Contour Plots

Assuming a Gaussian distribution for each class, contour lines (for k-values from 0 to 1) were plotted using the estimated parameters. The scatter plot of the data is overlaid, with 'x' markers indicating the mean of each class.



Part 4: Self-Organizing Map (SOM)

Note: This part of the analysis was performed on the 'iris' dataset.

13. SOM Explanation

A Self-Organizing Map (SOM) is an unsupervised neural network technique used for dimensionality reduction and clustering. It works by mapping high-dimensional input data onto a lower-dimensional (typically 2D) grid of 'neurons' or 'nodes'.

During training, each data point is presented to the network, and the 'winning' neuron (the one whose weight vector is most similar to the input data point, called the Best Matching Unit or BMU) is identified. The BMU's weights, as well as the weights of its neighboring neurons on the grid, are updated to become more similar to the input data point. This process "pulls" the weights of nearby neurons towards the data, preserving the topological relationships of the original data. Over time, the grid organizes itself to reflect the structure of the input data.

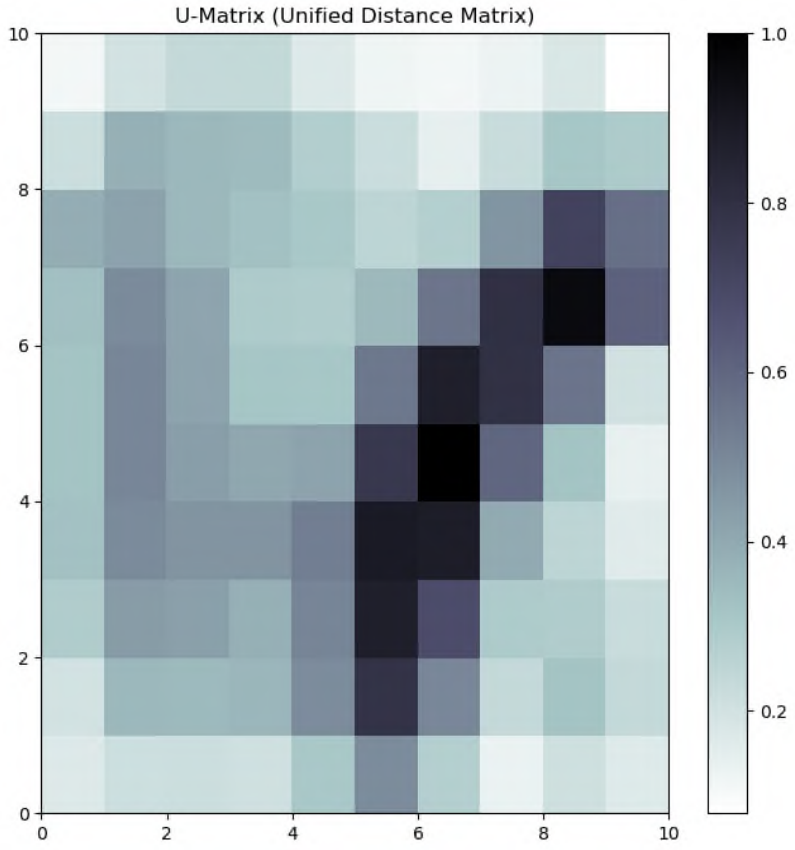
For dimensionality reduction, the 2D coordinates of the BMU for a given high-dimensional input sample serve as its new, lower-dimensional representation.

14. SOM Training

A 10x10 rectangular SOM was initialized and trained on the scaled 'iris' dataset for 500 iterations.

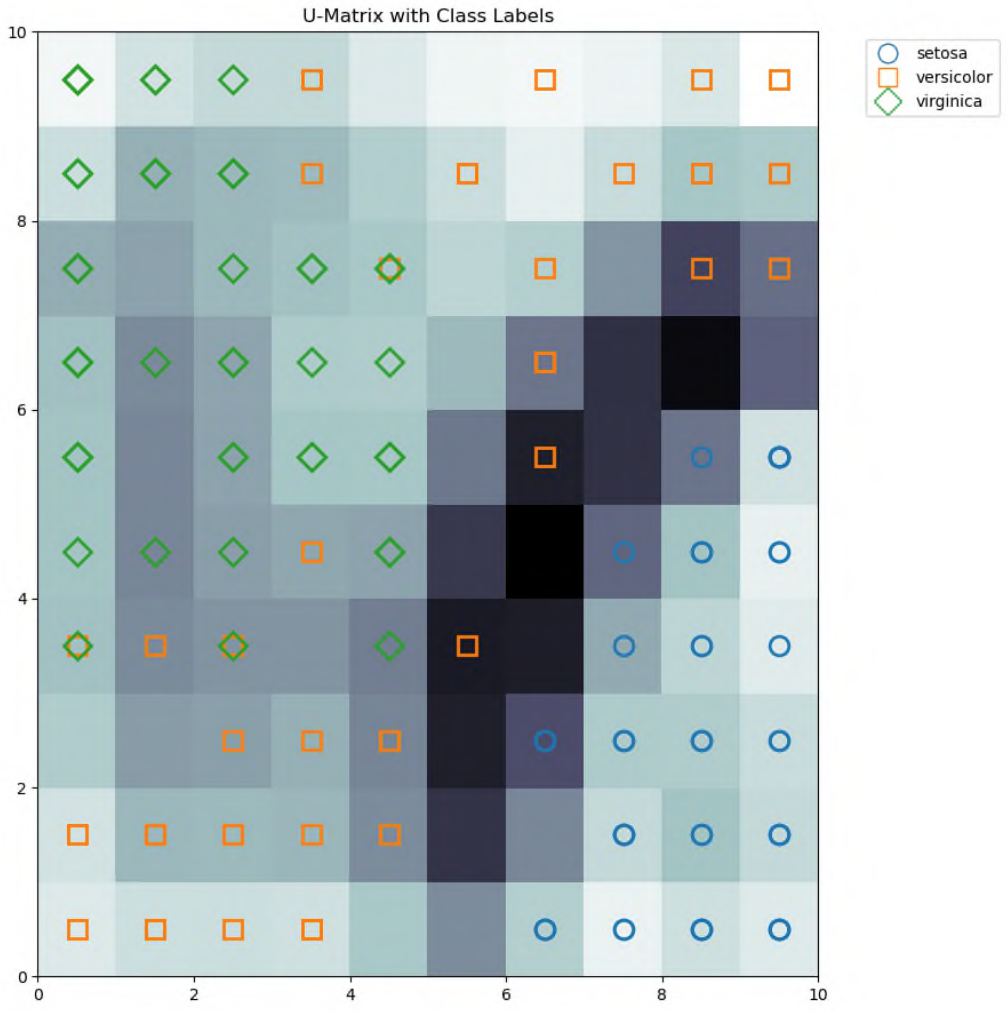
15. U-Matrix

The U-Matrix (Unified Distance Matrix) was plotted. It visualizes the average distance between each neuron and its neighbors. Darker colors indicate larger distances, representing potential cluster boundaries, while lighter areas indicate regions where neurons are similar, representing potential clusters. The dark "wall" in the plot suggests a strong separation between groups of neurons.



16. U-Matrix with Class Labels

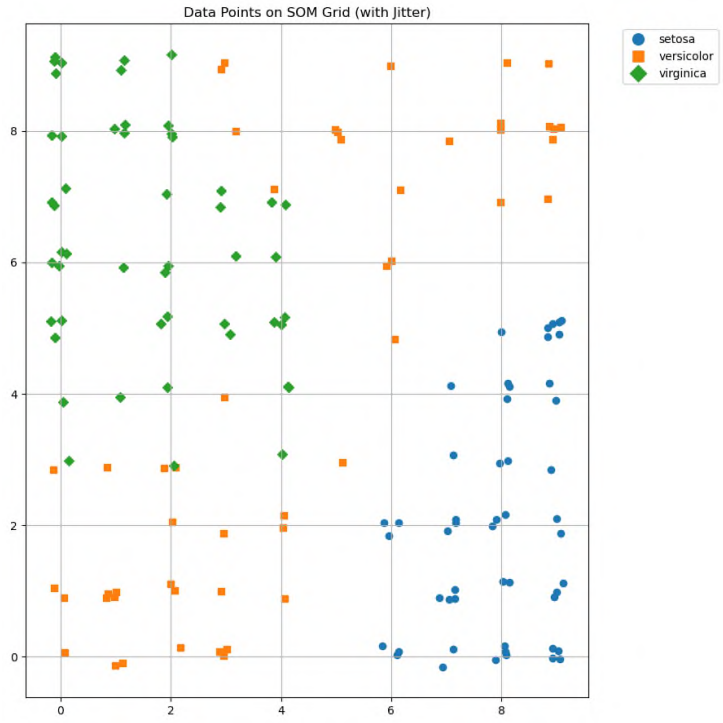
The U-Matrix is plotted again, this time with the data points from the 'iris' dataset mapped to their corresponding BMUs. The class of each data point (setosa, versicolor, virginica) is shown by a different marker.



Interpretation: The plot shows a clear separation of the classes on the map. The 'setosa' class (blue circles) has clustered in the bottom-right corner, which is a light-colored (low distance) area, indicating high similarity within that cluster. This region is also separated from the other classes by a dark (high distance) 'wall'. The 'versicolor' (orange squares) and 'virginica' (green diamonds) classes are less
clearly separated from each other, occupying the upper-left and middle-left regions, but they are distinctly separated from 'setosa'.

17. SOM Grid Scatter Plot

This plot shows the mapping of each data sample to its winning neuron on the 10x10 grid. A small amount of 'jitter' (random noise) is added to the coordinates to prevent overplotting and make individual points visible.



Interpretation: This plot confirms the findings from the U-Matrix with labels. The 'setosa' class (blue circles) is tightly clustered in the bottom-right. 'Versicolor' (orange squares) and 'virginica' (green diamonds) occupy distinct, though adjacent, regions of the map. This visualization clearly shows how the SOM has learned to topographically organize the data, mapping similar high-dimensional data points to nearby locations on its 2D grid.

Winning neuron coordinates for the first 10 samples:

Sample 0 (Class 0): (7, 1)
Sample 1 (Class 0): (9, 5)
Sample 2 (Class 0): (9, 3)
Sample 3 (Class 0): (7, 4)
Sample 4 (Class 0): (7, 1)
Sample 5 (Class 0): (8, 0)
Sample 6 (Class 0): (8, 2)
Sample 7 (Class 0): (7, 2)
Sample 8 (Class 0): (8, 4)
Sample 9 (Class 0): (9, 5)