Group 10, ECE 236A Final Project

Radha Lahoti
Department of Mechanical and Aerospace Engineering
UID: 005951565

Yongkyu Lee
Department of Mechanical and Aerospace Engineering
UID: 305713002

Kajal Sharma
Department of Computer Science and Engineering
UID: 906297199

Hanbyeol Yoon
Department of Mechanical and Aerospace Engineering
UID: 205528260

I. Task 1

To formulate Linear Programming (LP) for multi-class classification, we firstly define classifier variables for each cluster $j = 1, \cdots, k$: weight w_j and bias b_j, and construct the indicator constant y_{ij} based on the labeled train dataset as follows:

$$y_{ij} = \begin{cases} 1 & \text{if data } i \text{ belongs to cluster } j \\ 0 & \text{otherwise} \end{cases}$$

We design the classifier using the one-versus-all method [1]. The basic idea is to define a hyperplane with w_j and b_j to separate one cluster from all the others. To enforce this through an optimization problem, we formulate the following inequalities:

$$\begin{aligned} & w_j^T x_i + b_j \geq 1 - \alpha(1 - y_{ij}) \\ & w_j^T x_i + b_j \leq -1 + \alpha y_{ij} \\ & \text{for all } i = 1, \ldots, n, \text{ and } j = 1, \ldots, k \end{aligned}$$

where n is the number of data points and k is the number of clusters. $\alpha \in \mathbb{R}$ is a large positive constant to nullify the first inequality for the data points not in the cluster and the second for the points in the cluster.

To formulate the optimization problem determining w_j and b_j, we aim to minimize the penalty incurred for not adhering to the above inequalities:

$$\begin{aligned} \text{minimize } & \sum_{j=1}^{k} \sum_{i=1}^{n} \max\{0, 1 - \alpha(1 - y_{ij}) - (w_j^T x_i + b_j)\} \\ & + \max\{0, 1 - \alpha y_{ij} + (w_j^T x_i + b_j)\} \end{aligned}$$

where the optimization variables here are w_j and b_j, and the training data x_i and y_{ij} are given.

To translate the optimization problem into a linear program (LP), we introduce variables t_{ij} and z_{ij} to replace the max operations and formulate the LP as follows:

$$\begin{aligned} \text{minimize } & \sum_{j=1}^{k} \sum_{i=1}^{n} (t_{ij} + z_{ij}) \\ \text{subject to } & t_{ij} \geq 0, \quad z_{ij} \geq 0 \\ & t_{ij} \geq 1 - \alpha(1 - y_{ij}) - (w_j^T x_i + b_j) \\ & z_{ij} \geq 1 - \alpha y_{ij} + (w_j^T x_i + b_j) \\ & \text{for all } i = 1, \ldots, n, \text{ and } j = 1, \ldots, k \end{aligned}$$

As a result of the LP, we obtain w_j and b_j for each cluster $j = 1, \ldots, k$. We define $g()$ and $h()$ to classify data as follows:

$$g_j(x) = w_j^T x_i + b_j$$

$$h(g_j(x)) = \begin{cases} 1 & \text{if } g_j(x) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{for all } i = 1, \ldots, n, \text{ and } j = 1, \ldots, k$$

where x is classified into cluster j if $h(g_j(x))$ is 1. This approach holds when the dataset is perfectly separable.

To deal with generic cases, we use the max operation to classify given data. The following equation ensures a data point is classified into one cluster:

$$\text{predY} = \text{argmax}_j g_j(x)$$

II. Task 2

The outline of the proposed clustering algorithm is elaborated in the Algorithm 1 in Appendix: 1) intialization of centroids by random sampling, 2) calculation of distance of each point from centroids, 3) assignment of each data via LP, 4) update of centroids based on the new assignment, and 5) convergence check.

To formulate LP in the assignment step in the pseudo code for clustering, firstly, we define indicator variables x_{ij} to determine whether data point i belongs to cluster j as follow:

$$x_{ij} = \begin{cases} 1 & \text{if data } i \text{ belongs to cluster } j, \\ 0 & \text{otherwise.} \end{cases}$$

We choose the Euclidean distance as the distance metric for clustering. The distance of each point p_i to the centroid of cluster c_j can be obtained as follow:

$$d_{ij} = \|p_i - c_j\|_2 \quad \text{for } i = 1, \ldots, n, \text{ and } j = 1, \ldots, k$$

Now, the proposed ILP is as follows:

$$\begin{aligned}
\min \ & \sum_{i=1}^{n}\sum_{j=1}^{k} d_{ij}x_{ij} \\
\text{s.t.} \\
& \sum_{j=1}^{k} x_{ij} = 1 \quad \text{for } i = 1, \ldots, n \\
& \sum_{i=1}^{n} x_{ij} \geq 1 \quad \text{for } j = 1, \ldots, k \\
& x_{ij} \in \{0, 1\} \quad \text{for } i = 1, \ldots, n, \text{ and } j = 1, \ldots, k
\end{aligned}$$

where the objective function minimizes the sum of distances of each data point to its assigned cluster's centroid. The first constraint ensures each data belongs to one cluster, and the second constraint ensures each cluster includes at least one data point. The third constraint ensures the variables function as indicator variables.

To relax the above ILP to LP, we replace the last constraint with the following constraint, and approximate the ILP solution via max operation as follows:

$$0 \leq x_{ij} \leq 1 \quad \text{for } i = 1, \ldots, n, \text{ and } j = 1, \ldots, k.$$

$$x_{ij}^{approx} = \begin{cases} 1 & \text{if } j = \arg\max_k x_{ik} \quad \text{for } i = 1, \ldots, n, \\ 0 & \text{otherwise.} \end{cases}$$

We update the centroid of each cluster as the mean of the assigned data points as follow:

$$c_j = \frac{\sum_{i=1}^{n} p_i x_{ij}}{\sum_{i=1}^{n} x_{ij}}$$

For the convergence check, we examine whether the maximum Euclidean distance of the previous and updated centroids is within the threshold (10^{-6}) or whether it reaches the maximum iteration (10^6).

III. Task 3

A. Overview

The goal of this task is to identify L samples within the given dataset, that would maximize the accuracy of the classifier trained on these selected samples. Since the number of classes of our dataset K is given, we can also calculate K centroids from the dataset that will serve as our baseline. Our program requires a matrix $\bar{\mathbf{D}}$, where each column contains value of the distance to the closest centroid, and 0 everywhere else. Then, it selects an equal number of points from each of the K sets that lie furthest from the centroid, as shown in figure 7. By purposely selecting points at the decision boundary which are harder to classify, we can train a model with a more accurate boundary.

B. ILP Formulation

Given N data points and K classes, we can define the variables of our program as the following:

$$\mathbf{x} \in \mathbb{R}^{K \times N}.$$

x_{ij} is an indicator which holds value 1 if point j is selected from centroid i and 0 otherwise. The distance matrix is also calculated from the K centroids and the N data points:

$$\mathbf{D} \in \mathbb{R}^{K \times N},$$

where D_{ij} measures the Euclidean distance between centroid i and point j. Next, we can modify $\mathbf{D}$ to obtain $\bar{\mathbf{D}}$ which preserves the distance to the closest centroid for each point, and replace all other entries with zeros.

The following ILP selects a total of L points that lie closest to the K centroids and the constraints ensure no redudant selection and even selection of points from each centroid. Note that the constraint where the sum of each row is $\frac{L}{K}$ ensures that L points are selected. Since $\frac{L}{K}$ is required to be an integer, in practice, we round $\frac{L}{K}$ to the nearest integer.

$$\begin{aligned}
\max \quad & \text{tr}(\bar{\mathbf{D}}\mathbf{x}^T) \\
\text{s.t.} \\
& \sum_{i=1}^{K} x_{ij} \leq 1 \quad \forall \quad j \in \{1, 2, \cdots, N\} \\
& \sum_{j=1}^{N} x_{ij} = \frac{L}{K} \quad \forall \quad i \in \{1, 2, \cdots, K\} \\
& x_{ij} \in \{0, 1\}
\end{aligned}$$

C. LP Relaxation

The above ILP can be relaxed to an LP by defining the constraints for x_{ij} to lie in $[0, 1]$. The relaxed LP is defined as follows.

$$\begin{aligned}
\max \quad & \text{tr}(\bar{\mathbf{D}}\mathbf{x}^T) \\
\text{s.t.} \\
& \sum_{i=1}^{K} x_{ij} \leq 1 \quad \forall \quad j \in \{1, 2, \cdots, N\} \\
& \sum_{j=1}^{N} x_{ij} = \frac{L}{K} \quad \forall \quad i \in \{1, 2, \cdots, K\} \\
& 0 \leq x_{ij} \leq 1
\end{aligned}$$

After collecting all constraints in $\bar{\mathbf{A}}\bar{\mathbf{x}} \leq \bar{\mathbf{b}}$ form, where

$$\mathbf{x} = \begin{bmatrix} — & \mathbf{x}_1^T & — \\ — & \mathbf{x}_2^T & — \\ & \vdots & \\ — & \mathbf{x}_K^T & — \end{bmatrix}$$

$$\bar{\mathbf{x}} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_K \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{1}^T & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{1}^T & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{1}^T \\ -\mathbf{1}^T & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & -\mathbf{1}^T & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & -\mathbf{1}^T \\ \mathbf{e}_1^T & \mathbf{e}_1^T & \cdots & \mathbf{e}_1^T \\ \mathbf{e}_2^T & \mathbf{e}_2^T & \cdots & \mathbf{e}_2^T \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{e}_N^T & \mathbf{e}_N^T & \cdots & \mathbf{e}_N^T \end{bmatrix} \bar{\mathbf{x}} \leq \begin{bmatrix} \frac{L}{K} \\ \frac{L}{K} \\ \vdots \\ \frac{L}{K} \\ -\frac{L}{K} \\ -\frac{L}{K} \\ \vdots \\ -\frac{L}{K} \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

Given that $\bar{\mathbf{A}}$ is TUM, our optimal solution lies in an integer vertex, and thus our problem can be relaxed to an LP.

Figures 5 and 6 show the effectiveness of our selection algorithm. Results show slight improvement for select ratios for both datasets. However, random selection outperforms our algorithm when the ratio is low. This is intuitive because our approach prioritizes selection of points that are closer to the boundaries, which are not always the best representation of the entire dataset.

IV. Task 4

The task intends to compile the results obtained from Task 1, 2 and 3 respectively. The main agenda is to brief about the accuracy obtained for the classifier that indicates how well it classifies the data for the given dataset.

1) We tested the performance of the tasks for different no. of input data subsets [100, 250, 500, 1000] from the training data and also on [1200] by including the test data as well. However we observed that increasing the input samples doesn't necessarily lead to higher classification accuracy in case of supervised learning (see table I), especially for the synthetic data, the accuracy is more or less constant on increasing the no. of samples. Also, it is not necessarily true that using larger no. of samples with unsupervised learning can help achieve the same performance as supervised learning (see table II). Although a larger dataset may help the model identify more robust patterns or structures in the data, the performance also depends on the inherent structure of the data as well as the classification algorithm.
2) We observed that the clustering performance does change with a different no. of samples, but, a larger no. of samples doesn't guarantee better performance. However for a very small no. of samples [50 or 100] we observe a significant reduction in accuracy in the mnist data clustering (see table II). This probably is due to the higher dimension of the data in which case the algorithm struggles to find a pattern with too few samples and the cluster assignment becomes random.
3) Based on a data point that belongs to a cluster with a particular probability, a more robust classifier may be developed. We discovered that after assigning probability to the synthetic data points belonging to each of the three clusters, we were able to handle ambiguous data that had near membership across multiple clusters. Moreover, this led to better results with a smaller number of samples. We implemented soft decision clustering by assigning a probability to each cluster based on the distances from centroids. For every point in the data set, the probability value of it belonging to a certain cluster is checked and the final cluster assignment is done on based on the highest probability value of it associated to a cluster and hence leading to a better classification accuracy. Figures 8 and 9 show the effectiveness of our results based on the implemented soft decision clustering.
4) An alternative to using a majority vote which requires N true labels to assign the label to the clustered result, would be to identify 1 point per each cluster that are closest to each centroid. This method assumes that the centroid is best representative of that class. After obtaining the label of the point closest to the centroid, we can assign the labels of all other points in the cluster accordingly. This would require K ground truth labels instead of N.

TABLE I: Supervised learning classification accuracy for different no. of samples

Samples	**Synthetic data**	**MNIST data**
100	0.974	0.83
250	0.972	0.78
500	0.974	0.528
750	0.972	0.77
1000	0.972	0.834
1200	0.972	0.924

TABLE II: Unsupervised learning performance for different no. of samples

	Synthetic data		**MNIST data**	
Samples	classification	clustering	classification	clustering
50	0.968	0.852	0.546	0.293
100	0.97	0.884	0.83	0.31
250	0.966	0.881	0.628	0.408
500	0.966	0.867	0.798	0.562
750	0.966	0.844	0.784	0.572
1000	0.968	0.828	0.822	0.649
1200	0.966	0.826	0.826	0.599

References

[1] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

Appendix A
Task2 Unsupervised Clustering Algorithm

Algorithm 1: LP K-Means Clustering

Input : Data matrix $X \in \mathbb{R}^{n \times m}$,
Number of clusters $k \in \mathbb{R}$

Output: Cluster assignments

Initialization:
Randomly sample k number of data point from X and define them as centroids of clusters $C = \{c_1, c_2, \ldots, c_k\}$;

while *Not converged* **do**
- **Distance Metric:**
  Calculate Euclidean distance $d_{ij} \in \mathbb{R}$ between each data point $p_i \in \mathbb{R}^m$ and each centroid $c_j \in \mathbb{R}^m$;
- **Assignment Step:**
  Assign each data point p_i to the cluster with the nearest centroid by solving LP;
- **Update Step:**
  Update each centroid as the mean of the assigned data points;
- **Convergence Check:**
  Check if changes in centroids are within threshold or maximum iterations reached;

Output: Cluster assignments for each data point based on the final centroids;

Appendix B
Task 1.1 Another LP formulation for classification

For classifying input data $\mathbf{x} \in R^n$ into K groups, the maximum number of hyperplanes needed are $K-1$ (for eg. when all hyperplanes are parallel). Let us denote these $(K-1)$ hyperplanes as $\mathbf{w}_1^T\mathbf{x} + b_1 = 0, \mathbf{w}_2^T\mathbf{x} + b_2 = 0$, and so in till $\mathbf{w}_{K-1}^T\mathbf{x} + b_{K-1} = 0$,. Then, the hyperplanes divide the space so that all the K locations are characterized by one of the below cases.

- Case 1: $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0 \ \forall i \in \{1, K-1\}$
- Case 2: $\mathbf{w}_1^T\mathbf{x} + b_1 \leq 0$ and $\mathbf{w}_i^T\mathbf{x} + b_i \geq 0 \ \forall i \in \{2, K-1\}$
  And so on upto,
- Case K: $\mathbf{w}_i^T\mathbf{x} + b_i \leq 0 \ \forall i \in \{1, K-1\}$

After identifying these cases, we formulate the mathematical conditions for them.

We start with arranging the data so that data with the same labels is grouped together. Say the labels are $\{Y_1, Y_2, Y_3, \ldots Y_K\}$. The total no. of input data points ($\mathbf{x}$) are m. The points are distributed such that m_1 no. of points belong to the cluster labeled as Y_1, m_2 no. of points belong to the cluster labeled as Y_2, and so on. Clearly, $\sum_{i=1}^{K} m_i = m$.

Now we set up the linear program. Our variables are: $\mathbf{w}_i$ and $b_i \ \forall i \in \{1, K-1\}$.

- if $s_i = Y_p$ then we want $\mathbf{w}_1^T\mathbf{x} + b_1 \leq \epsilon \ \forall i \in \{1, p-1\}$ and $\mathbf{w}_i^T\mathbf{x} + b_i \geq \epsilon \ \forall i \in \{p, K-1\}$
- if the above condition doesn't hold, penalty is given by:

$$\sum_{i=m_1+m_2\ldots+m_{p-1}+1}^{m_1+m_2\ldots+m_p} \left[\sum_{j=1}^{p-1} \max\{\epsilon + \mathbf{w}_j^T\mathbf{x} + b_j, 0\} + \sum_{j=p}^{K-1} \max\{\epsilon - (\mathbf{w}_j^T\mathbf{x} + b_j), 0\} \right]$$

Thus the complete objective function can be written as,

$$\min \left\{ \sum_{p=1}^{K} \sum_{i=m_1+m_2\ldots+m_{p-1}+1}^{m_1+m_2\ldots+m_p} \sum_{j=1}^{p-1} \max\{\epsilon - (-1)^{\delta_{pj}}(\mathbf{w}_j^T\mathbf{x} + b_j), 0\} \right\}$$

where $\delta_{pj} = 1$ if $j < p$ and 0 if $j \geq p$.

The next step is to convert this to an LP, by making the objective function linear, using linear constraints. So we choose $m(k-1)$ slack variables z_{ij} and formulate the following LP.

$$\begin{aligned} \min \quad & \left\{ \sum_{p=1}^{K} \sum_{i=m_1+m_2\ldots+m_{p-1}+1}^{m_1+m_2\ldots+m_p} \sum_{j=1}^{p-1} z_{ij} \right\} \\ s.t. \quad & z_{ij} \geq 0 \\ & z_{ij} \geq \epsilon - (-1)^{\delta_{pj}}(\mathbf{w}_j^T\mathbf{x} + b_j) \end{aligned}$$

This Algorithm worked very well for synthetic dataset (2D data) achieving a classification accuracy of 0.978. But failed to accurately classify the higher dimension MNIST dataset. Hence we worked out the other proposed algorithm.

Appendix C
Plots and Performance Graphs

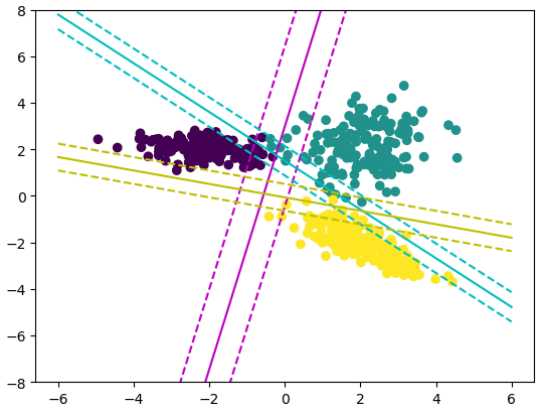
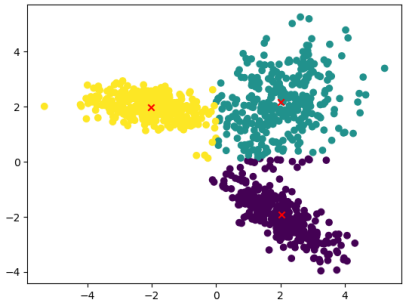
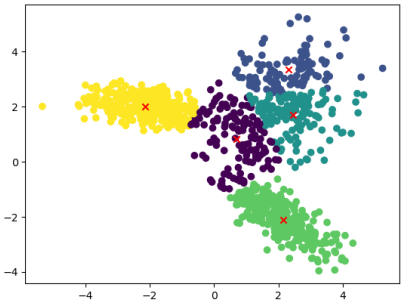


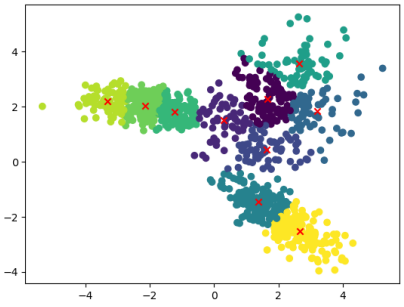
Fig. 1: Visualization of Synthetic Test Data Classification



(a) $K = 3$



(b) $K = 5$



(c) $K = 10$

Fig. 2: Clustering on synthetic data

Fig. 3: Clustering Evaluation

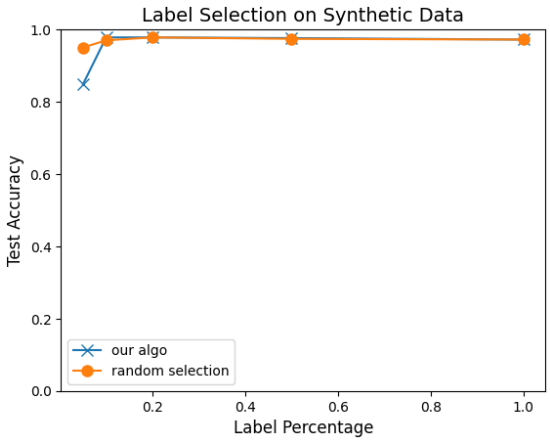


Fig. 5: Label Selection Evaluation on Synthetic Dataset

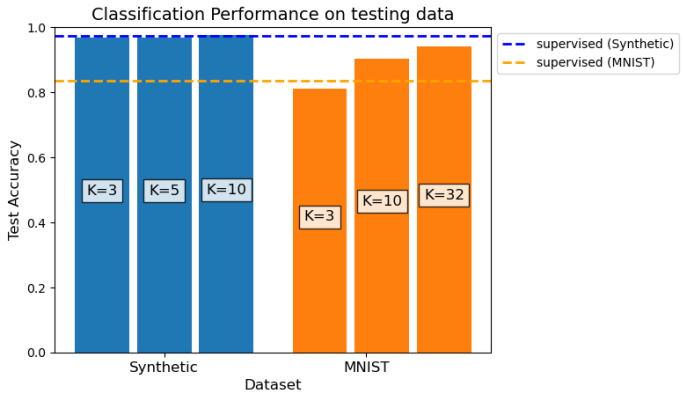


Fig. 4: Classification Evaluation

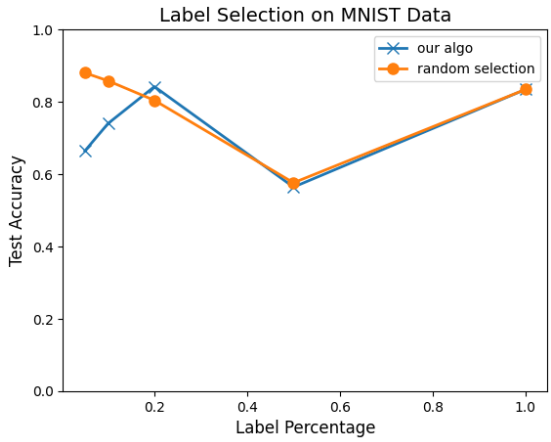


Fig. 6: Label Selection Evaluation on MNIST Dataset

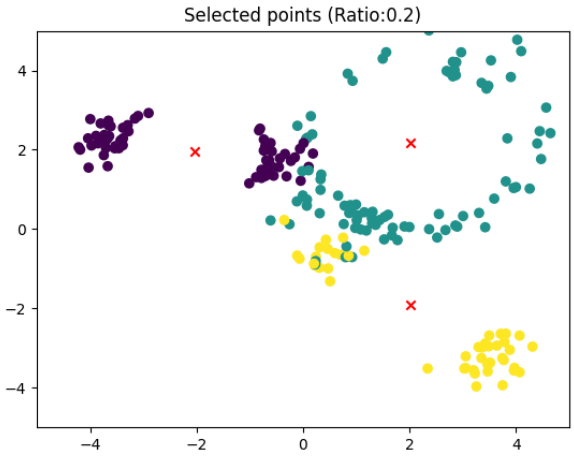


Fig. 7: Visualization of the selection algorithm

```
Point 1 Probabilities SYNTHETIC: [0.01098484 0.98210507 0.00691009]
Point 2 Probabilities SYNTHETIC: [7.17804316e-04 5.59984001e-03 9.936823
Point 3 Probabilities SYNTHETIC: [0.01020802 0.98421609 0.00557589]
Point 4 Probabilities SYNTHETIC: [9.90269253e-01 9.07767068e-03 6.530767
Point 5 Probabilities SYNTHETIC: [0.0173806  0.97069297 0.01192643]
Point 6 Probabilities SYNTHETIC: [0.98173639 0.01612283 0.00214077]
Point 7 Probabilities SYNTHETIC: [9.90070220e-01 9.01117448e-03 9.186057
Point 8 Probabilities SYNTHETIC: [0.00310519 0.02888032 0.96801449]
Point 9 Probabilities SYNTHETIC: [6.65413036e-04 5.87424060e-03 9.934603
Point 10 Probabilities SYNTHETIC: [0.00694475 0.97907174 0.01398351]
```

Fig. 8: Soft decision results for small number of samples

```
Final Cluster Assignments:
Point 1 assigned to Cluster in syn 2
Point 2 assigned to Cluster in syn 3
Point 3 assigned to Cluster in syn 2
Point 4 assigned to Cluster in syn 1
Point 5 assigned to Cluster in syn 2
Point 6 assigned to Cluster in syn 1
Point 7 assigned to Cluster in syn 1
Point 8 assigned to Cluster in syn 3
Point 9 assigned to Cluster in syn 3
Point 10 assigned to Cluster in syn 2
```

Fig. 9: Synthetic data results for softmax probabilities