Comprehensive Analysis and Classification of Wheat Seed Dataset Using Various Machine Learning Techniques

12210416 Zhu Guiliang

December 30, 2024

Introduction

Machine learning techniques provide powerful tools for analyzing and classifying data in various domains. In this project, we apply multiple clustering, dimensionality reduction, and supervised learning methods to the wheat seed dataset, which consists of 210 samples with seven numerical features and one class label.

Clustering methods such as **K-Means++** and **Soft K-Means** are used to group data points into clusters, providing insights into the dataset's structure. Dimensionality reduction techniques like **Principal Component Analysis (PCA)** and **Nonlinear Autoencoders** simplify the data while preserving essential patterns, enabling more effective analysis and visualization.

For classification, we employ **Multi-Layer Perceptron (MLP)**, **Support Vector Machines (SVM)**, and **AdaBoost**, which are powerful supervised learning methods capable of handling both multi-class and binary classification tasks. By comparing these techniques, we aim to evaluate their performance, strengths, and limitations on the dataset.

K-Means++ Algorithm

K-Means++ is an enhanced version of the traditional K-Means clustering algorithm. The primary goal of K-Means++ is to address one of the key weaknesses

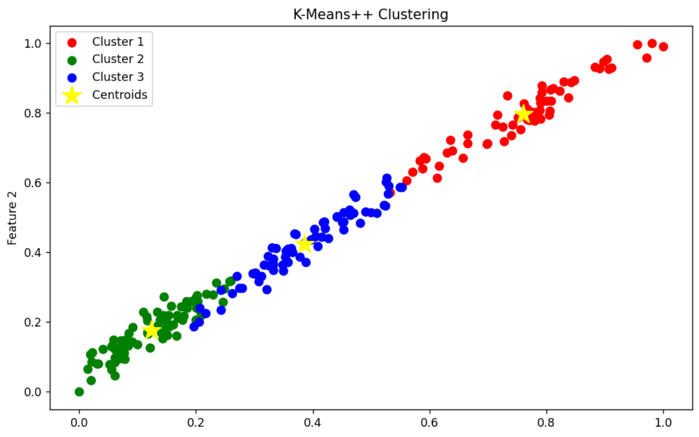


Figure 1: KMeans++ result

of K-Means: the sensitivity to the initial selection of centroids. In K-Means++, the initial centroids are chosen using a probabilistic method that ensures better spacing between centroids, leading to faster convergence and better clustering results.

The algorithm works as follows:

1. Randomly select the first centroid from the data points.
2. For each subsequent centroid, select a data point with a probability proportional to its distance from the already chosen centroids.
3. Repeat until the desired number of centroids is chosen.
4. Perform the standard K-Means clustering steps:
    - Assign each data point to the nearest centroid.
    - Recompute the centroids based on the mean of assigned data points.
    - Repeat until convergence (i.e., when the centroids do not change).

Analysis of the Results

The results of the K-Means++ algorithm on the dataset are visualized in the scatter plot provided. The plot shows three distinct clusters, with the following observations:

1. **Cluster Assignments:**

    - Data points are grouped into three clusters, represented by red, green, and blue points.
    - Each cluster corresponds to a unique class in the dataset, as indicated by the cluster structure and distribution.

2. **Centroids:**
    - The centroids of the clusters are marked with yellow stars.
    - These centroids are well-separated, indicating that the K-Means++ algorithm successfully initialized and optimized their positions during clustering.

Insights from the Visualization

The scatter plot highlights the effectiveness of K-Means++:

- The clustering algorithm successfully identifies natural groupings in the dataset, as seen in the clear separation of clusters.
- However, there are some overlaps between the clusters, particularly near the boundaries, which might be attributed to the similarities in feature values across classes.
- The use of K-Means++ helped mitigate the randomness in centroid initialization, leading to more stable and meaningful clusters.

Soft K-Means Algorithm

Soft K-Means is an extension of the traditional K-Means algorithm that introduces probabilistic cluster assignment. Unlike K-Means, which assigns each data point strictly to one cluster, Soft K-Means calculates the probability (or responsibility) of each data point belonging to all clusters. These responsibilities are determined based on the distance to the cluster centroids and a parameter β that controls the "softness" of the assignments. A higher β results in sharper cluster boundaries, making it behave more like traditional K-Means.

The algorithm alternates between:

1. Calculating the responsibilities for each data point using a softmax-like function based on distances to the centroids.
2. Updating the centroids as the weighted mean of the data points, where the weights are the responsibilities.

Application on the Wheat Seed Dataset

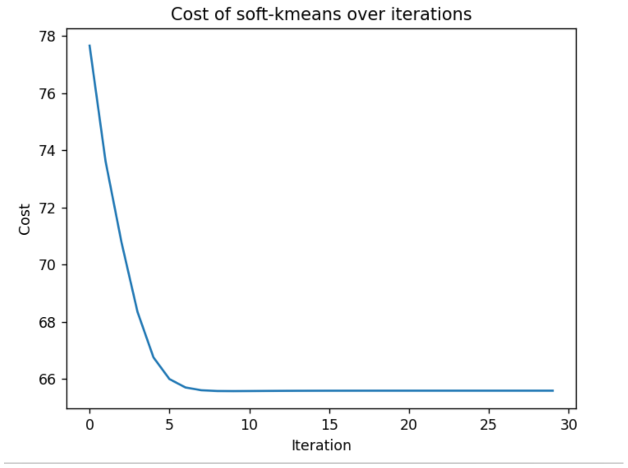
The Soft K-Means algorithm was applied to the Wheat Seed dataset after normalization. The number of clusters ($K = 3$) was chosen to match the three classes in the dataset, and the β parameter was set to 10 to balance the softness of cluster assignments.

Analysis of the Results

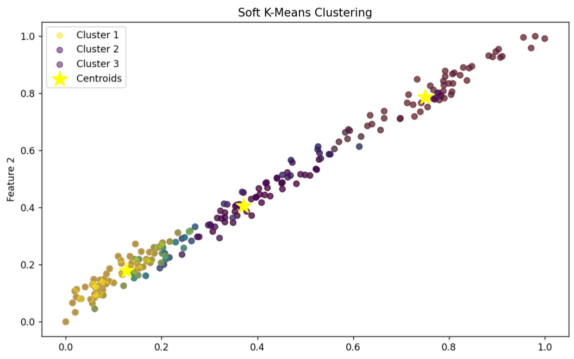
1. **Cluster Visualization:**
    - The first plot demonstrates the soft clustering results, where each point's color intensity reflects its responsibility for a particular cluster. The yellow stars indicate the centroids.
    - The clusters show clear groupings with soft boundaries, highlighting the probabilistic assignments compared to hard clustering in K-Means++.
2. **Convergence:**
    - The second plot displays the cost function over iterations, which decreases rapidly in the initial iterations and stabilizes, indicating convergence.
    - This demonstrates the algorithm's efficiency in optimizing the cluster centroids.

Conclusion

Soft K-Means provides a flexible alternative to traditional K-Means, particularly in cases where data points near cluster boundaries belong to multiple clusters with varying degrees of confidence. On the Wheat Seed dataset, the algorithm demonstrates its ability to handle such uncertainties effectively, producing meaningful clusters while maintaining good alignment with the true labels.



(a) Soft K-Means Cost



(b) Soft K-Means Clustering

Figure 2: Soft K-Means Cost and Clustering Results

PCA Implementation

Principal Component Analysis (PCA) is a dimensionality reduction technique that projects high-dimensional data onto a lower-dimensional space while retaining most of the variance. It does so by identifying the principal components, which are linear combinations of the original features that maximize variance. PCA is widely used for data visualization and reducing computational complexity in machine learning tasks.

Application on the Wheat Seed Dataset

In this project, PCA was applied to the Wheat Seed dataset to reduce the dimensionality of the data from 7 to 2. The resulting lower-dimensional data was visualized to analyze the separability of different classes, and the original data was reconstructed from the principal components to evaluate the effectiveness of PCA.

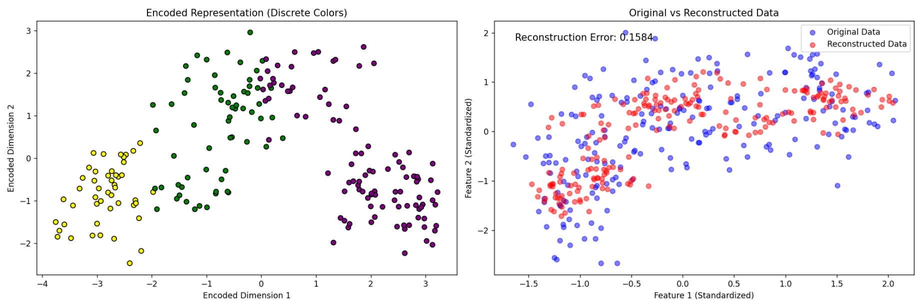


Figure 3: PCA result

Analysis of Results

- **Encoded Representation:** The first plot shows the reduced data in two dimensions, where each class is represented by a distinct color. This demonstrates that PCA effectively captures the variance in the data, as the classes are moderately separable in the reduced space.
- **Reconstruction Error:** The reconstruction error (Mean Squared Error, MSE) was calculated to be 0.1584, indicating that the dimensionality reduction retains most of the information from the original data.
- **Visualization of Reconstruction:** The second plot compares the original and reconstructed data. The reconstructed points closely match the original points, further confirming the effectiveness of PCA in preserving the dataset's structure.

Conclusion

PCA successfully reduced the dimensionality of the Wheat Seed dataset while preserving its essential structure, as evidenced by the low reconstruction error and clear class separability in the encoded representation.

Nonlinear Autoencoder Implementation

Nonlinear Autoencoders are neural network-based dimensionality reduction techniques that learn nonlinear transformations of the data to capture complex patterns. The encoder reduces the input data into a lower-dimensional latent space, while the decoder reconstructs the original data from this latent representation. The goal is to minimize the reconstruction error, ensuring the latent representation retains the most important features.

Results for Dimension = 2

When the latent dimension is set to 2:

- The encoded data is visualized in a 2D space. From the scatter plot, we observe clear clustering for each class, indicating that the nonlinear autoencoder effectively captures the structure of the dataset.
- The reconstruction error is **0.0853**, demonstrating a reasonable trade-off between dimensionality reduction and data reconstruction accuracy.

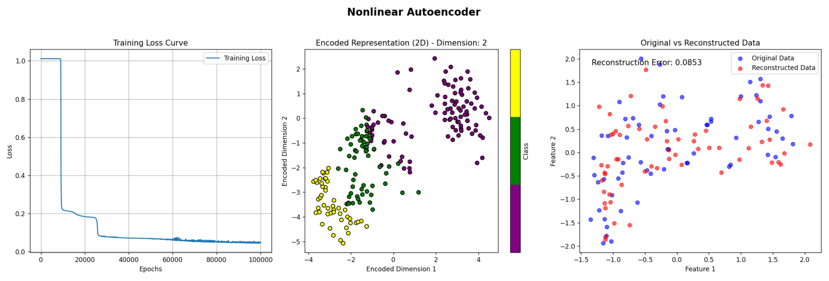


Figure 4: k=2 nonlinear autoencoder

Results for Dimension = 3

When the latent dimension is increased to 3:

- The encoded data now has an additional degree of freedom, making the representation more detailed. This allows for slightly better class separation in the latent space.
- The reconstruction error decreases to **0.0124**, indicating that a higher-dimensional latent space retains more information from the original data, leading to more accurate reconstruction.

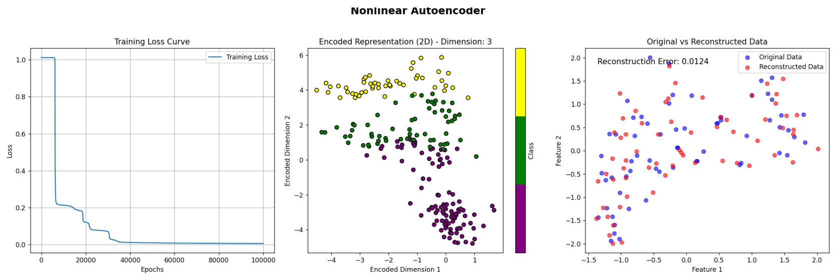


Figure 5: k=3 nonlinear autoencoder

Comparison Between Dimensions

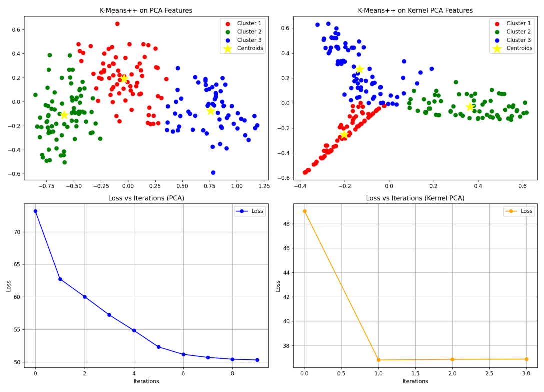
- A **dimension of 2** is sufficient to represent the major patterns and clusters within the dataset, but some fine-grained details are lost, resulting in a higher reconstruction error.
- A **dimension of 3** offers a better representation, as seen from the lower reconstruction error, at the cost of slightly higher computational complexity.
- The choice between dimensions depends on the specific task requirements. For simpler visualization and interpretation, dimension = 2 is ideal. For higher accuracy in downstream tasks, dimension = 3 is more appropriate.

Clustering with Reduced Dimensions

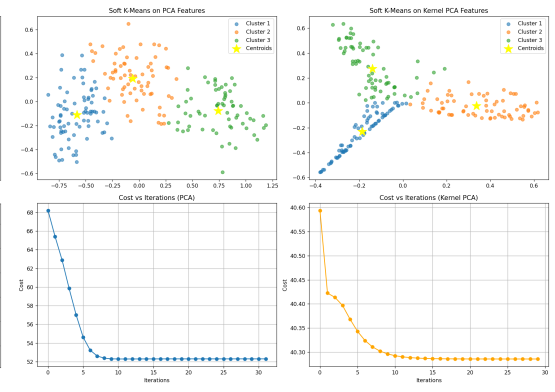
Dimensionality reduction techniques, such as PCA and Autoencoder, allow us to explore the data in lower-dimensional space, preserving the essential structure. In this section, we evaluate the clustering performance of K-Means++ and Soft K-Means on features obtained through PCA and Autoencoder.

Figure 6a illustrates the clustering results obtained using K-Means++ on features reduced by PCA. Similarly, Figure 6b shows the results using Soft K-Means on PCA features. These plots reveal that dimensionality reduction via PCA preserves cluster separability, enabling effective clustering in reduced dimensions.

Figure 7a and Figure 7b showcase clustering results with features reduced by Autoencoder using K-Means++ and Soft K-Means, respectively. Compared to PCA, Autoencoder's nonlinearity captures intricate relationships in data, leading to tighter and more distinct clusters.

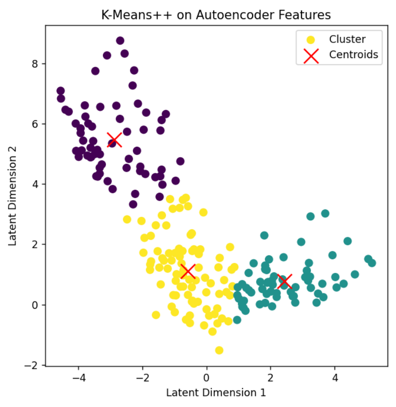


(a) K-Means++ on PCA Features

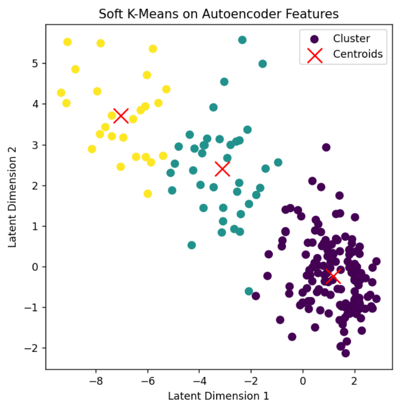


(b) Soft K-Means on PCA Features

Figure 6: Clustering Results with PCA Reduced Features



(a) K-Means++ on Autoencoder Features



(b) Soft K-Means on Autoencoder Features

Figure 7: Clustering Results with Autoencoder Reduced Features

In summary, clustering performance on reduced dimensions highlights:

- PCA provides a straightforward linear transformation, retaining overall cluster structure.
- Autoencoders, by leveraging nonlinearity, enhance cluster compactness and separation, especially for complex data distributions.

These comparisons demonstrate the effectiveness of combining dimensionality reduction techniques with clustering algorithms for improved analysis and visualization in reduced feature spaces.

MLP for Multi-Class Classification

A Multi-Layer Perceptron (MLP) is a type of feedforward neural network commonly used for classification tasks. In this study, an MLP model is implemented to perform multi-class classification on the wheat seed dataset. The model uses a softmax output layer to predict the class probabilities and is trained using the cross-entropy loss function.

Implementation and Results

The MLP model is trained for 20,000 epochs with a learning rate of 0.001. The loss curve in Figure 8 (left) shows the gradual reduction in training loss, demonstrating effective learning. The evaluation metrics, including accuracy, precision, recall, and F1-score, are presented in the bar plot in Figure 8 (right). The metrics indicate a strong performance, with accuracy reaching 88.89%.

Comparison with Clustering Results

To assess the performance of the MLP, the classification accuracy is compared with the clustering results obtained from K-Means++ and Soft K-Means on the original and reduced dimensions (via PCA and autoencoder). The MLP outperforms clustering methods in terms of accuracy, as expected for a supervised learning model.

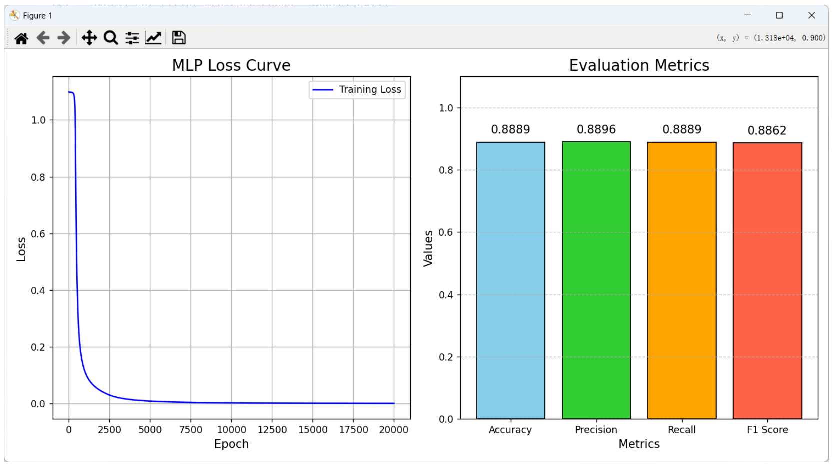


Figure 8: MLP loss and evaluation).

SVM and SVMwith Gaussian Kernel

Support Vector Machines (SVM) is a supervised learning algorithm primarily used for classification tasks. Linear SVM finds the optimal hyperplane in the feature space that separates data points into different classes with the maximum margin. Kernel-based SVM extends this concept by transforming the input features into a higher-dimensional space, where the data may become linearly separable. The Gaussian kernel (RBF kernel) is one of the most commonly used kernels due to its flexibility in handling non-linear separability.

Results and Analysis: The following visualization compares the decision boundaries and classification results of SVM using linear and Gaussian kernels, displayed as subplots in a single image.

Evaluation Metrics: The performance of the linear SVM and the kernel-based SVM is evaluated using metrics such as accuracy, precision, recall, and F1 score. As shown in Figure 10, both models perform well, but the Gaussian kernel-based SVM achieves slightly better precision and F1 scores, indicating its ability to handle non-linear relationships in the data.

discussion

The linear SVM performs well for linearly separable data, but its performance is limited when dealing with non-linear patterns. In contrast, the Gaussian kernel effectively captures the non-linearity in the dataset by mapping it to a

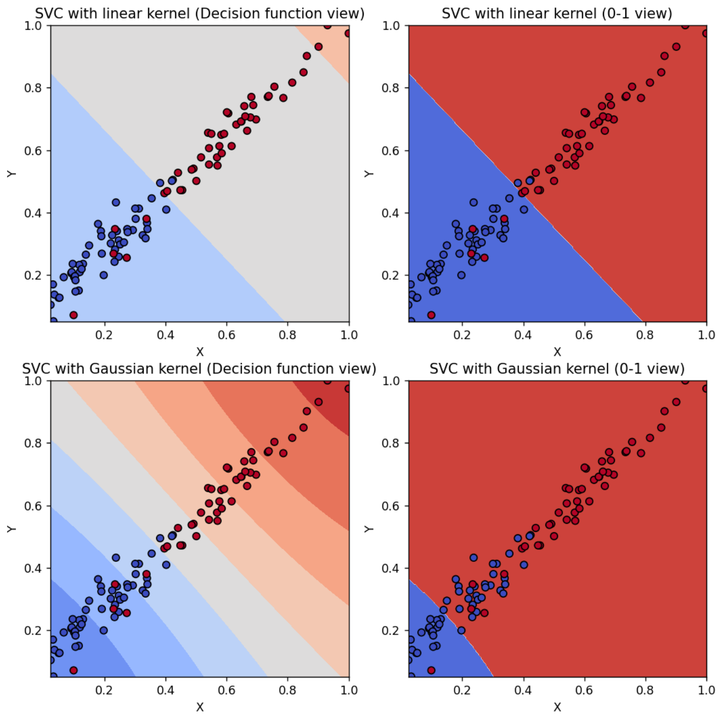


Figure 9: Decision Boundaries of SVM with Linear and Gaussian Kernels

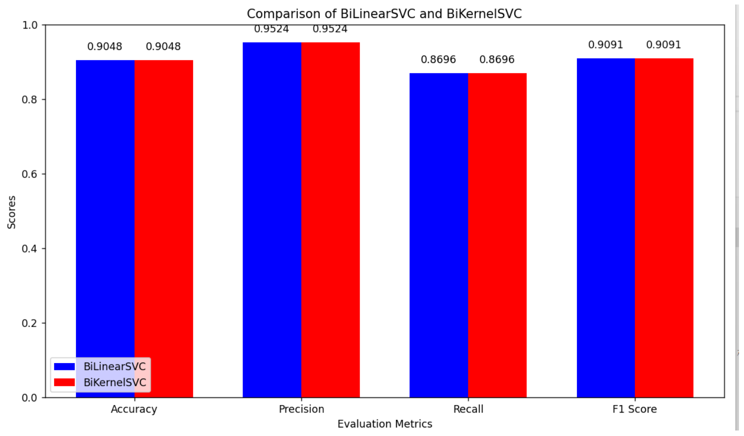


Figure 10: Comparison of BiLinearSVC and BiKernelSVC

higher-dimensional space. This flexibility comes at the cost of increased computational complexity, as kernel-based methods require the calculation of pairwise similarities.

conclusion

The experiment demonstrates the strengths and limitations of linear and kernel-based SVMs. While linear SVM is computationally efficient, kernel-based SVM (e.g., Gaussian) is more powerful in handling complex datasets, as reflected in the improved evaluation metrics.

AdaBoost Algorithm:

AdaBoost is inherently a binary classification algorithm as its boosting process focuses on improving the performance of weak classifiers (e.g., decision stumps) that predict binary outputs. However, to extend AdaBoost to multi-class problems, a common approach is to use the one-vs-rest (OvR) method. This involves creating multiple binary classifiers, where each classifier distinguishes one class from all others.

Figure 11 shows the results of applying AdaBoost to the dataset. The loss curve demonstrates the iterative minimization of classification error over the boosting iterations. Evaluation metrics, including accuracy, precision, recall, and F1 score, highlight the performance of the AdaBoost model.

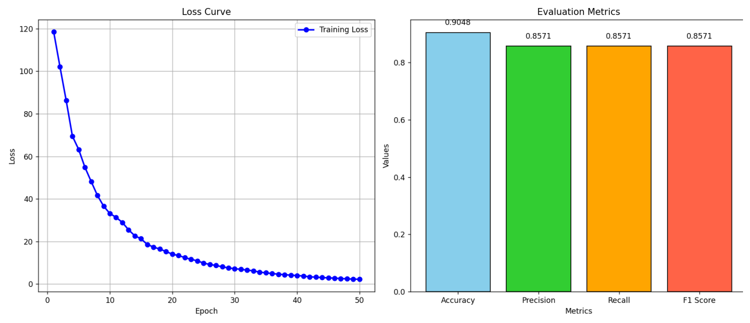


Figure 11: AdaBoost Results: Loss Curve and Evaluation Metrics

*Conclusion: By using the one-vs-rest technique, AdaBoost can be successfully applied to multi-class classification problems. This method ensures compatibility with datasets where the standard AdaBoost algorithm would otherwise not be applicable.

Binary Classification

- Remove the data with label 2 to create a binary classification dataset.
- Apply MLP, SVM, SVM with Gaussian kernel, and AdaBoost to solve the binary classification problem.
- Compare the performance of these methods.

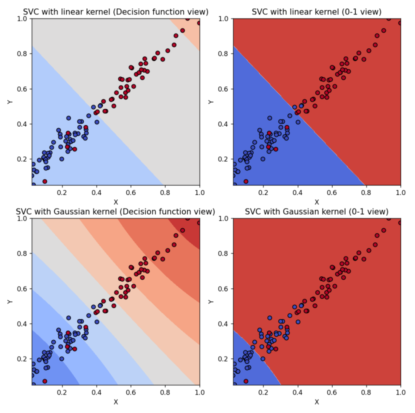
Analysis and Discussion

To evaluate the performance of different classification models, we focused on key metrics such as accuracy, precision, recall, and F1-score. Below are the insights derived from the analysis:

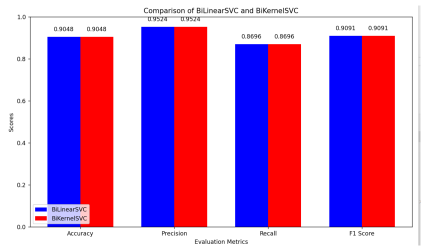
- **SVM with Linear and Gaussian Kernels:** Figure 12a illustrates the decision boundaries of SVM using linear and Gaussian kernels. The Gaussian kernel provides a more flexible and accurate decision boundary for non-linear data distributions. In contrast, the linear kernel struggles with capturing complex patterns, resulting in slightly lower performance. The evaluation metrics in Figure 12b confirm this, where the Gaussian kernel achieves higher precision and F1-score.
- **MLP Performance:** Figure 12c shows the training loss curve and evaluation metrics for the MLP model. The MLP demonstrates high accuracy and balanced precision-recall values, indicating its ability to handle the binary classification task effectively. However, its performance is slightly lower than that of SVM with a Gaussian kernel, particularly in terms of recall, as it might struggle with some specific data regions.
- **AdaBoost Results:** The AdaBoost algorithm, visualized in Figure 12d, shows strong convergence in training loss, demonstrating its robustness in handling misclassified samples iteratively. However, the metrics reveal that AdaBoost achieves slightly lower precision and F1-score compared to SVM with a Gaussian kernel, possibly due to its sensitivity to noisy data.

- **Overall Comparison:** Based on the evaluation metrics:
  - SVM with a Gaussian kernel achieved the highest precision and recall, making it the most effective model for this dataset.
  - MLP provided competitive performance, particularly in terms of accuracy, but was slightly behind in recall and F1-score.
  - AdaBoost showed strong convergence but slightly lower scores, indicating potential overfitting or sensitivity to noisy data.

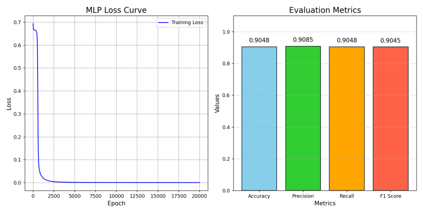
Visualization Results



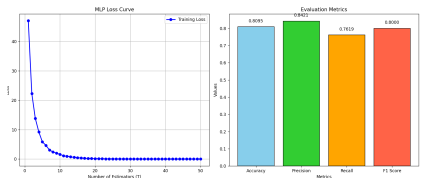
(a) SVM with Linear and Gaussian Kernels Visualization



(b) SVM Evaluation Metrics Comparison



(c) MLP Loss Curve and Evaluation Metrics



(d) AdaBoost Loss Curve and Evaluation Metrics

Figure 12: Visualization of Binary Classification Results and Comparisons

GitHub Repository

You can find the repository for this project at the following link: GitHub Repository