Data Description and Exploratory Analysis

The Pumpkin Seeds Dataset, available on Kaggle, provides data for classifying two distinct pumpkin seed varieties, Ürgüp Sivrisi and Çerçevelik, based on 12 morphological features. It contains 2,500 samples, evenly split between the two varieties, offering a balanced foundation for machine learning approaches. These 12 numerical features serve as the independent variables, while the categorical target variable, "Class," indicates the specific seed variety. This study aims to employ Support Vector Machine (SVM) to classify the two seed varieties according to their physical characteristics, a task of particular importance for agricultural applications such as seed sorting, breeding research, and quality evaluation.

The dataset contains 13 columns, with 12 numerical features (X) and one target variable, "Class" (Y). All features are either integer or float values, and there are no missing entries. From the 12 histograms, features such as Area and Perimeter appear slightly right-skewed, whereas Eccentricity and Roundness show tighter distributions. The final bar chart confirms that the dataset is balanced between the two classes ("Çerçevelik" and "Ürgüp Sivrisi"). The pairplot indicates some overlap between the classes, but features like Area, Perimeter, and Major/Minor Axis Length offer partial separation, whereas Eccentricity and Compactness exhibit greater overlap.

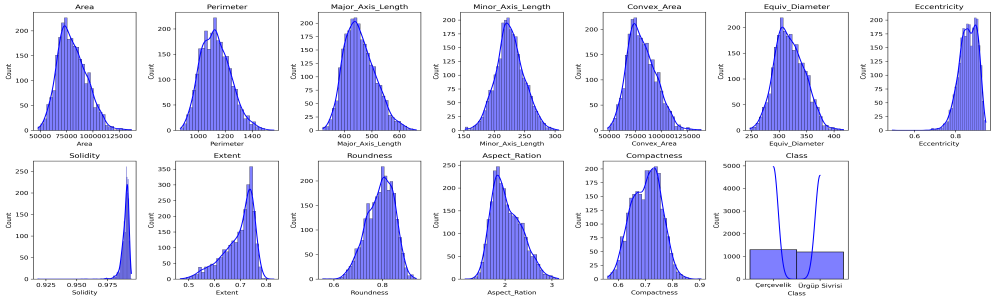

Figure 1: Distribution of All Variables

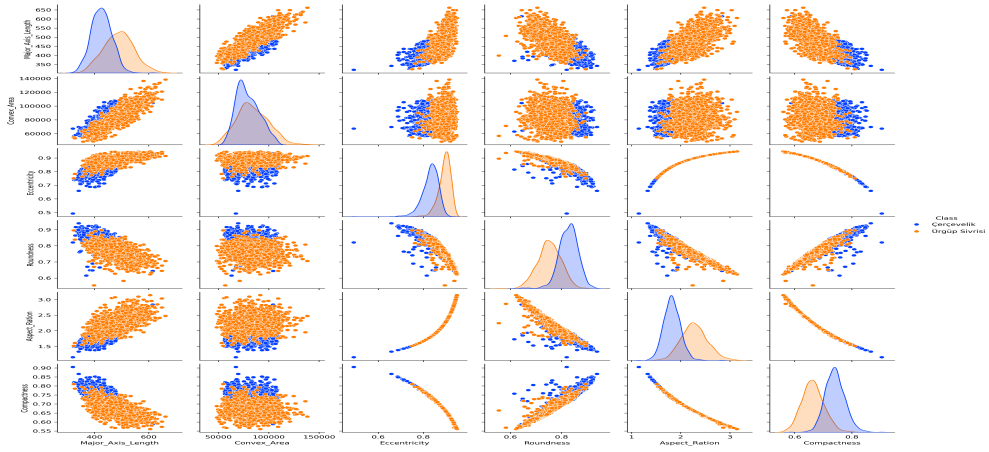

Figure 2: Pair Plot of Key Variables

Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm mainly used for classification tasks. Its goal is to find the best way to split data into distinct categories, essentially by drawing a clear boundary between different variables. Mathematically, if we have a training set $\{(x_i, y_i)\}$ with $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, SVM finds a weight vector $\mathbf{w}$ and bias b that minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i,$$

subject to

$$y_i\big(\mathbf{w}\cdot x_i + b\big) \geq 1 - \xi_i, \quad \xi_i \geq 0,$$

where C is a regularization parameter, and ξ_i are "slack variables" allowing for margin violations.

In both linearly separable and non-linear cases, SVM seeks a boundary known as a hyperplane that maximizes the margin, or the distance from the boundary to the nearest data points in each class. These nearest points are called support vectors, and they are crucial because they define the position of the hyperplane. By maximizing the margin, SVM aims to generalize well to new, unseen data.

For more complex data, SVM relies on kernel functions to map the original feature space into a higher-dimensional space where a straightforward boundary (hyperplane) can separate the classes. Common kernels include linear, polynomial, radial basis function (RBF), and sigmoid, each suitable for different patterns in the data.

When the data are not perfectly separable, SVM adopts a soft-margin approach that leverages the C parameter to control the trade-off between widening the margin and reducing classification errors. A lower C allows a wider margin at the cost of more misclassifications, while a higher C narrows the margin to minimize errors. Because only the support vectors affect the hyperplane's position, SVM can be efficient even in high-dimensional spaces and is generally less prone to overfitting than methods that rely on every data point.

SVM with Linear Kernel

Initially, the Class column was converted to a categorical variable, and the dataset was stratify split into training and testing sets in an 80/20 ratio. The first kernel tested (Linear) produced a high "C" value during grid search, indicating that the model prioritizes minimizing classification errors over maximizing the margin. This suggests the data is nearly linearly separable, but some overlap remains, requiring a narrower margin. The best 10-fold cross-validation (CV) result on the training set averaged 88.75%, showing robust performance and suggesting that the model generalizes well within the training data. The test accuracy of 87.6% slightly lower than the CV accuracy which is expected because the test set is unseen data (with scaling applied after splitting to prevent leakage). This small drop hints at good generalization, although minor overfitting may occur due to the high C value.

An SVM with a linear kernel was then trained on the Pumpkin Seeds Dataset to classify "Çerçevelik" and "Ürgüp Sivrisi" varieties, achieving a test accuracy of 87.6% after hyperparameter tuning (C=100). Cross-validation yielded 88.75% accuracy, indicating strong generalization. The model demonstrated high recall (0.92) for "Çerçevelik" and high precision (0.90) for "Ürgüp Sivrisi," with F1-scores of 0.88 and 0.87, respectively, reflecting balanced performance. These results align with previous benchmarks (88% SVM accuracy) and suggest that the linear kernel effectively captures the dataset's morphological distinctions, making it well-suited for applications like seed sorting.

SVM with Radial Kernel

The RBF kernel results with $C = 1$ suggest a moderate balance between margin maximization and error tolerance, less aggressive than the linear model's $C = 100$. Meanwhile, $\gamma = 0.1$ is flexible but not overly localized (compared to, say, $\gamma = 1$). The best cross-validation (CV) accuracy of 0.8905 (89.05%) indicates slightly better performance on the training set, while the test accuracy of 0.876 (87.6%) suggests that both models generalize equally well on the test set.

In terms of training performance, the RBF kernel's 89.05% CV accuracy outperforms the linear kernel's 88.75% by a small margin, implying it captures the data's subtle non-linearities (e.g., interactions between eccentricity and aspect ratio). However, both achieve the same 87.6% accuracy on the test set, with identical classification metrics, so there is no practical difference in real-world performance.

From a simplicity standpoint, the linear kernel relies on a single parameter (C) and assumes near-linear separability, which appears sufficient for this dataset. The RBF kernel, with its two parameters (C and γ), is more complex and computationally demanding due to its non-linear mapping. While this flexibility might offer advantages with different splits or noisier data, the linear kernel's high C value seems robust enough for this balanced, clean dataset.

Overall, the RBF kernel model $(C = 1, \gamma = 0.1)$ performs marginally better in cross-validation (89.05% vs. 88.75%), suggesting greater consistency across training folds and potentially making it a safer bet for truly unseen data. Yet, the identical test accuracy (87.6%) and classification metrics mean there is no practical difference in the final outcome of this experiment.

Clustering Models

(a) MCLUST Classtering

First, unsupervised learning with `MCLUST` clustering was performed, indicating that 9 clusters provided the best fit among the default range tested (1 to 9 clusters). Each cluster used the `VVV` covariance structure, having distinct covariance matrices. The best model had a BIC score of 98599.44 and a similar ICL score of 98523.61, showing low overlap between clusters.

Second, semi-supervised learning set the cluster count to 2, matching the actual classes in the dataset. The model again used the `VVV` covariance structure but achieved a significantly lower BIC score of 58549.63. Comparing predictions with true classes resulted in a misclassification rate of 0.27 and an ARI of 0.21, indicating the model partially captured the real class structure but still had noticeable mismatches.

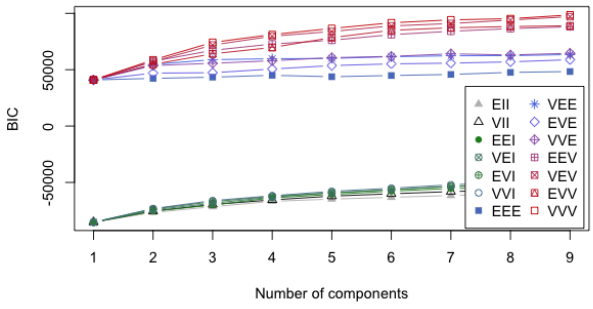

Figure 3: MCLUST: BIC of 1 to 9 Clusters

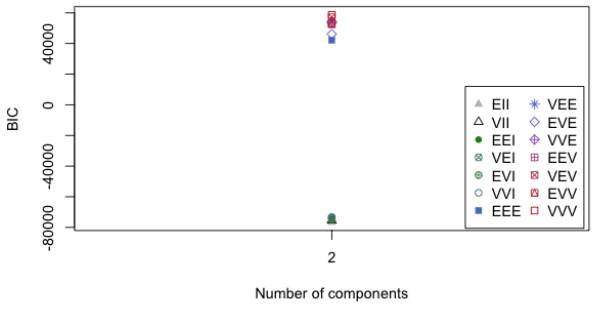

Figure 4: MCLUST: BIC of 2 Clusters

(b) VSCC Clustering

In the unsupervised VSCC run, the algorithm identified nine clusters (VVV model) with a BIC of 99095.11, retaining all 12 variables. This indicates that removing any variable did not improve the separation enough to justify dropping it. However, when the number of clusters was constrained to two, the BIC decreased to 58549.63, again using all 12 variables. This shows that while the data can be split into two clusters to match the number of actual classes, it does not capture as much structural complexity as the nine-cluster solution, indicating a trade-off between following the known two-class labels and capturing the more detailed structure in the data.

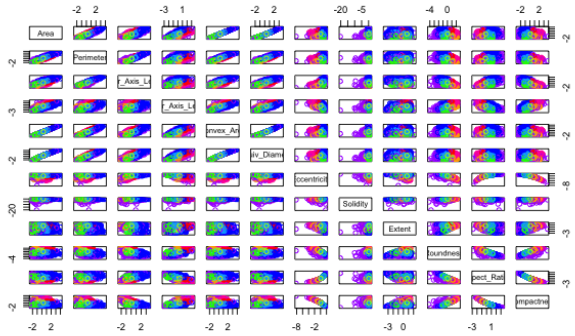

Figure 5: VSCC: BIC of 1 to 9 Clusters

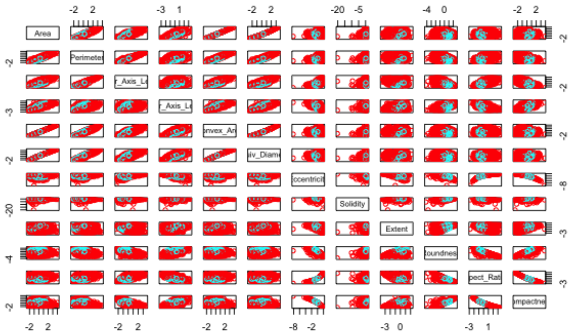

Figure 6: VSCC: BIC of 2 Clusters

(c) Comparison of Clustering Results

In the purely unsupervised setting, both MCLUST and VSCC indicate a nine-cluster solution (VVV model), suggesting that the data may naturally form multiple subgroups when allowed to freely select the number of clusters. However, under a semi-supervised approach that restricts the analysis to two clusters (reflecting the real class structure), both methods continue to use all 12 variables but produce different levels of performance. Specifically, MCLUST achieves a misclassification rate (MCR) of 0.27 and an Adjusted Rand Index (ARI) of 0.21, whereas VSCC achieves a lower MCR of 0.22 and a higher ARI of 0.31. This indicates that while both approaches uncover similar multi-cluster structure in unsupervised mode, VSCC offers slightly better alignment with the known two-class labels under semi-supervised conditions.

Model	**Predictor**	**Cluster**	**MCR**	**ARI**
MCLUST	12 variables	2	0.27	0.21
VSCC	12 variables	2	0.22	0.31

Table 1: Semi-supervised Clustering Comparison