Flower Image Classification Project Report

Ori Yair Yaakov 207723198
Shlomi Asraf 207970252

Abstract

This report presents our work on classifying flower images into five categories: **Daisy**, **Dandelion**, *rose*, **Sunflower**, and **Tulip**. The project explores several classical machine learning models: K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Machine (SVM), and Naive Bayes. The process includes feature extraction from grayscale images, dimensionality reduction (PCA), and feature selection based on mutual information scores. Each model is evaluated based on classification performance, with results demonstrated through statistical metrics and visual analysis.

1 Introduction

1.1 Project Goal

The goal of this project is to classify flower images into one of five categories: **Daisy**, **Dandelion**, *rose*, **Sunflower**, or **Tulip**. The classification is performed using classical machine learning methods. Several models are developed and evaluated: K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Machine (SVM), and Naive Bayes.

1.2 Dataset

The dataset for this project consists of grayscale images resized to a uniform resolution of 32×32 pixels. Each image is labeled according to its corresponding flower class: **Daisy**, **Dandelion**, *rose*, **Sunflower**, or **Tulip**. For feature extraction, the raw pixel intensities were used along with two additional engineered features: the mean pixel intensity and the standard deviation of pixel values for each image. Furthermore, to ensure fair model training and evaluation, the dataset was balanced so that each flower class contains an equal number of samples.

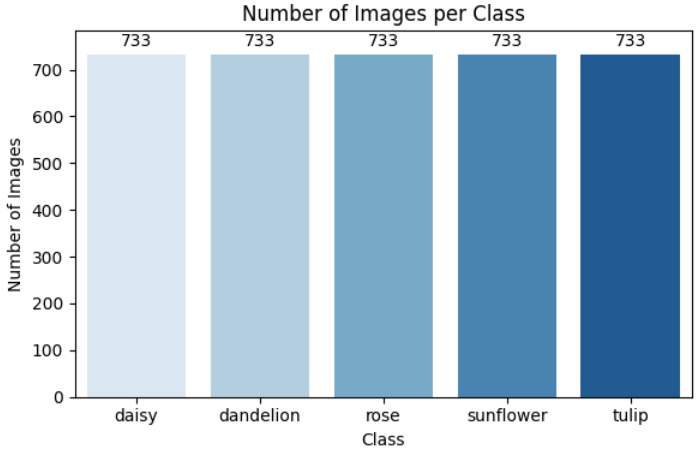

Figure 1: Graph of the Data Count per Class.

1.3 Data Preprocessing

The preprocessing phase involved preparing the raw images for machine learning. Each image was resized, converted to grayscale, and flattened into a feature vector. After appending the engineered mean and standard deviation features, all pixel values were normalized to the range [0,1]. Subsequently, feature standardization was applied to ensure zero mean and unit variance. Finally, the dataset was split into training and testing subsets using an 80%-20% split.

1.4 Feature Engineering

Following basic preprocessing, additional feature transformations were conducted to further enhance model performance. Principal Component Analysis (PCA) was applied to reduce the dimensionality of the feature space to 150 principal components, preserving the most significant variance in the data. Next, polynomial feature expansion of degree 2 was performed to capture potential interactions between features. Lastly, feature selection was applied using the SelectKBest method based on mutual information scores, selecting the top 100 most informative features for model training.

1.5 About The Models

This project explores different classical machine learning models for classifying flower images into five categories. We start with the K-Nearest Neighbors (KNN) algorithm, which classifies images based on the majority class of the nearest samples using the Manhattan distance metric. Logistic Regression is then applied, building on a linear decision boundary optimized via regularization techniques. Support Vector Machine (SVM) with an RBF kernel is used next, enabling the model to handle non-linear relationships in the data. Finally, a Naive Bayes classifier is employed, offering a simple yet efficient probabilistic approach for classification based on feature independence assumptions. In addition, a weighted soft-voting ensemble model is constructed to combine the strengths of all individual models and improve overall classification performance.

2 Exploratory Data Analysis (EDA)

In this section, we explore the structure and basic statistical properties of the dataset to gain insights before applying machine learning models.

2.1 Pixel Intensity Statistics

We begin by analyzing the distribution of pixel intensity statistics (mean and standard deviation) across different flower classes.

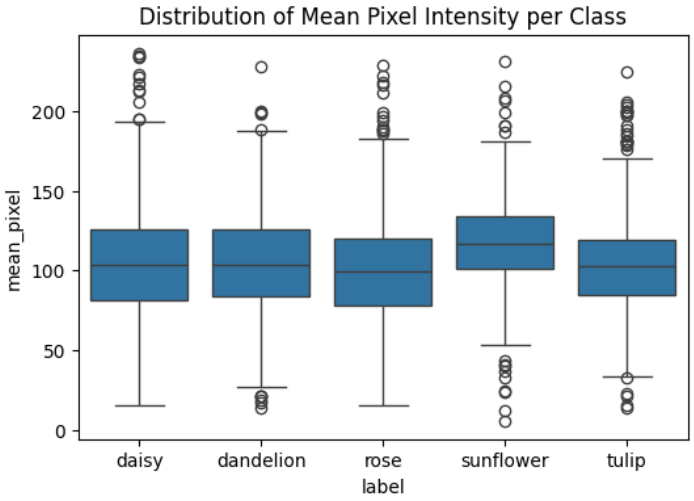

Figure 2: Distribution of Mean Pixel Intensity per Class.

As seen in the figure, **Sunflower** images generally have slightly higher mean pixel intensity compared to other classes, suggesting that they tend to be brighter.

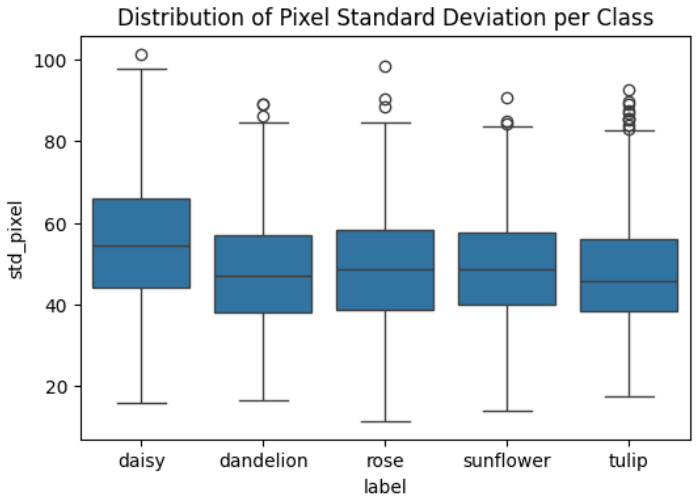

Figure 3: Distribution of Pixel Standard Deviation per Class.

The distribution of pixel standard deviations shows that **Daisy** images exhibit more variation in pixel intensities, while *rose* and **Tulip** classes have somewhat lower variability, potentially reflecting differences in texture complexity.
These statistics are not only useful for visualization but were also incorporated as engineered features to enhance model training.
Specifically, the mean pixel intensity of an image provides a measure of overall brightness, while the standard deviation captures the variation or texture complexity within the image.
Flowers such as daisies, with complex textures and varied pixel intensities, typically exhibit higher standard deviation values. In contrast, flowers with smoother appearances, such as roses, tend to have lower variation.
By adding both the mean and standard deviation alongside the raw pixel features, we enriched the dataset with higher-level descriptors that improve the model's ability to differentiate between flower classes, especially in cases where pixel values alone may not provide sufficient information.

2.2 PCA Visualization

To gain a better understanding of the dataset's structure, Principal Component Analysis (PCA) was performed to project the data onto two dimensions.

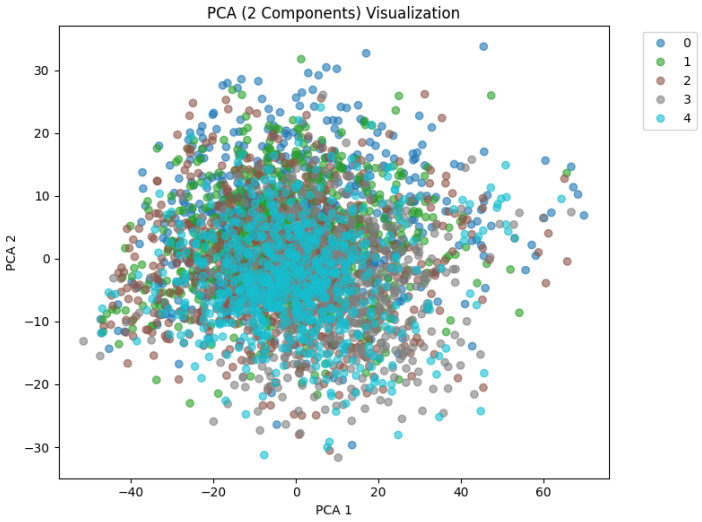

Figure 4: PCA 2D Visualization of the Dataset.

PCA reduces the dimensionality of the original feature space while preserving as much variance as possible. The 2D projection reveals that the samples from different flower classes are densely clustered together with significant overlap.
This suggests that raw pixel intensities and basic statistical features (mean and standard deviation) are insufficient for linear separation between certain flower classes. The lack of clear boundaries between clusters highlights the challenge of distinguishing between visually similar flower types.
These findings reinforce the need for more sophisticated techniques—such as polynomial feature expansion, feature selection, and non-linear classifiers like SVM with RBF kernel—to improve model performance.

2.3 Handling Nonlinear Complexity

As previously introduced in the Feature Engineering section, we applied several transformations to enhance model capacity. This section explains how those techniques help to address the nonlinear structure observed in the data.
The PCA visualization demonstrates that the flower classes are not linearly separable in the feature space. To address this nonlinear complexity, multiple advanced techniques were applied. Polynomial feature expansion of degree 2 was introduced to capture interactions between the original features, enabling models to learn nonlinear decision boundaries. Feature selection was then performed using the SelectKBest method based on mutual information scores, ensuring that only the most informative features were retained for training. Furthermore, Support Vector Machine (SVM) with an RBF kernel was utilized, allowing the model to project the data into a higher-dimensional space where a linear separation becomes possible. K-Nearest Neighbors (KNN) was also used with the Manhattan distance metric to better capture local neighborhood structures. Finally, a weighted ensemble voting classifier was built by combining the outputs of all models, leveraging the strengths of each to improve robustness and overall classification performance.

3 Feature Insights and Contribution Analysis

Beyond standard preprocessing, feature importance analysis provides deeper insights into the classification challenges.
To further enhance the classification performance, feature selection was performed using the SelectKBest method based on mutual information scores. Mutual information measures the dependency between each feature and the class label, allowing us to identify the most informative features for the classification task.
The top 10 features ranked by their mutual information scores are shown below:

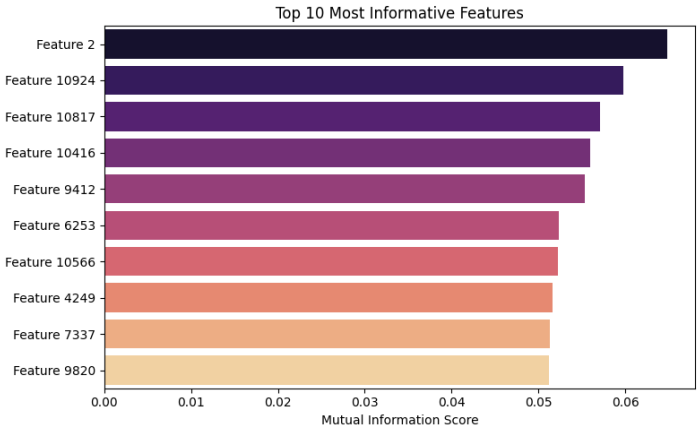

Figure 5: Top 10 Most Informative Features Based on Mutual Information.

3.1 Feature Distribution Analysis

To better understand the relevance of the selected features, we analyzed the distribution of several top-ranked features across the different flower classes.

Feature 2 shows class-dependent variations: *daisy* and *dandelion* have slightly higher median values compared to other classes, while *sunflower* and *tulip* show lower distributions. Although overlaps exist, this feature captures useful separation trends that contribute to classification.

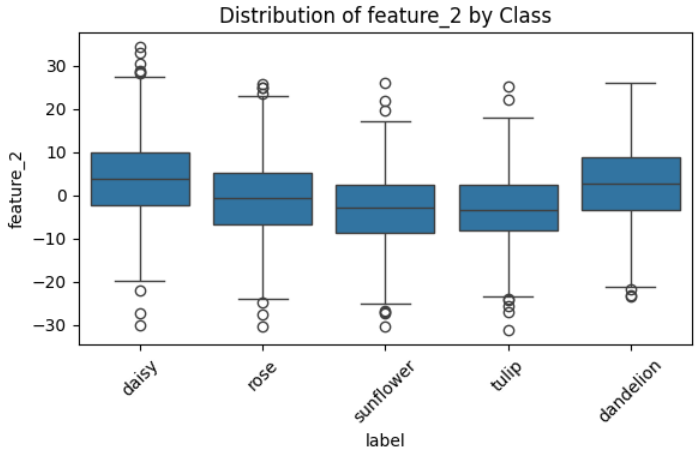

Figure 6: Distribution of Feature 2 by Flower Class.

Feature 297 exhibits a more compressed distribution across all classes. Most classes have feature values clustered near zero, with only minor variations. This suggests that while this feature was selected by the model, its individual discriminative power may be limited and it might be useful mainly through interactions with other features.

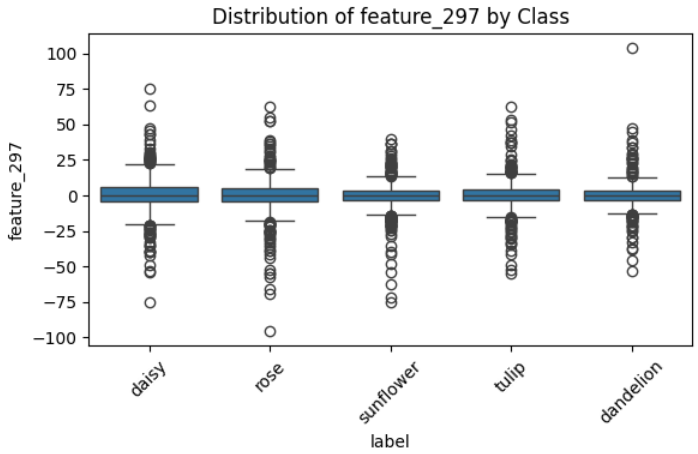

Figure 7: Distribution of Feature 297 by Flower Class.

Feature 347 behaves similarly to Feature 297, showing tightly centered distributions around zero for all classes, with a few outliers. Its selection highlights the model's ability to capture subtle, potentially nonlinear relationships between features that are not easily visible through direct analysis.

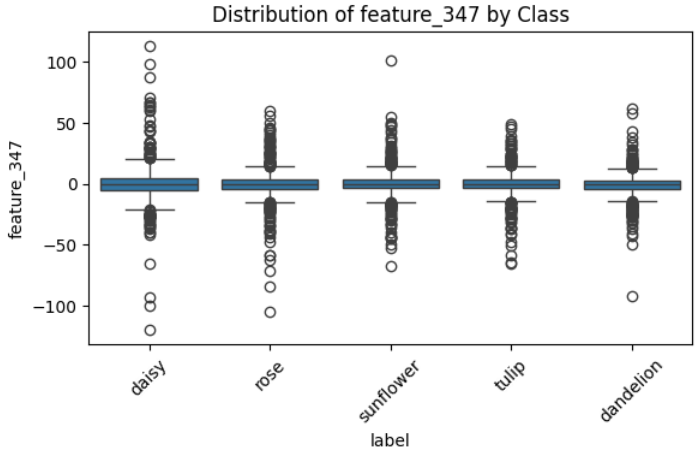

Figure 8: Distribution of Feature 347 by Flower Class.

This analysis emphasizes the importance of combining multiple features to achieve effective class separation, as some features individually offer limited discriminative power but contribute significantly when used together in nonlinear models.

4 Model Training

In this section, we train and evaluate four classical machine learning models: K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Machine (SVM), and Naive Bayes. Each model was trained on the selected features obtained after dimensionality reduction, polynomial expansion, and feature selection. Grid search was used to optimize hyperparameters where applicable.

4.1 K-Nearest Neighbors (KNN)

We trained KNN using Manhattan distance and performed a grid search over the number of neighbors and weight functions.

- **Best Parameters:** n_neighbors = 7, weights = distance

- **Accuracy:** 27.1%
- **Precision:** 30.0%
- **Recall:** 27.5%
- **Training Time:** 4.61 seconds

4.2 Logistic Regression

Logistic Regression was optimized using L2 regularization and hyperparameter tuning for regularization strength.

- **Best Parameters:** C = 0.1, max_iter = 500
- **Accuracy:** 29.3%
- **Precision:** 28.8%
- **Recall:** 28.9%
- **Training Time:** 0.81 seconds

4.3 Support Vector Machine (SVM)

We used an SVM with an RBF kernel and performed grid search over the regularization and kernel parameters.

- **Best Parameters:** C = 1, gamma = scale
- **Accuracy:** 33.8%
- **Precision:** 33.7%
- **Recall:** 33.7%
- **Training Time:** 23.22 seconds

4.4 Naive Bayes

Gaussian Naive Bayes was trained without hyperparameter tuning. It served as a simple baseline.

- **Accuracy:** 30.6%
- **Precision:** 28.1%
- **Recall:** 30.2%
- **Training Time:** 0.01 seconds

4.5 Training Summary

- **SVM** achieved the best overall performance, though at the cost of significantly longer training time.
- **Naive Bayes** was surprisingly competitive given its simplicity and no tuning.
- **KNN** performed the worst, likely due to limitations in handling complex, high-dimensional feature spaces.
- These results provide a baseline for further evaluation and ensemble modeling.

5 Model Evaluation

In this section, we evaluate the performance of each classification model using accuracy, precision, recall, and confusion matrices. Each confusion matrix visualizes how well the model classified each flower class, with the diagonal values representing correct predictions and off-diagonal values indicating misclassifications.

5.1 K-Nearest Neighbors (KNN)

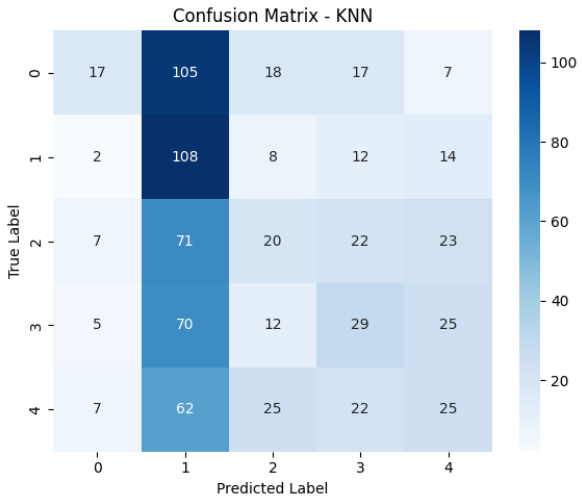

Figure 9: Confusion Matrix - KNN

KNN achieved an accuracy of 27.1%, with precision and recall scores of 30.0% and 27.5%, respectively. The model correctly classified a majority of class 1 (**Dandelion**) samples, but significantly confused classes 2 (*rose*), 3 (**Sunflower**), and 4 (**Tulip**) with class 1. This behavior is common in high-dimensional settings where distances between points become less meaningful. The optimal parameters were $k = 7$ and `weights='distance'`.

5.2 Logistic Regression

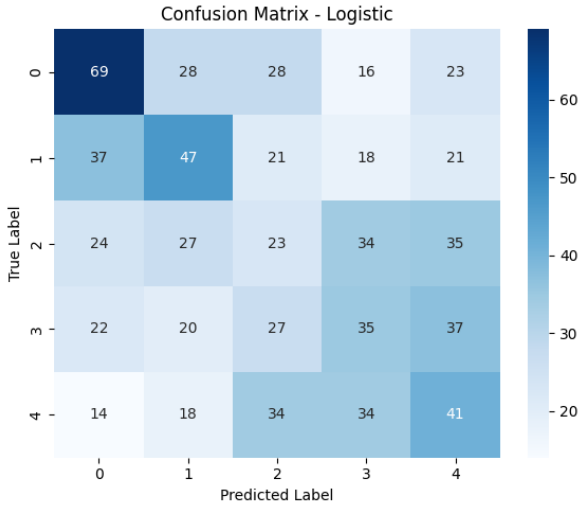

Figure 10: Confusion Matrix - Logistic Regression

With an accuracy of 29.3%, Logistic Regression demonstrates a relatively uniform confusion across all classes. Class 0 (**Daisy**) and class 1 are reasonably well-classified, but classes 2, 3, and 4 are frequently misclassified as each other. This outcome reflects the limitations of a linear model on a non-linearly separable problem. Best parameters were `C=0.1`, `max_iter=500`.

5.3 Support Vector Machine (SVM)

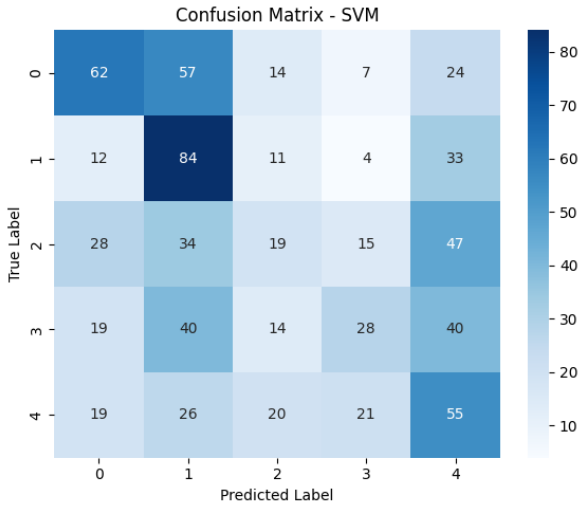

Figure 11: Confusion Matrix - SVM (RBF Kernel)

SVM achieved the best individual performance with an accuracy of 33.8%, precision of 33.7%, and recall of 33.7%. It handles class 1 (**Dandelion**) well, but classes 2 and 3 still exhibit substantial confusion with one another and with class 4. The use of an RBF kernel allows better non-linear boundaries, explaining its improved performance over logistic regression.

5.4 Naive Bayes

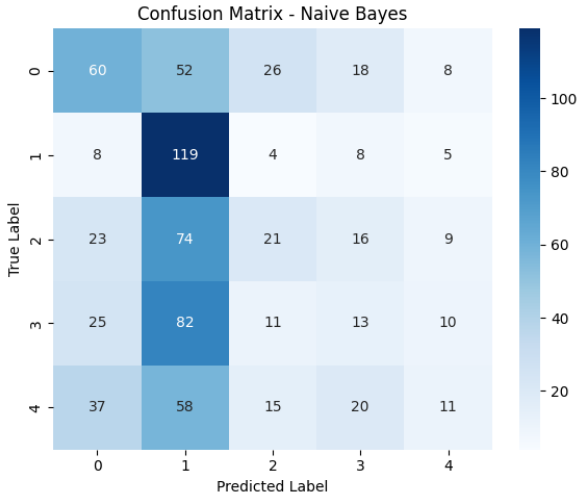

Figure 12: Confusion Matrix - Naive Bayes

Naive Bayes performed better than expected given its simplicity, achieving 30.6% accuracy. It excelled on class 1, but struggled particularly with classes 3 and 4, often misclassifying them as class 1. This

behavior stems from the assumption of feature independence, which does not hold well in image data with correlated pixels.

5.5 Summary

- **KNN** shows a strong bias toward class 1, indicating that local pixel similarities tend to favor that class.
- **Logistic Regression** offers balanced, yet weak, performance due to the linear nature of its decision boundary.
- **SVM** handles non-linearity better and shows improved accuracy across most classes.
- **Naive Bayes** provides a solid baseline, surprisingly competitive despite its naive assumptions.

5.6 Weighted Voting Ensemble

To combine the strengths of the individual models, we constructed a weighted soft-voting ensemble classifier. Each model's output was assigned a weight proportional to its individual accuracy score, allowing the ensemble to give more influence to stronger models while still benefiting from the diversity of all classifiers.
The normalized weights were calculated based on each model's accuracy as follows:

- KNN: 0.2246
- Logistic Regression: 0.2427
- SVM: 0.2799
- Naive Bayes: 0.2528

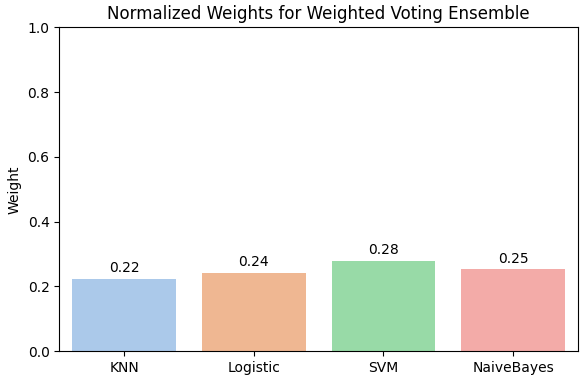

Figure 13: Normalized Weights for Weighted Voting Ensemble.

The ensemble classifier achieved the following performance on the test set:

- Accuracy: **31.2%**
- Precision (Macro): **29.3%**
- Recall (Macro): **30.9%**

Although the ensemble does not dramatically outperform the individual models, it shows slightly more balanced performance, benefiting from the complementary strengths of the base classifiers.
The confusion matrix below illustrates the distribution of predictions by the ensemble model:

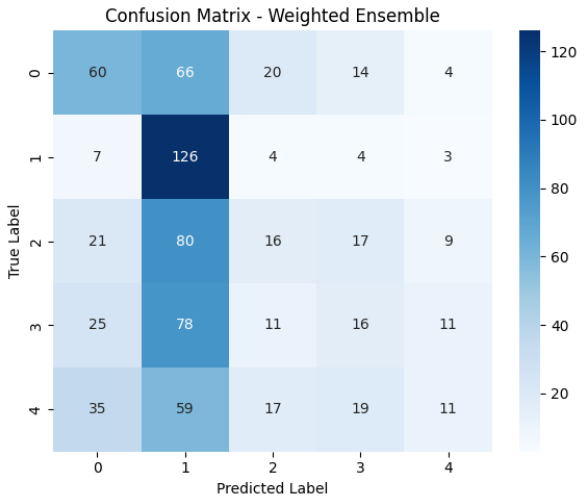

Figure 14: Confusion Matrix - Weighted Ensemble.

From the confusion matrix, it is evident that class *dandelion* (label 1) is most accurately predicted, with 126 correct classifications. However, significant confusion remains among classes such as *daisy*, *rose*, and *tulip*, which indicates overlapping patterns in the feature space even after ensemble voting. These results support the idea that flower classes with more distinctive textures and pixel patterns (e.g., dandelions) are easier to separate, whereas visually similar classes remain challenging.

6 Model Comparison

To understand the relative performance of each classification model, we compared their accuracy, precision, and recall scores. Figure **??** presents a bar chart that visually summarizes these metrics across all models, including the weighted ensemble.

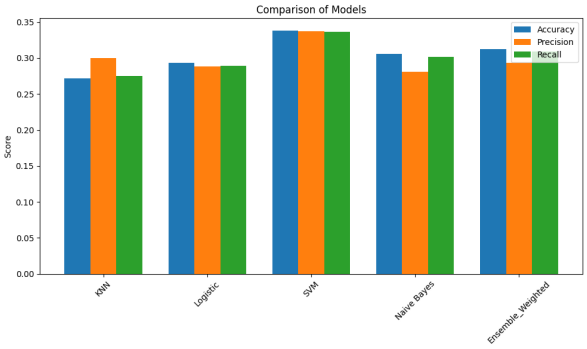

Figure 15: Comparison of Accuracy, Precision, and Recall Across Models.

Observations:

- **SVM** achieved the highest individual accuracy (**0.338**) and recall, indicating it was the best standalone model for overall classification performance.
- **KNN** showed relatively low accuracy but had the highest precision (**0.300**), suggesting that when it predicted a class, it was likely to be correct, but it failed to recall many true examples.
- **Naive Bayes** achieved fast training time with reasonable scores but was not the top performer in any metric.
- **Logistic Regression** offered balanced metrics across all three dimensions, but didn't outperform the others.

- The **Weighted Ensemble** did not outperform SVM in accuracy, but it showed competitive results across all metrics, demonstrating its ability to balance the strengths of individual models.

Overall, the ensemble provides more robust and stable performance than most individual models, especially in multi-class tasks where consistency across metrics is desirable.

7 Training Time Comparison

Beyond accuracy and precision, it is also important to evaluate the computational efficiency of each model. Figure **??** displays the training time in seconds required for each model.

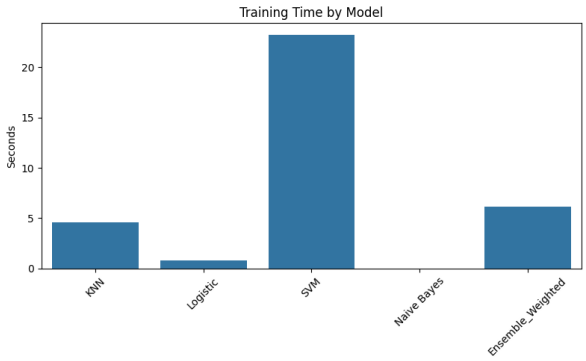

Figure 16: Training Time by Model.

Observations:

- **Naive Bayes** was the fastest model, completing training in a fraction of a second, due to its simple probabilistic nature.
- **Logistic Regression** also trained very quickly (less than one second), making it efficient for real-time or large-scale applications.
- **KNN** required a moderate training time, mostly due to internal distance calculations and parameter tuning via grid search.
- **SVM** had the longest training time, exceeding 20 seconds, which reflects the computational cost of RBF kernel optimization and hyperparameter tuning.
- **Weighted Ensemble** took slightly more time than KNN and Logistic Regression, as it involves training and combining multiple classifiers.

This comparison shows that while complex models like SVM offer better accuracy, they come at the cost of significantly longer training times. For time-sensitive applications, simpler models like Naive Bayes or Logistic Regression may be more suitable.

8 Error Analysis and Model Diagnostics

To better understand where the ensemble model struggles, we conducted an error analysis using the confusion matrix. By identifying the most commonly confused classes, we can gain insight into which flower types are harder to distinguish.

8.1 Most Frequently Confused Classes

We analyzed the confusion matrix of the ensemble model and zeroed out the diagonal to focus solely on misclassifications. The class pair with the highest confusion was identified as:

Most confused pair: *rose* $\rightarrow$ *dandelion* (80 misclassifications)

This suggests that the model frequently misclassifies images of roses as dandelions, possibly due to similarities in their grayscale texture, petal structure, or overlapping feature values in the reduced feature space.

8.2 Visual Analysis Using t-SNE

To further explore the confusion between these two classes, we used t-distributed Stochastic Neighbor Embedding (t-SNE) to project their feature vectors into a 2D space. This dimensionality reduction technique helps visualize how closely instances of these two classes are positioned relative to one another.

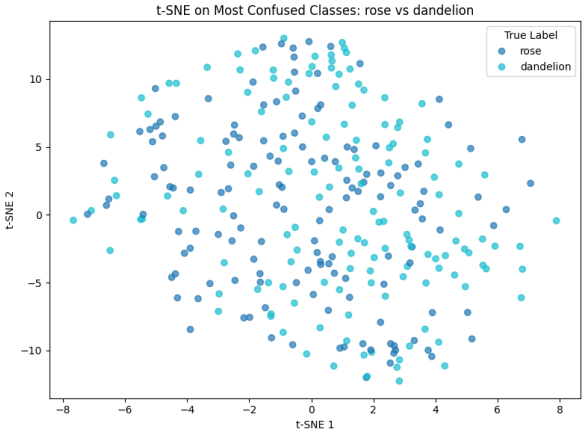

Figure 17: t-SNE Visualization of Most Confused Classes: *rose* vs *dandelion*.

As shown in the figure, there is substantial overlap between the *rose* and *dandelion* samples in the 2D space, making it difficult for classifiers to draw a clear boundary. This reinforces the earlier observation from PCA that certain flower classes are not linearly separable and require more expressive, nonlinear modeling techniques.

8.3 Class-wise Error Counts

To quantify model performance per class, we analyzed the confusion matrix of the ensemble model by computing the total number of misclassifications for each true class (i.e., excluding the diagonal). This helps identify which flower types are more prone to classification errors.

- **Tulip:** 130 errors
- **Rose:** 127 errors
- **Sunflower:** 125 errors
- **Daisy:** 104 errors
- **Dandelion:** 18 errors

From these results, it is evident that *tulip*, *rose*, and *sunflower* are the most error-prone classes, which aligns with their overlapping structure observed in previous PCA and t-SNE visualizations. In contrast, *dandelion* stands out as the most consistently classified class.

8.4 Hardest Test Samples

To further analyze classification difficulty, we examined individual test samples misclassified by multiple models. A sample was considered "hard" if it was misclassified by at least 3 out of the 4 models (KNN, Logistic Regression, SVM, Naive Bayes). The distribution of these hardest test cases by true label is shown below:

- **Rose:** 123 hard examples
- **Sunflower:** 112 hard examples
- **Daisy:** 99 hard examples
- **Tulip:** 98 hard examples
- **Dandelion:** 36 hard examples

This result reinforces the earlier findings: *rose* and *sunflower* classes are especially challenging, not just on average but also at the sample level. These classes likely have higher intra-class variation or share visual features with others, making them harder to distinguish reliably across models.

8.5 Visual Analysis of Hardest Samples

To gain further insights into the most challenging test samples—those misclassified by at least three out of the four models—we performed a two-dimensional t-SNE projection.

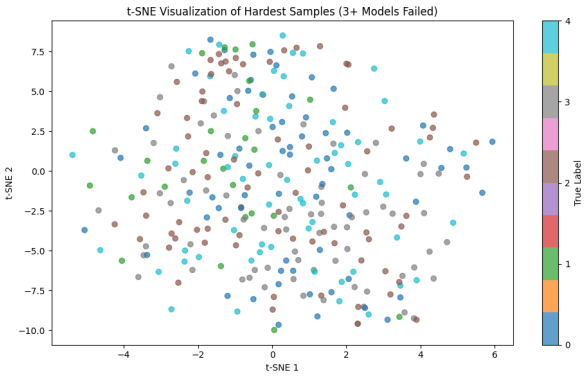

Figure 18: t-SNE Visualization of Hardest Test Samples (Misclassified by ≥ 3 Models).

The visualization reveals that these difficult samples are scattered throughout the 2D space without forming distinct clusters per class. There is significant overlap between flowers such as *rose*, *sunflower*, and *tulip*, suggesting that these categories share common visual characteristics that confuse all models. This lack of separability may arise from subtle differences in texture, structure, or image quality.

In particular, flowers with similar petal arrangements or grayscale intensity distributions tend to occupy overlapping regions, making class boundaries hard to define. This observation emphasizes the limitations of traditional feature-based machine learning for fine-grained visual classification. In future work, incorporating domain-specific features or leveraging deep learning methods that can automatically extract higher-level visual abstractions might help better distinguish between these challenging samples.

9 Conclusion

In this project, we developed a flower image classification pipeline using classical machine learning techniques. We applied preprocessing, feature engineering (including PCA and polynomial expansion), and mutual information-based feature selection to build a meaningful feature representation. Four models—KNN, Logistic Regression, SVM, and Naive Bayes—were trained, tuned, and evaluated. Among them, SVM achieved the highest accuracy, while the weighted ensemble model showed competitive performance by combining the strengths of all individual classifiers. Visualizations of feature distributions, confusion matrices, and t-SNE projections helped us diagnose model behavior, identify confused class pairs (notably *rose* vs. *dandelion*), and understand which classes posed greater classification challenges. While the models performed reasonably well, the overall accuracy remained modest due to

the high visual similarity between some flower classes and the limited expressiveness of hand-crafted features. Future improvements could include the integration of deep learning models (e.g., CNNs), advanced data augmentation, or feature extraction based on pre-trained neural networks.
Overall, the project demonstrated the potential and limitations of classical machine learning for image classification tasks, and provided a foundation for future experimentation with more powerful learning techniques.