# Classifying Images of Diseased Cassava Plants

**Clifton Harris*****
*clifton.harris@berkeley.edu*

**Jacob Jones*****
*jacoblhjones@berkeley.edu*

**Rodrigo Cochran*****
*rodrigocochran@berkeley.edu*

**Tamar Brand-Perez*****
*terez@berkeley.edu*

**Associated with the University of California, Berkeley*
***Associated with the Berkeley School of Information*
****Associated with the Masters of Information and Data Science*

*Order is random and does not indicate weight of contribution.*

## Table of Contents:


| | |
|---|---|
| Abstract | 3 |
| 1. Introduction | 4 |
| 2. Feature Extraction | 6 |
| 3. Classification | 13 |
| 4. Generalizability | 20 |
| 5. Efficiency vs. Accuracy | 21 |
| 6. Discussion and Limitations | 22 |
| 7. Conclusion | 26 |
| 8. References | 27 |
| 9. Appendices | 28 |

## Abstract

Cassava, a crucial crop for global food security, faces threats from diseases that impair yield and quality. This study develops a classifier to identify Cassava plant diseases using leaf images, combining feature extraction methods like Histogram of Oriented Gradients (HOG), Canny edge detection, color histograms, and Fourier transforms with the self-supervised transformer Dinov2. The dataset, sourced from Kaggle, includes high-resolution images divided into five classes: four disease categories and healthy leaves. Class imbalance is addressed through sampling strategies to ensure fair representation. Principal Component Analysis (PCA) reduced dimensionality, retaining 95% variance to mitigate overfitting and reduce computational complexity. Tree-based algorithms, XGBoost and Random Forest, were compared: XGBoost achieved the highest accuracy (80.88%) on the unbalanced dataset with the full feature set. Using PCA-reduced features and a balanced dataset, a model achieved 59.56% accuracy while reducing training time to 1.9 seconds per trial. These results highlight trade-offs between accuracy and efficiency, offering flexibility for varying resource constraints. The models generalized effectively, with test set performance closely matching validation results. Challenges included reduced interpretability with Dinov2, difficulty detecting non-visual symptoms, and high computational demands, limiting scalability in resource-constrained settings.

# 1. Introduction

## Problem Statement

Cassava, a staple crop for millions of people across tropical regions, is an essential component of global food security and agricultural economies. In 2023, the global Cassava processing market reached a volume of approximately 285 million metric tons, with projections to grow at a compound annual growth rate (CAGR) of 2.9%, reaching 368.5 million metric tons by 2032 ("Global Cassava Processing Market Report and Forecast 2022-2027"). For comparison, the global output of wheat was 785 million metric tons and rice 523.5 million metric tons for the period of 2023/2024 ("Wheat | USDA Foreign Agricultural Service", "Rice | USDA Foreign Agricultural Service"). The Cassava tuber, a potato like root, cannot be consumed raw as it contains compounds that produce cyanide, necessitating proper processing for safe consumption, however, it is a key ingredient in products such as flour, alcohol and tapioca (Alitubeera et al.).

While Cassava is a vital crop, its cultivation is challenged by its susceptibility to various diseases, which threaten both yield and quality (Valentine Otang Ntui et al.). Addressing these challenges is crucial to sustaining Cassava production and mitigating the socioeconomic impacts on communities dependent on this crop.

This project aims to develop an accurate classifier capable of identifying Cassava plant diseases based on leaf images. By using a combination of simple feature extraction together with advanced machine learning techniques, the study seeks to enhance the detection and classification of Cassava diseases, offering a tool to aid farmers and agricultural stakeholders in early diagnosis and disease management.

## Dataset Information

The dataset used in this study is sourced from Kaggle and comprises high resolution images (800 x 600 pixels) categorized into five classes:

Cassava Bacterial Blight (CBB) class 0, causes water soaked lesions that often turn brown with necrotic edges. The leaves wilt and drop prematurely. In severe cases it looks like the plant is burnt. This disease typically starts on the leaf edges and spreads inward. It will severely impact tuber production.

Cassava Brown Streak Disease (CBSD) class 1, have blotches or streaks along the veins. Leaves remain generally flat but in severe cases may be slightly curled. This disease is most prominent near the major veins. Yellowing is most visible on older leaves on the lower half of the plant. Roots rot which significantly reduces yield.

Cassava Green Mottle (CGM) class 2, leads to uneven green patches and uneven coloration. The shape of the leaf is maintained. The location is on the upper surface of the leaf. This disease reduces photosynthetic efficiency and impacts tuber quality.

Cassava Mosaic Disease (CMD) class 3, is when the leaves display a yellow and green mosaic pattern. Leaves may be twisted, curled, or deformed. This disease is widespread across the plant and it leads to less tuber yield.

Healthy Cassava leaves class 4.

Below is a sample of images illustrating these variations:


While the dataset contains a rich diversity of images, it suffers from class imbalance, with certain conditions underrepresented. In the process of creating the different models, a sampling strategy was implemented in some of them, to ensure an equal number of samples per category, enhancing the robustness and generalizability of those models. Additionally, the dataset exhibited noise in the form of similar color distributions across classes, overlapping symptoms, and varying backgrounds, which complicated the extraction of distinct features. Variations in the number of leaves per image, their orientation, and inconsistent lighting conditions further added to the challenges of accurate classification.

The proposed models integrate advanced feature extraction methods and state-of-the-art classification models to optimize for two outcomes: one model optimized for accuracy and the other model optimized for efficiency. Techniques such as Histogram of Oriented Gradients (HOG) and color detection are combined with pre-trained deep learning Dinov2 model, offering a comprehensive solution for Cassava disease classification. This paper presents the dataset characteristics, methodology, and results of the classification system, providing valuable insights for agricultural disease management and contributing to the broader effort to improve global Cassava production.

## 2. Feature Extraction

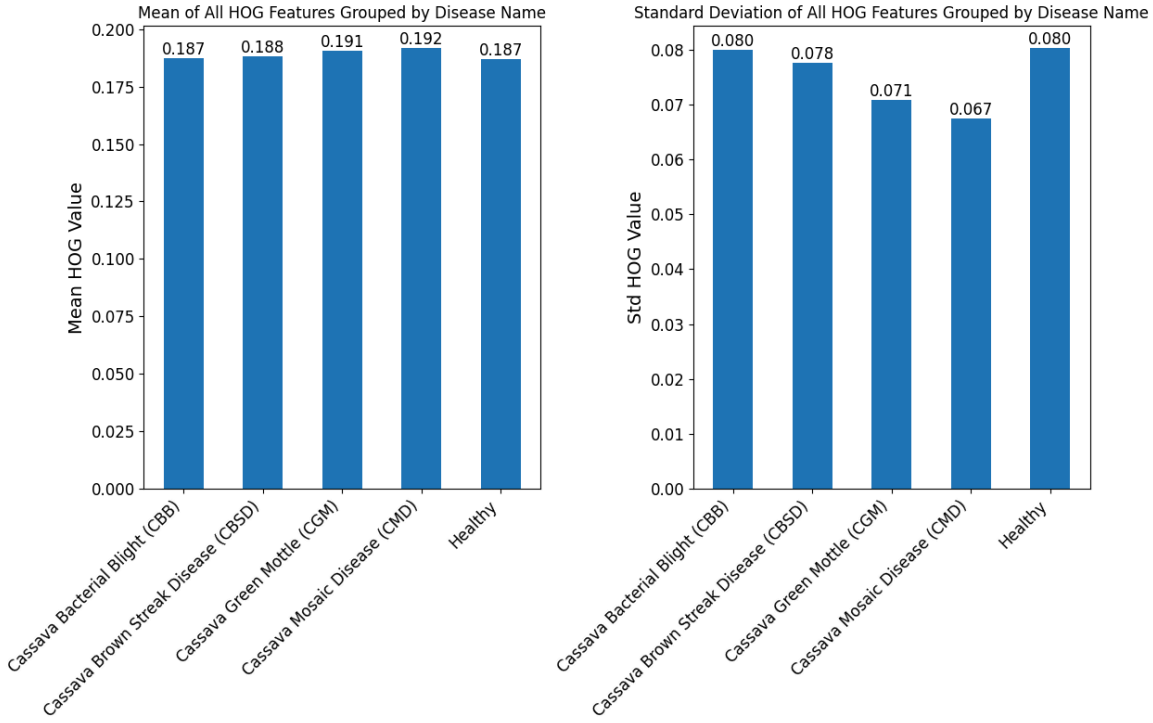
The simple features utilized in this study included HOG for detecting leaf shape, Canny edge detection for identifying changes in the shape of leaves as well as vein patterns, color histograms (means and standard deviations) for analyzing changes in leaf color, and Fourier transform for evaluating changes in leaf texture. To ensure efficiency, consistency, and reduced computational complexity during feature extraction, all images were resized to 224 x 224 pixels. Additionally, images used for HOG, edge detection, and Fourier transform were converted to grayscale.

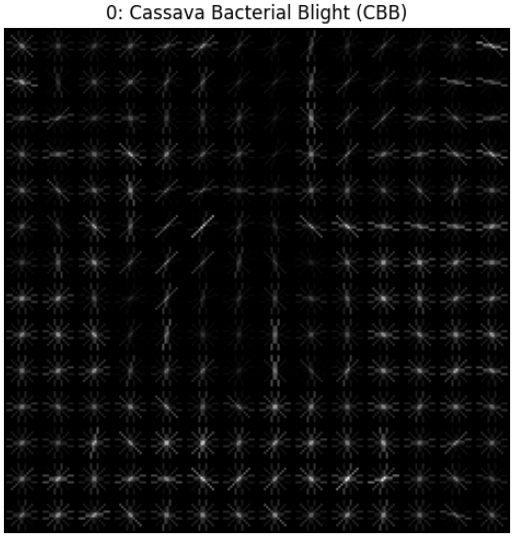
### Additional Computational Considerations

Given the nature of the data, data manipulation can be computationally expensive; image loading and feature extraction can be very costly operations. To reduce compute time for our i/o bound processes (like image loading), we employed multi-threading to parallelize the process and saw significant speed increase of 17.5x on the Google Colab default A100 GPU machine. For feature extraction, which is a CPU-bound/GPU-bound task, we employed multiprocessing and saw a significant speed increase of 16x on the same Google Colab default A100 GPU machine.

## Histogram of Oriented Gradients (HOG)

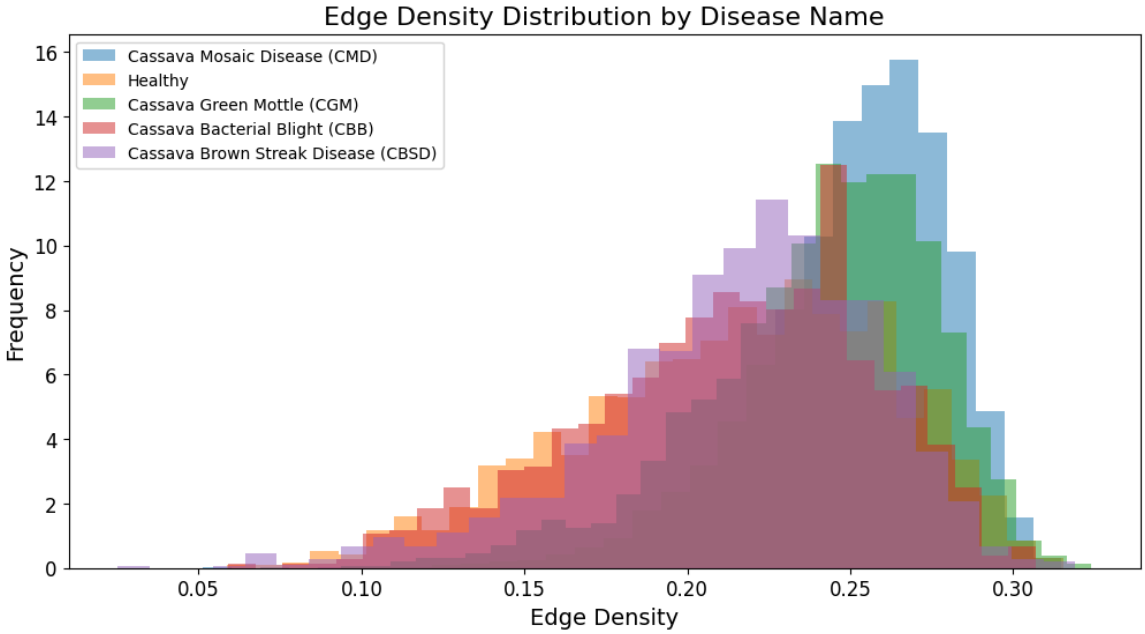
HOG was employed to capture the shape of the leaves. The mean values for HOG features were relatively consistent across classes. However, when examining the standard deviations, Cassava Mosaic Disease (class 3) and Green Mottle (class 2) exhibited lower standard deviations, suggesting more uniform leaf textures in these categories.

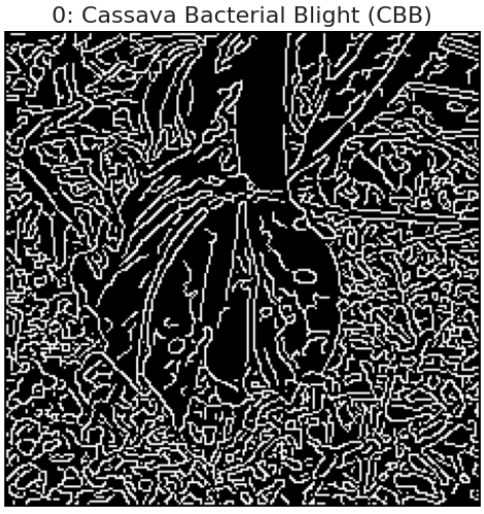


### Canny Edge Detection

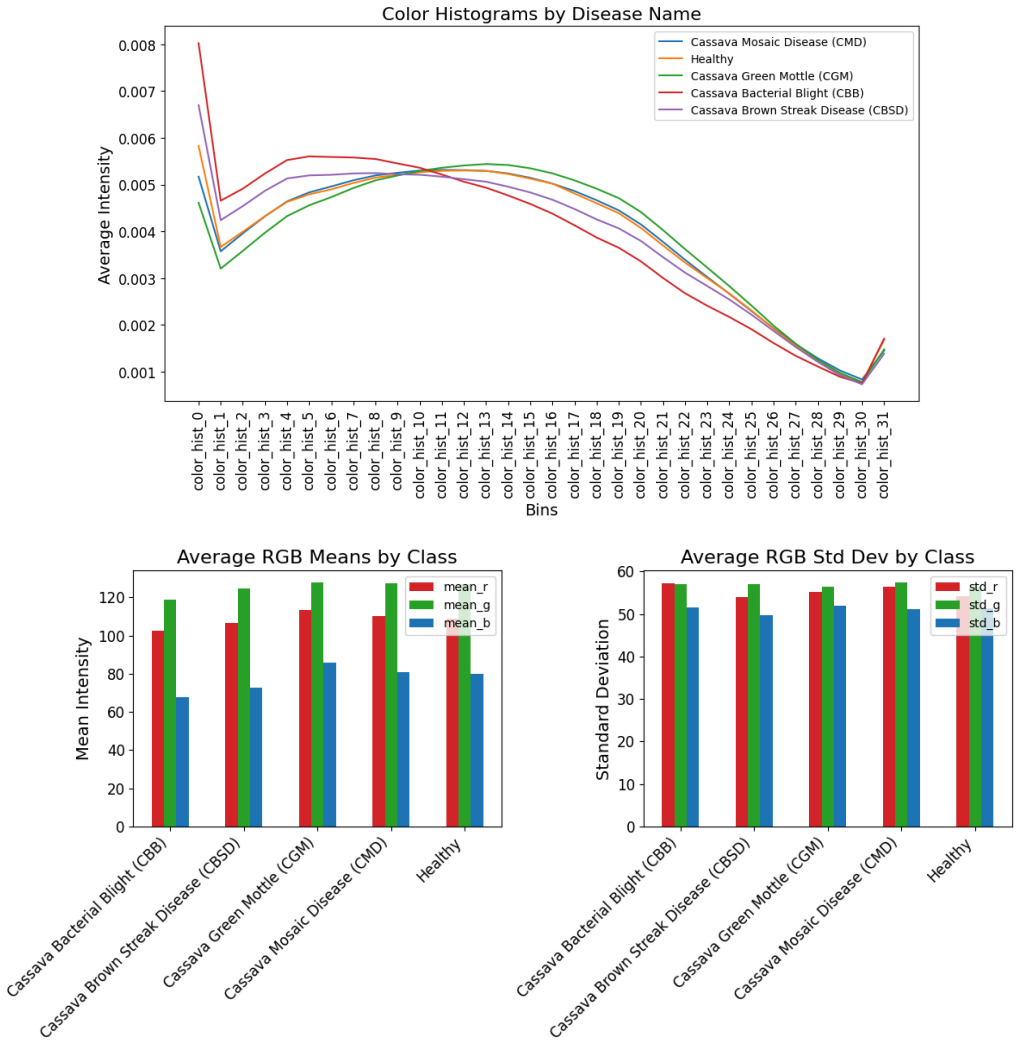
Canny edge detection was used to detect changes in leaf shape. For instance, one of the symptoms of Cassava Mosaic Disease (class 3) is the deformation of leaf shapes, making this feature particularly relevant. Additionally, we hypothesized that changes in vein shading due to Brown Streak Disease (class 1) might be captured through higher edge density. However, this expectation was not met. Instead, we observed slightly higher edge densities for Cassava Mosaic Disease (class 3) and Green Mottle (class 2). A higher edge density in Canny edge detection indicates a larger number of high-contrast transitions, which can arise from fine details, textures, or simply images with more leaves. Since the number of leaves varied across images, this feature alone was insufficient to distinguish between classes with high confidence.



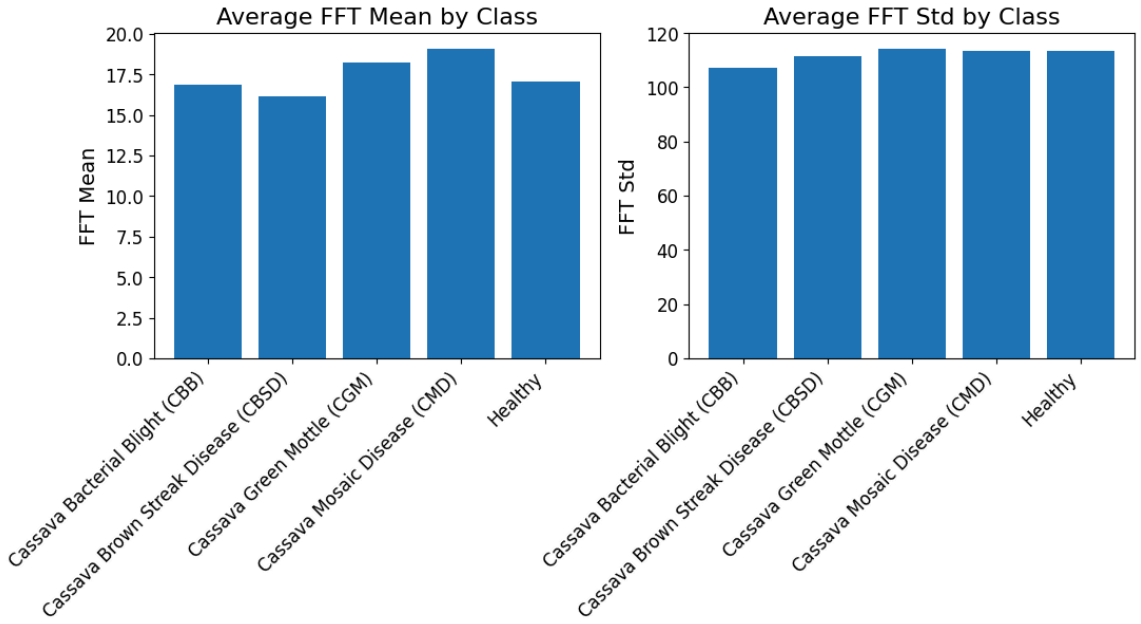
## Color Intensity

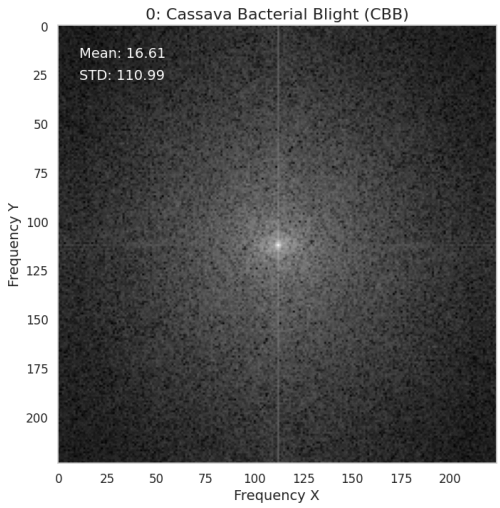
Color histograms (means and standard deviations) were used to analyze color intensities by disease category. The results showed minimal differences across classes. Bacterial Blight (class 0) displayed slightly higher intensity in lower bins and lower intensity in higher bins, but the averages remained similar across all classes. This aligns with the observation that images in the dataset generally have similar color schemes—primarily green, yellow, and brown tones in both leaves and backgrounds. Consequently, this feature alone did not provide substantial discriminatory power.



**Fourier Transform**

The Fourier transform was applied to detect textures and patterns in the images. Average frequencies were slightly higher for Cassava Mosaic Disease (class 3) and Green Mottle (class 2), while standard deviations were consistent across classes.



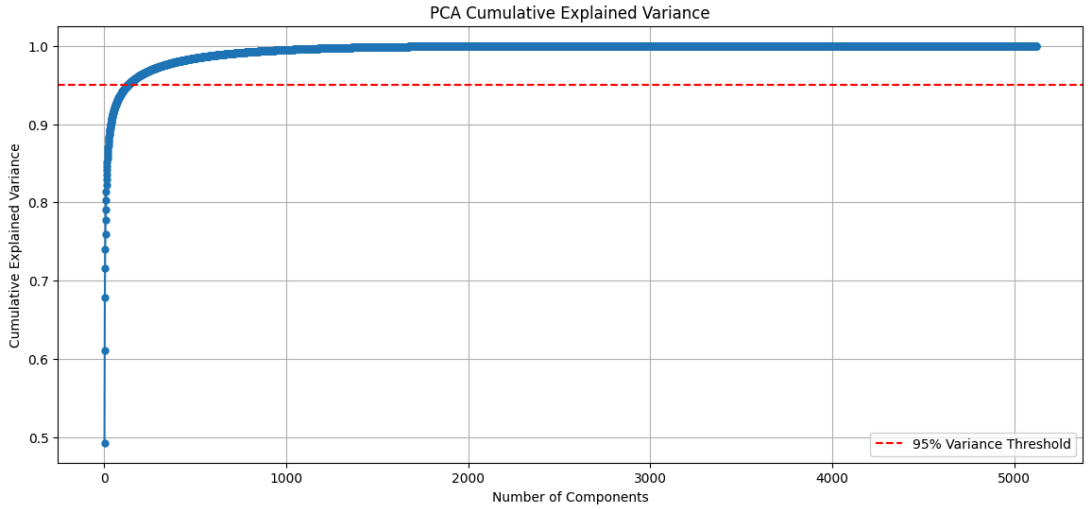

Overall when looking at each of the simple features, slight differences were present but those differences were not enough to enable classification to the 5 classes with good accuracy.

## DinoV2

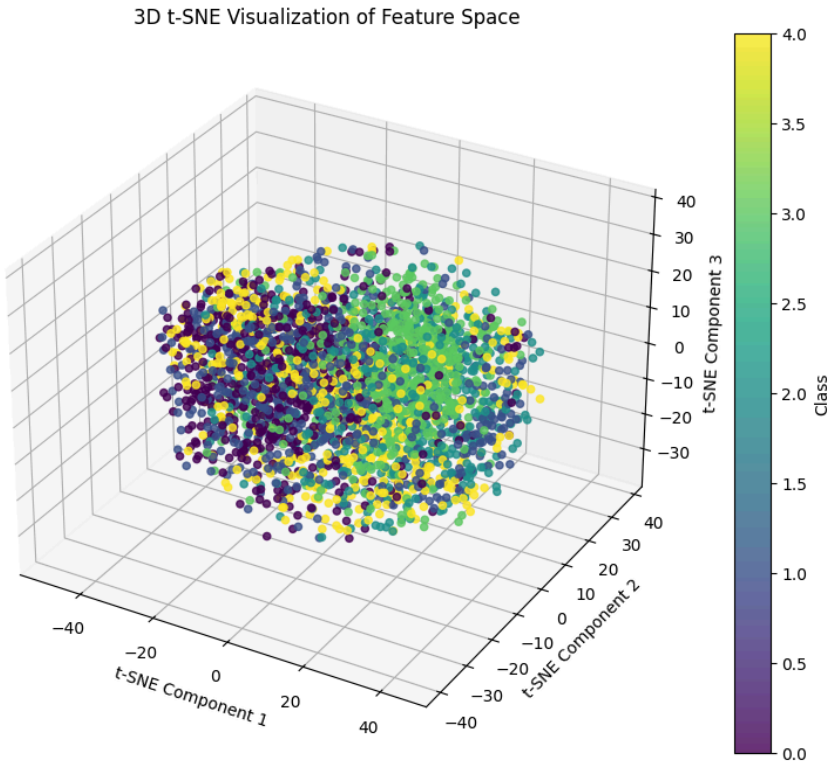
To supplement these simple features, we used Dinov2 to extract a complex feature vector from the images for classification. Dinov2, a self-supervised model developed by Meta, was pre-trained on 142 million images. It relies on a self-supervised learning paradigm that incorporates both patch-level and full-image representations through two separate ViT models that learn collaboratively. This robust pre-training enables Dinov2 to capture a wide range of visual features and makes the model suitable for a wide range of computer vision tasks such as classification, semantic segmentation, depth estimation, and sparse matching. This versatility is why we selected DINOv2. For this task, we extracted the [CLS] token from the final hidden state of Dinov2, which encodes a general representation of the image.

## Principal Component Analysis

After combining all of these features together we had 5,121 total features. To reduce dimensionality and simplify computing we used PCA and reduced the number of features to 130 while maintaining 95% of the cumulative explained variance. We then visualized the feature space using a three-dimensional t-SNE plot. While some separation between classes is evident, the complexity of the data results in substantial overlap among the classes, even in three dimensions.




PCA Cumulative Explained variance plot displaying that 95% of variance can be explained with 130 features.




Three dimensional t-SNE visualization of our feature space.

## 3. Classification

Identifying diseases based on image data naturally presents itself as a classification problem. For this project, we employed 2 tree-based algorithms for training classification models using different configurations of the input dataset.

**Overview**

XGBoost and Random Forest were the most natural fit for the problem given the nature of our data and the problem at-hand. Some of the most important challenges we face in this problem are classifying images into one of several classes, the high dimensionality of the data, and the inherent noise within each image.

XGBoost and Random Forest are both robust algorithms suited for this task due to their ability to infer multiple classes, manage high dimensional data, and handle noise. Additionally, their tree-based design can capture and model non-linear relationships which are often present in high dimensional data that is extracted from distribution-based features like HOG. This capability isn't unique to non-linear relationships, as it also applies to noise in data; the algorithms are suited for discounting noise by having multiple iterations over a variety of different trees. XGBoost uses boosting to incrementally improve predictions over each tree sequentially and Random Forest uses bagging (bootstrap aggregation) to aggregate information across each parallel tree. Lastly, they both are naturally parallelizable due to the independence of certain trees.

**Implementation**

Our model consists of several steps in order to prepare the dataset for training, validation, and evaluation. The first step is to take a subset of the data and hold it out for evaluation at the end; it is not used during the training or validation periods. It is important to do this to eliminate chances of data leakage. Then we split the training data into a smaller training set and validation set; these will both be used during training and validation. Additionally, we apply dimensionality reduction using principal component analysis to lower the probability of overfitting the model to the training data.

**Training, Validation, and Test Datasets**

| Dataset | Initial Shape | Train Shape | Test Shape | Validation Shape |
|---|---|---|---|---|
| Initial Dataset | (17938, 5121) | (14350, 5121) | (3588, 5121) | None |
| Full Feature Set - Unbalanced | (14350, 5121) | (11480, 5121) | None | (2870, 5121) |
| Full Feature Set - Balanced | (14350, 5121) | (2948, 5121) | None | (737, 5121) |
| PCA Feature Set - | (14350, 130) | (11480, 130) | None | (2870, 130) |

| | | | | |
|---|---|---|---|---|
| Unbalanced | | | | |
| PCA Feature Set - Balanced | (14350, 130) | (2948, 130) | None | (737, 130) |

**Hyperparameter Tuning**

Our implementation employed Bayesian hyperparameter search for approximating the best hyperparameters to train our model. We chose to use Bayesian search for its computational efficiency over grid search and its superior ability to converge to optimal values compared to random search due to its probabilistic-based approach. This method leverages the balance between exploration (searching broadly for potential solutions) and exploitation (refining the best solutions) to identify the most effective hyperparameters. We utilized a Python implementation of Bayesian hyperparameter search from the open-source library HyperOpt.

For XGBoost, the search focused on six hyperparameters:

- **n_estimators**, representing the number of boosting rounds or trees in the ensemble, it was tested using uniform discrete values ranging from 50 to 300 in steps of 10.
- **max_depth**, controlling the maximum depth of each tree and balancing model complexity and overfitting, ranged from 3 to 15 with uniform discrete values.
- **learning_rate**, which shrinks the contribution of each tree to control convergence speed, was explored on a logarithmic scale between 0.001 and 1.0.
- **gamma**, defining the minimum loss reduction required to split a node, was tested with uniform continuous values between 0 and 5.
- **subsample**, dictating the fraction of training data used per boosting round to mitigate overfitting, ranged from 0.5 to 1.0 with uniform continuous values.
- **colsample_bytree**, specifying the fraction of features used to grow each tree, also ranged from 0.5 to 1.0 with uniform continuous values.

For Random Forest, the search focused on four parameters:

- **n_estimators**, the number of trees in the forest, ranged from 50 to 300 in steps of 10 with uniform discrete values.
- **max_depth**, controlling tree complexity, was explored between 5 and 30 in steps of 1 with uniform discrete values.
- **min_samples_split**, the minimum number of samples required to split an internal node, ranged from 2 to 10 in uniform discrete steps.
- **min_samples_leaf**, defining the minimum number of samples required at a leaf node to reduce overfitting, was searched between 1 and 5 using uniform discrete values.

The optimization process used negative accuracy as the minimizing metric, ensuring a focus on improving model performance by minimizing error. This systematic exploration allowed the search method to effectively identify hyperparameter sets tailored for each model, balancing computational cost and predictive accuracy.

**Training the Models**

To ensure robust model training and minimize overfitting, we employed a comprehensive strategy combining data preparation and algorithmic techniques. As mentioned, each dataset was split into training and validation subsets, enabling us to properly evaluate performance in an unbiased manner without risk of data leakage. Class balancing techniques were applied to a subset of the models to address potential biases from class imbalances, ensuring fair representation across all classes in the training and validation datasets. Additionally, given the high dimensionality of the original dataset—where the number of features exceeded the number of training records—overfitting was a significant concern. To mitigate this, Principal Component Analysis (PCA) was utilized to reduce dimensionality, allowing the models to focus on the most meaningful features while discarding noise. We selected the minimal number of features that would capture 95% of the variance. Finally, our hyperparameter search was carefully constrained to avoid configurations that could exacerbate overfitting, such as excessively deep trees. However, due to computational constraints, the search was limited to five trials per model, as eight models were trained in total. However, in a production setting, this would be increased to at least 25 trials to enhance model robustness. This systematic approach balanced the need for computational efficiency with the goal of developing accurate and generalizable models.

In terms of efficiency, training time and cost are the defining metrics for this type of problem; both of which we would like to reduce while minimizing performance loss. All of our models were trained on the same Google Colab default A100 GPU machine.

| **Dataset** | **Model** | **Training Rate** | **Number of Trials** | **Training Time** | **Machine Cost / hour** | **Total Training Cost** |
|---|---|---|---|---|---|---|
| Full Feature Set - Unbalanced | Random Forest | 130.46s/trial | 5 | 00:10:52.3 | $1.18 | $0.2138 |
| Full Feature Set - Balanced | Random Forest | 25.94s/trial | 5 | 00:02:19.7 | $1.18 | $0.0458 |
| PCA Feature Set - Unbalanced | Random Forest | 22.37s/trial | 5 | 00:01:51.9 | $1.18 | $0.0367 |
| PCA Feature Set - Balanced | Random Forest | 4.40s/trial | 5 | 00:00:22.0 | $1.18 | $0.0072 |
| Full Feature Set - Unbalanced | XGBoost | **220.91s/trial** | 5 | 00:18:24.55 | $1.18 | $0.3620 |
| Full Feature Set - Balanced | XGBoost | 54.30s/trial | 5 | 00:04:31.5 | $1.18 | $0.0890 |
| PCA Feature Set - Unbalanced | XGBoost | 5.12s/trial | 5 | 00:00:25.6 | $1.18 | $0.0084 |

| | | | | | | |
|---|---|---|---|---|---|---|
| PCA Feature Set - Balanced | XGBoost | **1.90s/trial** | 5 | 00:00:09.5 | $1.18 | $0.0031 |

**Results**

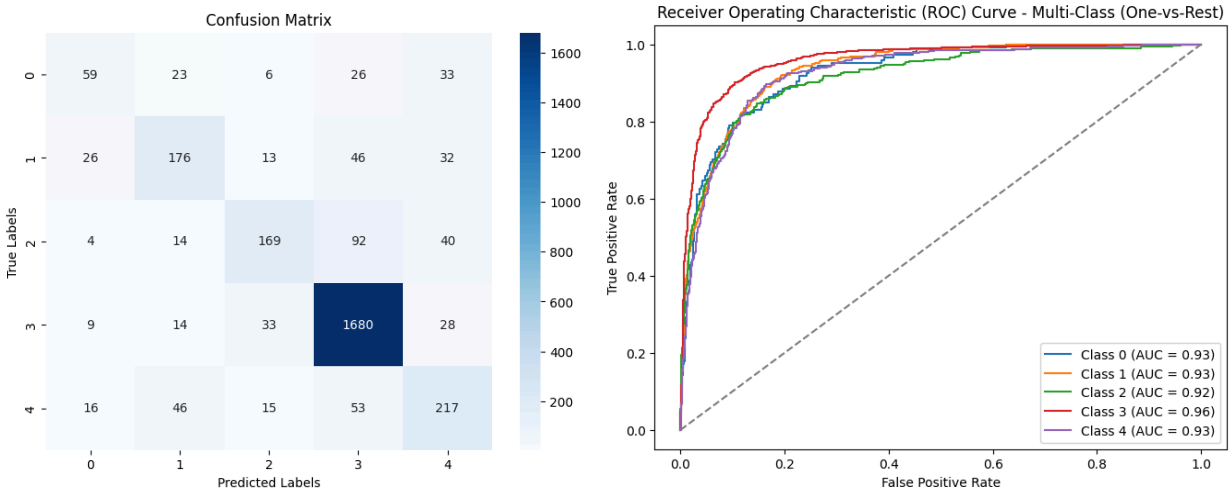
This table shows the overall macro average weighted error metrics for our models evaluated against the test set. Green highlights are for the highest metric for the category and red are for the lowest metric for the category.

| Feature Set | Balanced | Model | Accuracy | Macro Avg Precision | Macro Avg Recall | Macro Avg F1 | Weighted Avg Precision | Weighted Avg Recall | Weighted Avg F1 |
|---|---|---|---|---|---|---|---|---|---|
| Full | TRUE | RF | 0.7007 | 0.5424 | 0.6213 | 0.5605 | 0.7629 | 0.7007 | 0.7211 |
| Full | TRUE | XGB | 0.7054 | 0.5516 | **0.6366** | 0.5764 | 0.7704 | 0.7054 | 0.7268 |
| Full | FALSE | RF | 0.7701 | 0.6145 | 0.6143 | 0.6119 | 0.7732 | 0.7701 | 0.7705 |
| Full | FALSE | XGB | **0.8088** | **0.6776** | 0.6295 | **0.6498** | **0.7989** | **0.8088** | **0.8011** |
| PCA | TRUE | RF | 0.6215 | 0.4623 | 0.5383 | 0.4772 | 0.7103 | 0.6215 | 0.6504 |
| PCA | TRUE | XGB | **0.5956** | **0.4459** | 0.5173 | 0.4615 | 0.7044 | **0.5956** | **0.6309** |
| PCA | FALSE | RF | 0.7149 | 0.5618 | **0.4405** | **0.4640** | **0.6836** | 0.7149 | 0.6757 |
| PCA | FALSE | XGB | 0.7386 | 0.5699 | 0.4810 | 0.5093 | 0.7087 | 0.7386 | 0.7144 |

Below, we will go into the results for the most accurate and the most efficient models. In the interest of maintaining focus on the most important results in this paper, we will provide a link to the notebook that has the outputs saved for all of the models.

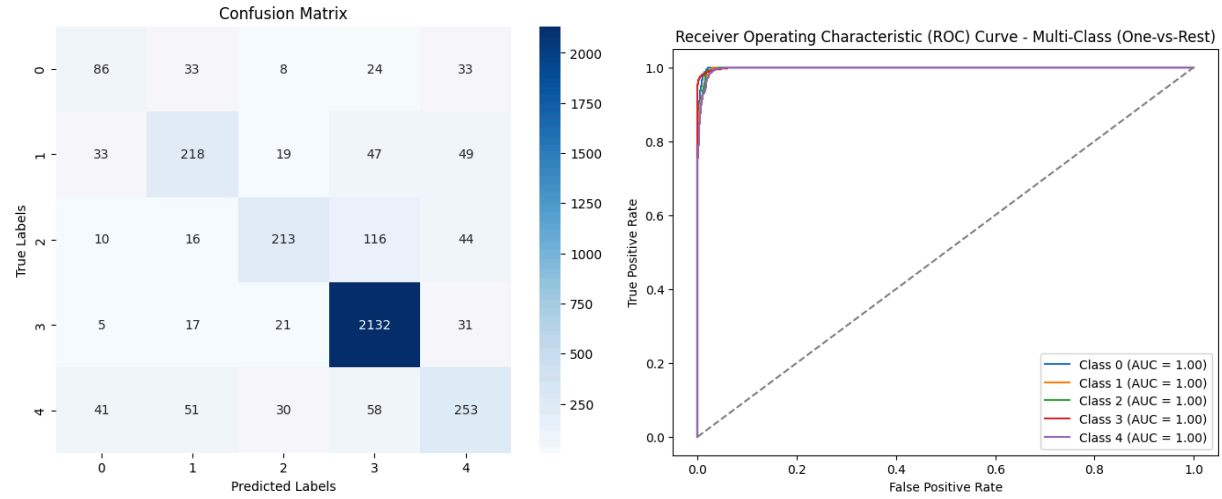
**Most Accurate Model - Unbalanced, Full Feature, XGBoost Model - Training Results**

```
Balanced train-test split: False
Model Algorithm: xgb
X train shape: (11480, 5121)
y train shape: (11480,)
X test shape: (2870, 5121)
y test shape: (2870,)
100%|██████████| 5/5 [18:24<00:00, 220.91s/trial, best loss: -0.0]
Best Parameters: {'colsample_bytree': 0.5568981335918035, 'gamma': 3.8392819888268797, 'learning_rate': 0.35177173
25722587, 'max_depth': 14.0, 'n_estimators': 210.0, 'subsample': 0.9068277038201439}
Final Accuracy: 0.8017421602787457
Classification Report:
                                precision    recall  f1-score   support

     Cassava Bacterial Blight (CBB)       0.52      0.40      0.45       147
Cassava Brown Streak Disease (CBSD)       0.64      0.60      0.62       293
        Cassava Green Mottle (CGM)       0.72      0.53      0.61       319
       Cassava Mosaic Disease (CMD)       0.89      0.95      0.92      1764
                            Healthy       0.62      0.63      0.62       347

                           accuracy                           0.80      2870
                          macro avg       0.68      0.62      0.64      2870
                       weighted avg       0.79      0.80      0.79      2870
```




## Most Accurate Model - Unbalanced, Full Feature, XGBoost Model - Evaluation Results

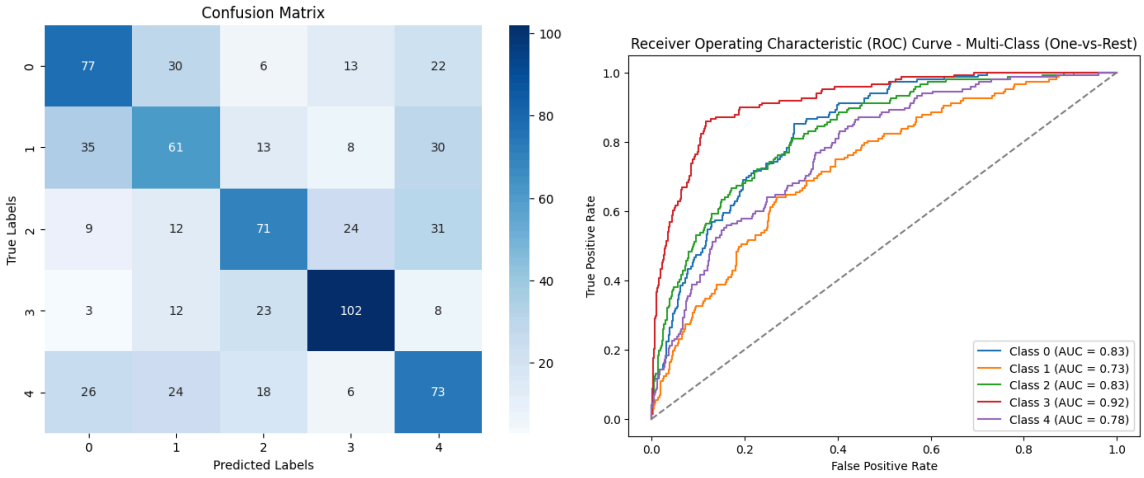
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Cassava Bacterial Blight (CBB) | 0.49 | 0.47 | 0.48 | 184 |
| Cassava Brown Streak Disease (CBSD) | 0.65 | 0.60 | 0.62 | 366 |
| Cassava Green Mottle (CGM) | 0.73 | 0.53 | 0.62 | 399 |
| Cassava Mosaic Disease (CMD) | 0.90 | 0.97 | 0.93 | 2206 |
| Healthy | 0.62 | 0.58 | 0.60 | 433 |
| accuracy | | | 0.81 | 3588 |
| macro avg | 0.68 | 0.63 | 0.65 | 3588 |
| weighted avg | 0.80 | 0.81 | 0.80 | 3588 |



## Most Efficient Model - Balanced, PCA Features, XGBoost Model - Training Results

```
Balanced train-test split: True
Model Algorithm: xgb
(array(['Cassava Bacterial Blight (CBB)',
       'Cassava Brown Streak Disease (CBSD)',
       'Cassava Green Mottle (CGM)', 'Cassava Mosaic Disease (CMD)',
       'Healthy'], dtype=object), array([737, 737, 737, 737, 737]))
X train shape: (2948, 130)
y train shape: (2948,)
X test shape: (737, 130)
y test shape: (737,)
100%|██████████| 5/5 [00:09<00:00,  1.90s/trial, best loss: -0.0]
Best Parameters: {'colsample_bytree': 0.5568981335918035, 'gamma': 3.8392819888268797, 'learning_rate': 0.35177173
25722587, 'max_depth': 14.0, 'n_estimators': 210.0, 'subsample': 0.9068277038201439}
Final Accuracy: 0.5210312075983717
Classification Report:
```

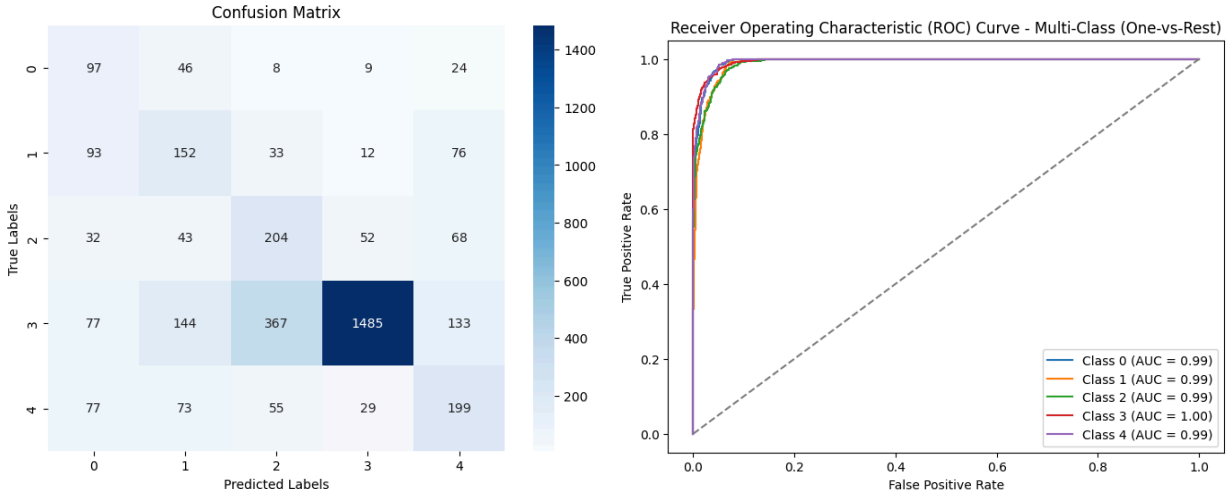
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Cassava Bacterial Blight (CBB) | 0.51 | 0.52 | 0.52 | 148 |
| Cassava Brown Streak Disease (CBSD) | 0.44 | 0.41 | 0.43 | 147 |
| Cassava Green Mottle (CGM) | 0.54 | 0.48 | 0.51 | 147 |
| Cassava Mosaic Disease (CMD) | 0.67 | 0.69 | 0.68 | 148 |
| Healthy | 0.45 | 0.50 | 0.47 | 147 |
| accuracy | | | 0.52 | 737 |
| macro avg | 0.52 | 0.52 | 0.52 | 737 |
| weighted avg | 0.52 | 0.52 | 0.52 | 737 |




## Most Efficient Model - Balanced, PCA Features, XGBoost Model - Evaluation Results

```
Classification Report:
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Cassava Bacterial Blight (CBB) | 0.26 | 0.53 | 0.35 | 184 |
| Cassava Brown Streak Disease (CBSD) | 0.33 | 0.42 | 0.37 | 366 |
| Cassava Green Mottle (CGM) | 0.31 | 0.51 | 0.38 | 399 |
| Cassava Mosaic Disease (CMD) | 0.94 | 0.67 | 0.78 | 2206 |
| Healthy | 0.40 | 0.46 | 0.43 | 433 |
| accuracy | | | 0.60 | 3588 |
| macro avg | 0.45 | 0.52 | 0.46 | 3588 |
| weighted avg | 0.70 | 0.60 | 0.63 | 3588 |




## Cross-class Performance

Cassava Mosaic Disease (CMD) consistently outperformed other classes, likely due to its prevalence in the unbalanced dataset. Interestingly, the performance remained similar before and after applying class balancing techniques. We hypothesize that this is attributable to the distinctive visual features of CMD, having a deformed leaf shape which make it more easily distinguishable from other classes. These unique characteristics may inherently enable the model to classify CMD with higher accuracy, independent of dataset balancing.

## 4. Generalizability

The model's generalizability was assessed by evaluating its performance on both the validation and test sets as seen in the graphs in the results of the classification section above. Notably, the model's performance on the test set closely matched that on the validation set, indicating effective generalization. For example, the overall accuracy of the training set for the unbalanced full feature model was 80% and the overall accuracy for the test set for the same model was 81%. The overall accuracy for the balanced reduced feature set that scored highest on efficiency was 52% and the overall accuracy for the same model with the test set was 60%. To mitigate overfitting, we implemented two key strategies. First, the hyperparameter search space was constrained to avoid selecting configurations that could lead to overly complex models or inefficiencies. Second, we employed Principal Component Analysis (PCA) to address the potential overfitting caused by the high dimensionality of the feature set, which exceeded the number of records. PCA reduced the dataset's dimensionality to capture at least 95% of the variance, ensuring the model focused on the most meaningful features. In future studies, to further improve generalizability, increasing the number of records for underrepresented classes would enhance the model's robustness and fairness. Instead of reducing the number of images to match the smallest class, artificial data augmentation techniques could be utilized, such as rotating, flipping, or applying other transformations to existing images in the smaller classes. This would allow for an increased number of images in each class, helping to maintain a larger overall dataset and potentially improving the balanced model's accuracy.

## 5. Efficiency vs. Accuracy

In evaluating the trade-offs between efficiency and accuracy, we tested various feature sets and models with the goal of balancing performance with computational costs. Models were trained using both full feature sets and PCA-reduced feature sets, with both balanced and unbalanced datasets. Training times and costs varied significantly across configurations. For example, using the full feature set with XGBoost on an unbalanced dataset took approximately 220.91 seconds per trial, resulting in a total training cost of $0.3620. In contrast, using PCA-reduced features with a balanced dataset led to a much faster training time of just 1.90 seconds per trial, reducing the total training cost to $0.0031. While both models' costs seem low, adding more configurations or using the models in a large scale on hundreds of datasets would escalate the cost implications. In that scenario, the PCA-reduced model's efficiency would offer significant cost savings.

Regarding model performance, the XGBoost model on the full feature set without class balancing achieved the highest accuracy of 0.8088, but it came at a substantial computational cost. In addition, the performance of this model was not consistent across the classes. The Mosaic disease class was overrepresented and performed significantly better than the other classes with an f1 score of 92% compared to the lowest performing class that had an f1 score of 45%. On the other hand, models trained on the PCA feature set with class balancing showed reduced accuracy, with XGBoost achieving a macro average accuracy of 0.5956, but these models were far more efficient, with reduced training costs. In addition, these more efficient models demonstrated a much more balanced performance across the classes.

The most accurate models, especially XGBoost with the full feature set on an unbalanced dataset, are more prone to overfitting due to the higher complexity of the feature set. In contrast, using PCA reduced overfitting risks but led to lower performance. Ultimately, a balance must be struck between computational cost and predictive accuracy, depending on the real-world application and the resources available.

These results highlight the real-world challenge of balancing model performance and computational efficiency, especially in applications like agriculture, where speed and accuracy are both crucial. The efficient model still offers practical benefits, enabling faster disease identification in cassava crops, but with a trade-off in predictive power.

## 6. Discussion and Limitations

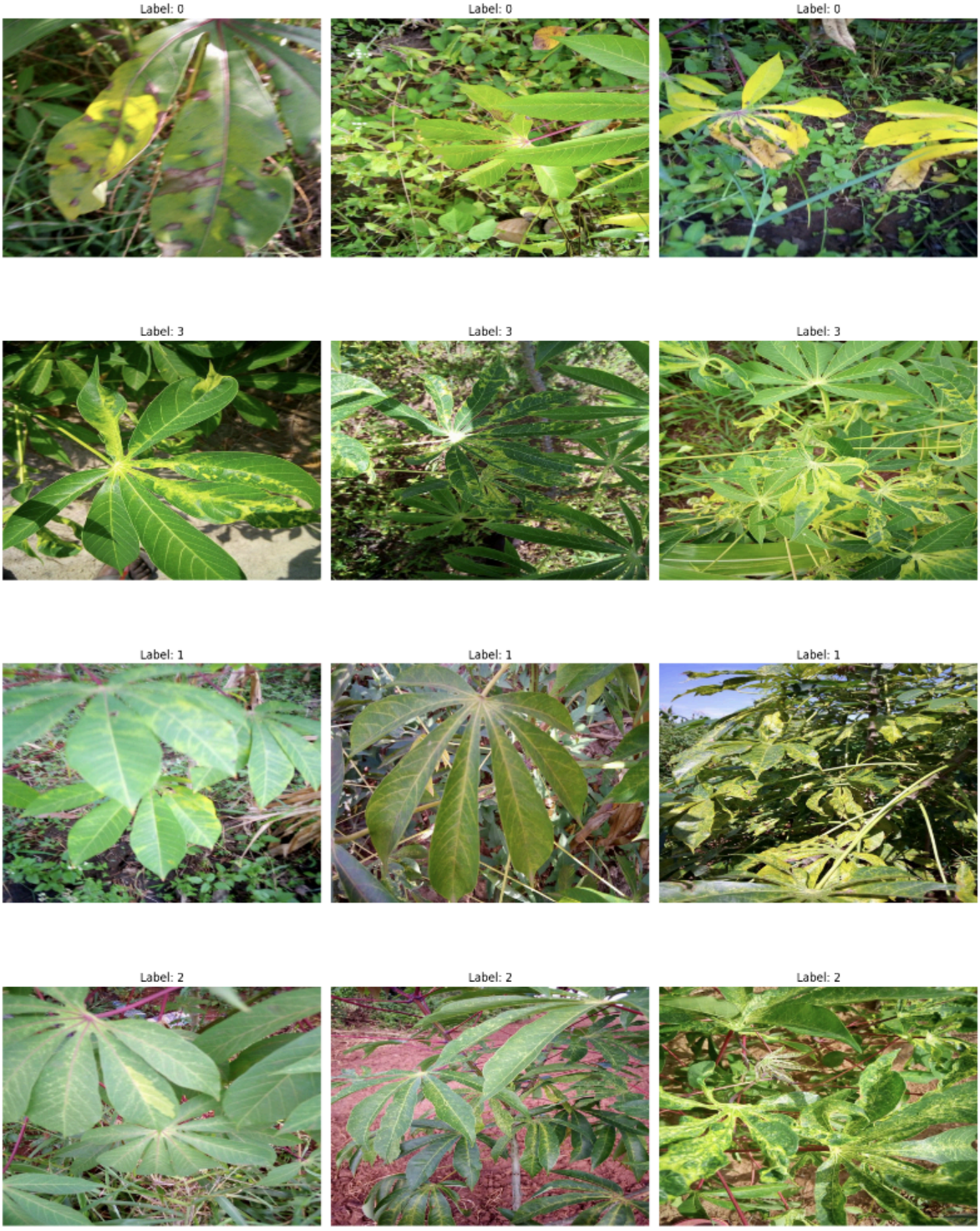
The study presented here underscores the significance of Cassava as a critical crop for global food security and the importance of developing reliable tools to address its susceptibility to diseases. In the initial stages of this study, we focused on extracting simple visual features, such as Histogram of Oriented Gradients (HOG), color distribution, and Fourier transform, to evaluate their effectiveness in cassava disease classification. These features alone yielded modest results, with accuracy stabilizing around 30% (detailed code is available on Github). However, the combination of these simple visual features with Dinov2 significantly improved the model's performance. This marked improvement highlighted the complementary strengths of traditional feature extraction techniques and deep learning-based approaches.

The decision to integrate both Dinov2 and simple visual features into the modeling pipeline was motivated by the need to leverage the strengths of each method. Simple visual features, while limited in their standalone effectiveness, are computationally efficient and can capture key visual patterns in the data, such as texture and color gradients. Dinov2, on the other hand, excels at identifying complex, high-level patterns through its transformer architecture but may overlook simpler, low-level features that are crucial for specific tasks. By combining these approaches, the models were able to exploit the broader spectrum of features present in the Cassava leaf images, leading to improved classification accuracy.

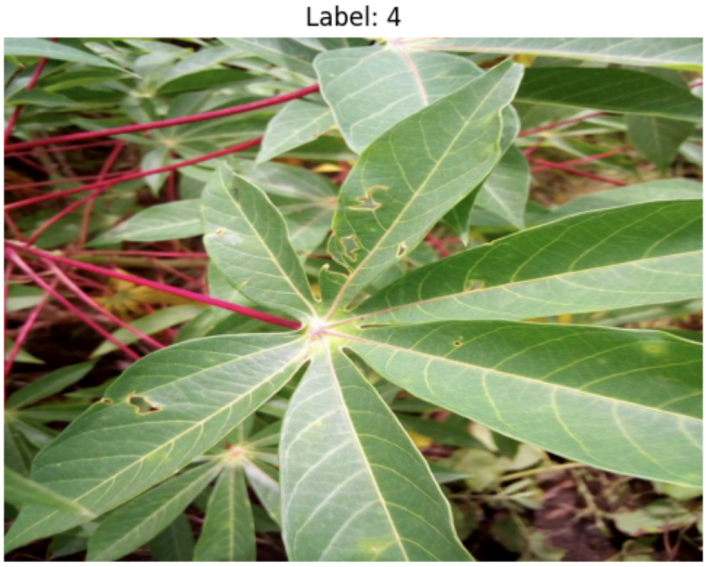
Based on these observations, the study focused on exploring variations of models that integrated Dinov2 with simple visual features. While the results demonstrated strong performance in identifying and classifying Cassava leaf diseases, several challenges and limitations emerged during the course of the study, providing valuable insights for future work.

One of the major challenges was the visual similarity between the colors of Cassava leaves, the symptoms of the diseases, and the backgrounds in the images. This similarity often made it difficult for the models to distinguish between the classes. Moreover, some diseases exhibited overlapping symptoms, complicating the classification process further. For example, all diseases had some form of yellowing of the leaves (as illustrated in the images below).

Additionally, during the preprocessing stage, there were instances where the classification of certain images in the dataset appeared questionable or unexpected. In the following group of images, the bottom right image labeled as class 2, looks much more like it should belong to class 3 due to the deformed shape of the leaves. In Green Mottle (class 2) the shape of the leaves is maintained.


Label: 0
Label: 0
Label: 0
Label: 3
Label: 3
Label: 3
Label: 1
Label: 1
Label: 1
Label: 2
Label: 2
Label: 2

The image below is classified as class 4 or healthy, it is not clear if the damage to the leaf is due to a disease or not.




Additionally, the dataset suffered from class imbalance, with certain disease classes being underrepresented. Sampling strategies were employed to address this imbalance, which helped mitigate this concern. However, this resulted in a reduced total number of images in the dataset and consequently may have reduced the model's accuracy. In the future, a different approach to sampling could be explored. Instead of reducing the number of images to match the smallest class, artificial data augmentation techniques could be utilized, such as rotating, flipping, or applying other transformations to existing images in the smaller classes. This would allow for an increased number of images in each class, helping to maintain a larger overall dataset and potentially improving the balanced model's accuracy.

A notable finding in our analysis was consistent overperformance of class 3, representing Cassava Mosaic Disease (CMD), across all models. In some cases, this was attributed to the greater number of images available for this class, which led to overprediction. However, even when the dataset was balanced, CMD still overperformed compared to other classes. This may be due to the distinctive symptoms of CMD, such as leaf distortion, curling, and deformation, which made the disease more easily recognizable. Tools like Dinov2 and traditional edge-detection methods likely excelled at identifying these unique visual features, highlighting how the model's architecture can influence performance.

This project also revealed several limitations that should be considered in the broader application of these models. First, the reliance on Dinov2, a deep learning model, reduced interpretability compared to simpler methods. While Dinov2 achieved high accuracy, the complexity of its

abstract feature extraction makes it difficult to explain why certain predictions were made. Farmers or other end users might struggle to trust and act on these highly abstract features.

Another limitation is the high cost associated with achieving high performance. The models performed well with comprehensive datasets and numerous features, but accuracy suffered with reduced datasets or simplified feature sets. This reliance on extensive data and computational resources could pose barriers to adoption in resource-constrained settings, where smaller datasets or less powerful hardware might be the norm.

Finally, the models relied on visual features, as both Dinov2 and traditional methods focused on image-based characteristics of the leaves. While effective for detecting diseases with prominent visual manifestations, these models are less capable of identifying diseases with subtle or non-visual symptoms, such as those involving internal tissue damage or physiological changes. This limitation highlights the need for complementary approaches, such as integrating special or biochemical data, to create more holistic diagnostic tools.

By leveraging a self supervised transformer model and combining it with simple visual feature extraction methods, this project offers an innovative approach to Cassava disease classification. This approach not only demonstrated the potential of combining traditional and advanced methods but also provided a foundation for building robust and efficient classification systems tailored to the specific challenges posed by Cassava disease detection.

## 7. Conclusion

This study underscores the importance of developing robust and efficient tools to address the challenges posed by Cassava plant diseases, which significantly impact global food security and agricultural economies. By combining simple visual features, such as HOG, Canny Edge detection, color histograms, and Fourier transforms, with the advanced self-supervised transformer model Dinov2, we created a hybrid framework capable of accurately classifying Cassava leaf diseases. While the best-performing model achieved an accuracy of 80.88% using an unbalanced dataset and the full feature set, the most efficient model balanced computational cost and performance, achieving an accuracy of 59.56% with PCA-reduced features and a balanced dataset. The high accuracy model took 220.91 seconds per trial and the high efficiency model took 1.9 seconds per trial.

Despite these achievements, limitations such as reduced interpretability, computational demands, and the focus on visual features highlight areas for improvement. Challenges such as class imbalance, overlapping symptoms, and the similarity in image colors also presented obstacles to classification. Future work should focus on integrating additional data types, such as spectral or biochemical data, and exploring advanced data augmentation techniques to address these limitations and improve the balanced model's accuracy while maintaining generalizability.

This project demonstrates the potential of combining traditional feature extraction methods with cutting-edge deep learning techniques, providing a foundation for scalable and effective Cassava disease classification systems. By continuing to refine these approaches, this work contributes to the broader effort of enhancing Cassava crop health, supporting farmers, and sustaining global food production.

## 8. References


"Wheat | USDA Foreign Agricultural Service." *Fas.usda.gov*, fas.usda.gov/data/production/commodity/0410000.

"Global Cassava Processing Market Report and Forecast 2022-2027." *Www.expertmarketresearch.com*, www.expertmarketresearch.com/reports/cassava-processing-market.

"Rice | USDA Foreign Agricultural Service." *Fas.usda.gov*, 2024, fas.usda.gov/data/production/commodity/0422110.

Alitubeera, Phoebe H., et al. "Outbreak of Cyanide Poisoning Caused by Consumption of Cassava Flour — Kasese District, Uganda, September 2017." *MMWR. Morbidity and Mortality Weekly Report*, vol. 68, no. 13, 5 Apr. 2019, pp. 308–311, www.cdc.gov/mmwr/volumes/68/wr/pdfs/mm6813a3-H.pdf, https://doi.org/10.15585/mmwr.mm6813a3.

Valentine Otang Ntui, et al. "Cassava Molecular Genetics and Genomics for Enhanced Resistance to Diseases and Pests." *Molecular Plant Pathology*, vol. 25, no. 1, 7 Nov. 2023, https://doi.org/10.1111/mpp.13402. Accessed 11 Feb. 2024.

## 9. Appendices

Link to the dataset
https://www.kaggle.com/datasets/mexwell/crop-diseases-classification?resource=download-directory

Link to the presentation
https://docs.google.com/presentation/d/1917Tv_SfIEpo9rpufzZnMI0zvcfncSS2Afbr9bZOSFk/edit

Link to Github repository
https://github.com/tamarbrandperez/w281_final_project_plants/tree/final