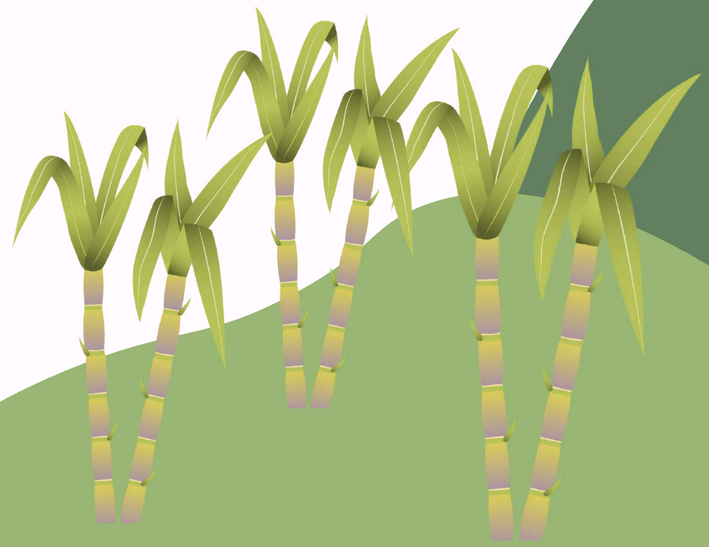
Disease detection in sugarcane leaves using ML

Done by-
Jayakrishna Reddy



Objective

In order to detect the disease in a sugarcane leaf, different datasets are acquired containing the images of the leaves in jpg/jpeg format. The datasets used for this research contain leaves of plants with the following three conditions: A Healthy leaf, Red Rot, Red Rust. Transfer learning is then used to train a CNN model on the available datasets for the feature extraction purpose i.e., only forward propagation takes place for updating the weights and parameters. The transfer learning models used are Densenet 201 and VGG16, both of which are trained on the images dataset. The output layers used with the CNN models are: SVM (Support Vector Machine), KNN (K-Nearest Neighbours) and Random Forest, all of which are classification models which finally output the disease predicted for a particular sugarcane leaf.

The proposed methodology involves categorizing sugarcane images into healthy and unhealthy/diseased groups using datasets of sugarcane leaves, featuring three distinct disease classes. Training is carried out on three datasets; each dataset being trained by different algorithms to find the best classification algorithm for that dataset.The paper also highlights future research areas aimed at further improving the system. The ultimate goal is to empower farmers with informed decision-making capabilities in the face of crop diseases and related challenges.

Methodology

1. Data Acquisition and Preparation:

i) Collect Images:

Labelled sugarcane leaf images with various diseases and healthy leaves available online for free are collected. The diversity in terms of lighting, camera angles, disease severity, and image backgrounds has been addressed in the preprocessing step.

ii) Preprocess Images:

Resizing, sharpening and augmenting of the images to increase data diversity and robustness has been employed. Techniques like flipping, rotating, and adding noise can be used.

Split Data: The dataset is split into training, validation, and testing sets for model training, hyperparameter tuning, and final evaluation, respectively. A common split is 70% training, 20% validation, and 10% testing.

2. Model Selection and Training:

i) Choose a Model: Deep learning models, particularly Convolutional Neural Networks (CNNs), are ideal for image classification. Pre-trained models like VGG16 or ResNet can be fine-tuned for sugarcane disease prediction. Transfer learning models such as Dense-Net 21 has also been employed due its number of deep layers. Various ML classification algorithms such as SVM, Random Forest and K-nearest Neighbors have been used and a comparison has been made in terms of accuracy and other metrics.

ii) Training Process:

Training of the model with the training data is carried out with various parameters such as Batch Size and number of epochs while monitoring performance metrics like accuracy, precision, recall, and F1-score on the validation set. Techniques like early stopping and hyperparameter tuning are used to optimize the model.

iii) Training Techniques:

Data augmentation, transfer learning, dropout layers, and regularization are employed improve model generalizability and to prevent overfitting

3. Comparison

A table is made to compare the various merics such as accuracy, F1 score, time complexity for the different methods and classification algorithms on different datasets.

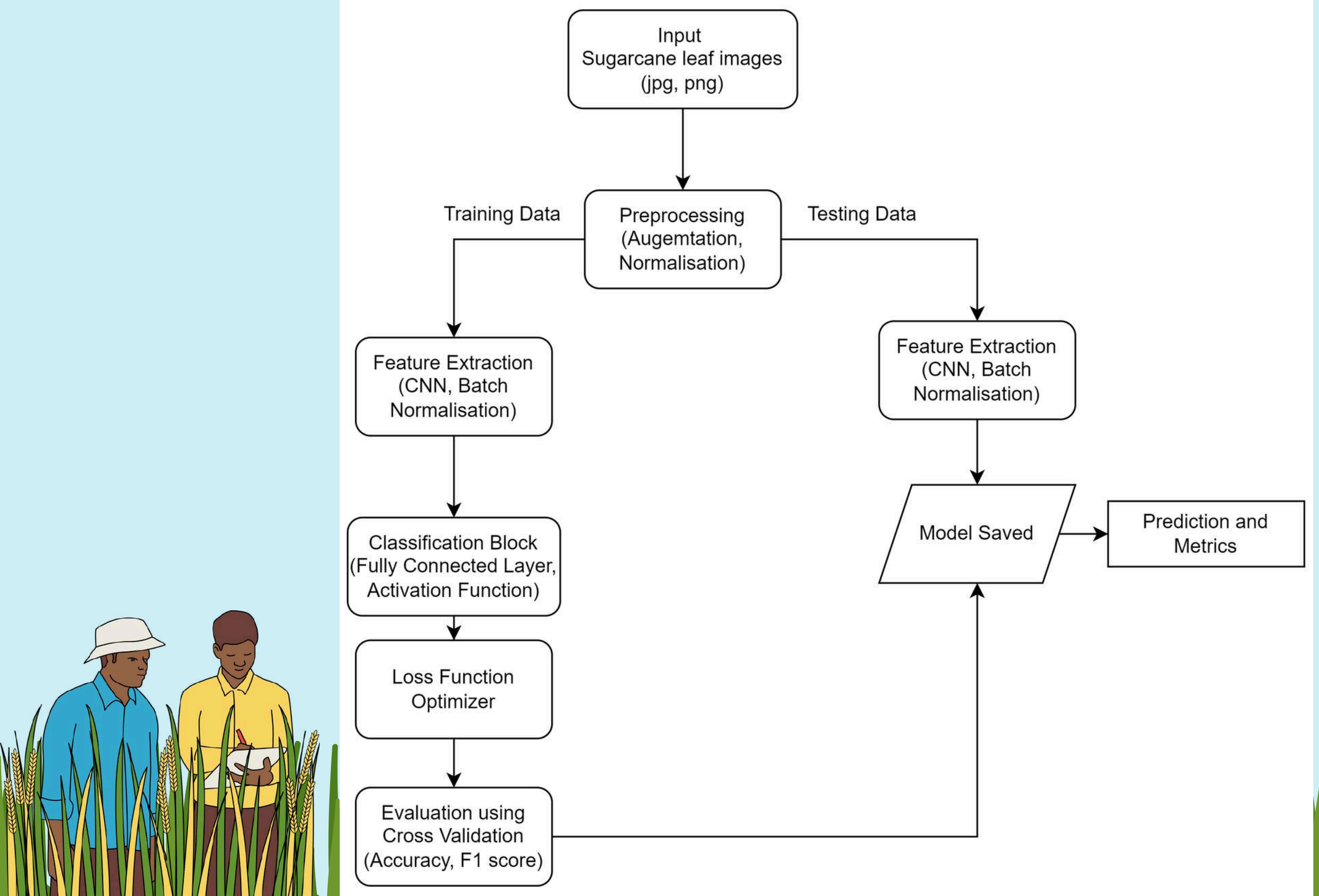
Healthy

RedRot

RedRust

Datasets

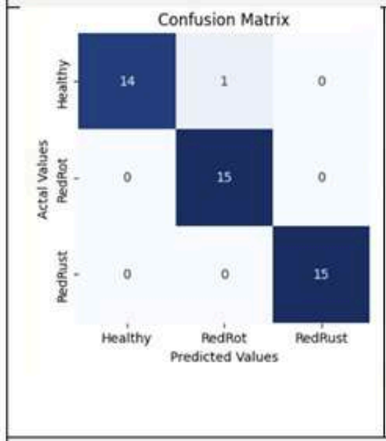
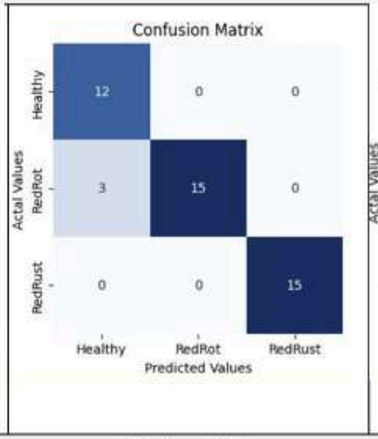
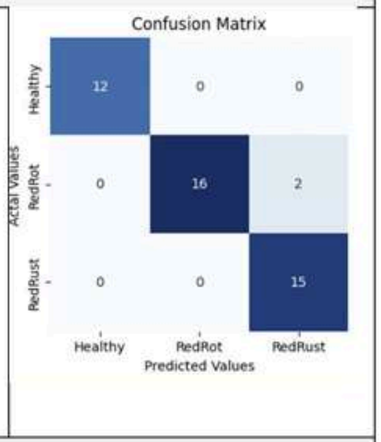
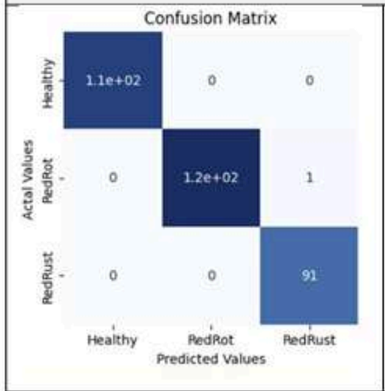
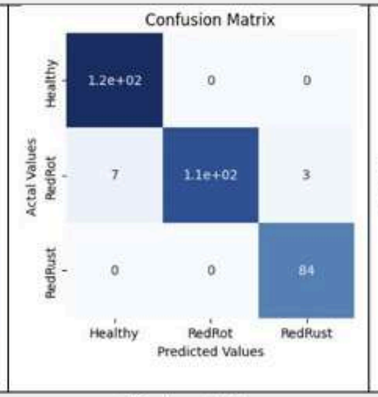
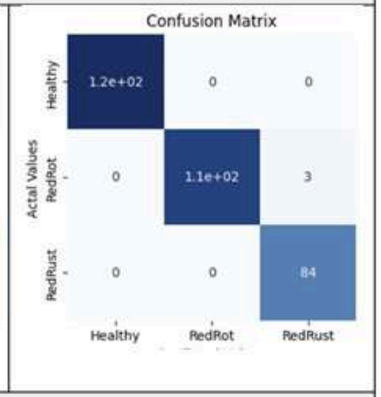
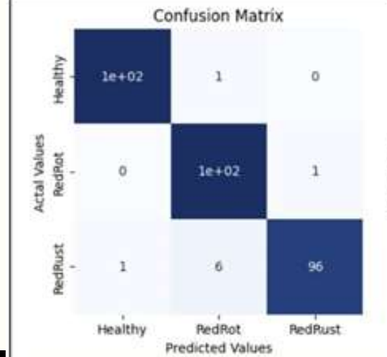
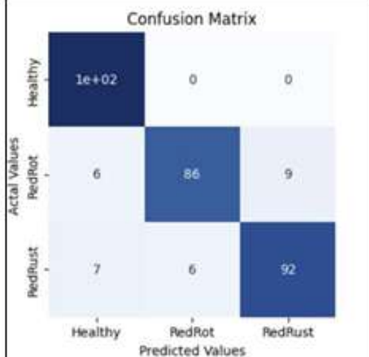
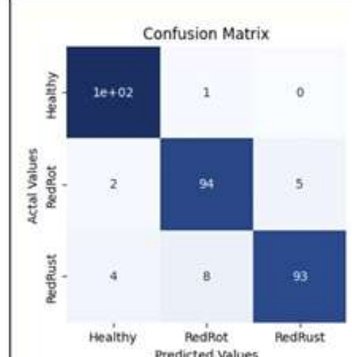
	Dataset-1	Dataset-2	Dataset-3
Healthy Leaves	75	556	522
Red Rot Leaves	73	610	518
Red Rust Leaves	75	445	514
Total Images	223	1621	1554

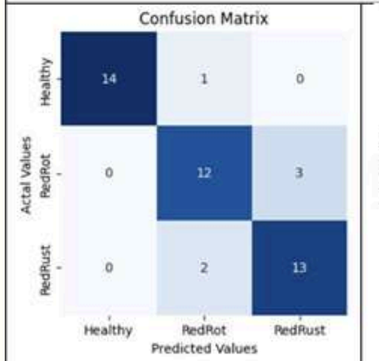
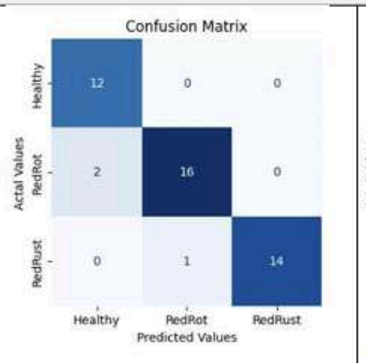
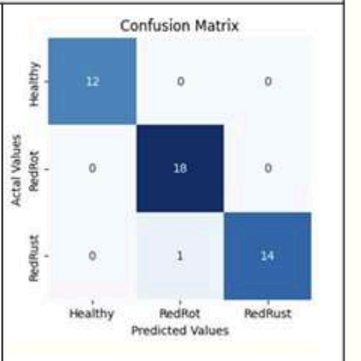
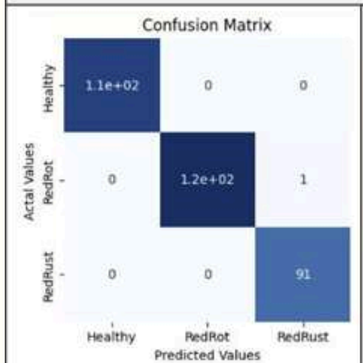
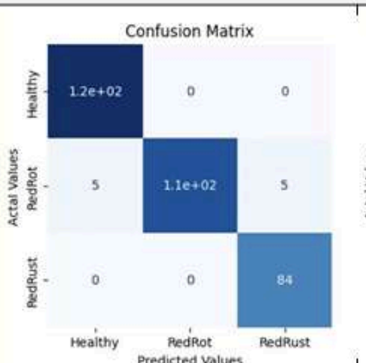
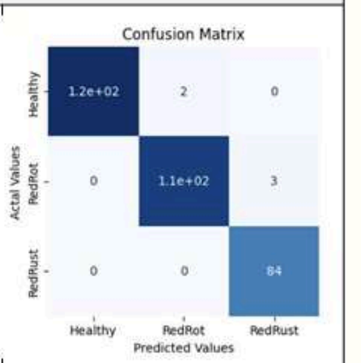
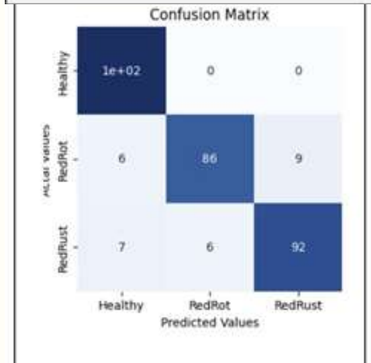
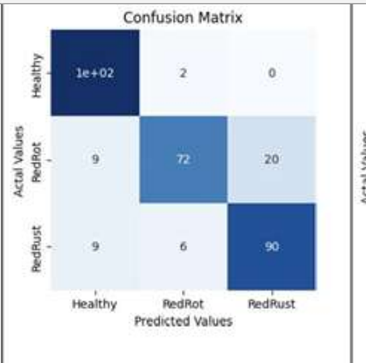
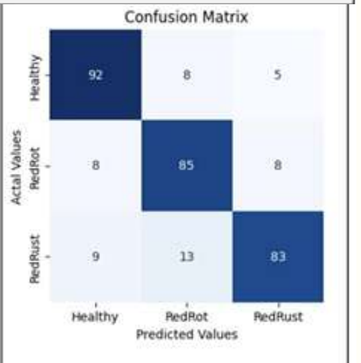

Input
Sugarcane leaf images
(jpg, png)
Preprocessing
(Augemtation,
Normalisation)
Training Data
Testing Data
Feature Extraction
(CNN, Batch
Normalisation)
Feature Extraction
(CNN, Batch
Normalisation)
Classification Block
(Fully Connected Layer,
Activation Function)
Model Saved
Prediction and
Metrics
Loss Function
Optimizer
Evaluation using
Cross Validation
(Accuracy, F1 score)

Results

Observations

Datasets		DenseNet201			VGG-16		
		SVM	KNN	RF	SVM	KNN	RF
1	Accuracy (%)	97.65	93.33	96.00	86.67	93.30	97.77
	Precision	0.98	0.94	0.96	0.87	0.94	0.98
	F1-score	0.98	0.93	0.96	0.87	0.93	0.98
2	Accuracy (%)	100	96.92	99.07	99.69	97.53	98.46
	Precision	1.00	0.97	0.99	1.00	0.97	0.98
	F1-score	1.00	0.97	0.99	1.00	0.97	0.98
3	Accuracy (%)	93.56	90.99	93.56	82.31	85.2	83.6
	Precision	0.97	0.91	0.94	0.84	0.86	0.84
	F1-score	0.97	0.91	0.94	0.82	0.85	0.84

DenseNet201		
SVM	**KNN**	**Random Forest**
Dataset-1		
		
Dataset-2		
		
Dataset-3		
		

VGG-16		
SVM	**KNN**	**Random Forest**
Dataset-1		
		
Dataset-2		
		
Dataset-3		
		

Pre-processing

Pre-processing •••

Image pre-processing prepares raw images for analysis by cleaning them up, enhancing features, and standardizing them for better performance in downstream tasks.

Image Sharpening

Enhances image clarity:
Sharpening boosts edge contrast, making details "pop" and improving visual quality.

I**mproves feature analysis:** Sharper edges help algorithms better recognize objects and extract details in downstream tasks.

Image Filtering

Cleaning Up the Image:
Image filtering in pre-processing acts like a digital vacuum, removing unwanted noise and artifacts like dust specks or camera blur. This leaves behind a clearer, more consistent image for further analysis.

Tailored Enhancements:
Different filters target specific issues. Some smoothen textures, others sharpen edges, and some correct uneven lighting.

Image Resizing

Standardization for Algorithms:
Resizing ensures all images have the same dimensions, making them compatible with machine learning algorithms that require specific input sizes. It's like fitting everyone into the same uniform for easier processing.

E**fficiency Boost:**
Smaller images take less time and memory to process, especially for computationally heavy tasks like training deep learning models. Think of it as making the data lighter to carry for the algorithm.

Image Augmentation

Data Expansion:
Augmentation artificially creates new, diverse versions of existing images (flips, rotations, color shifts). This "expands" your training data, making models more robust to real-world variations.

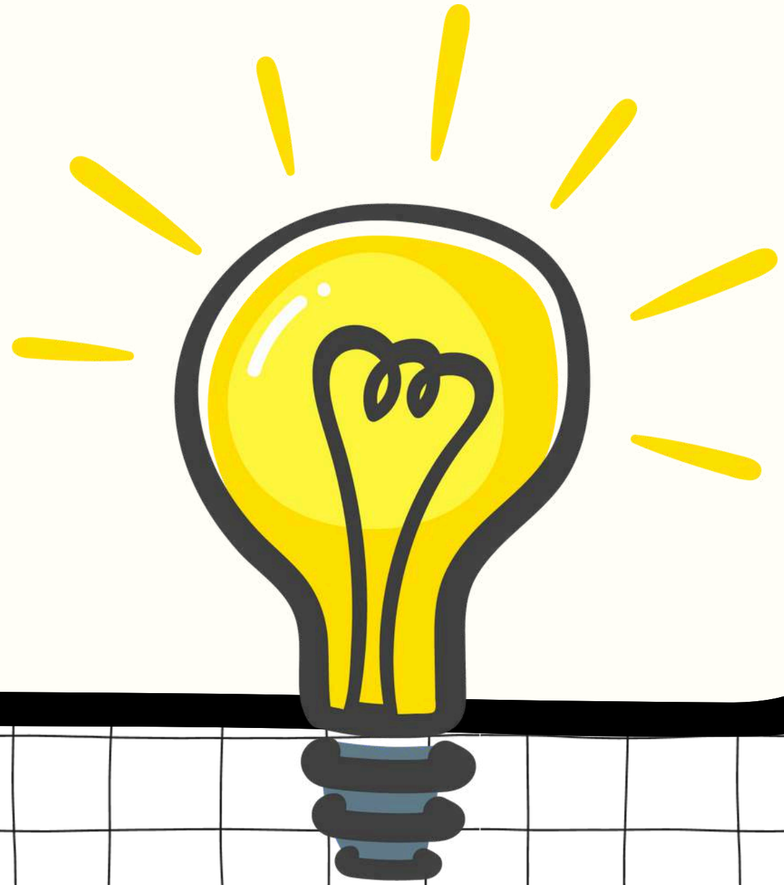
Generalization Boost: By seeing images in different contexts, models learn to identify the core features, not just specific details. This improves their ability to generalize to unseen data.

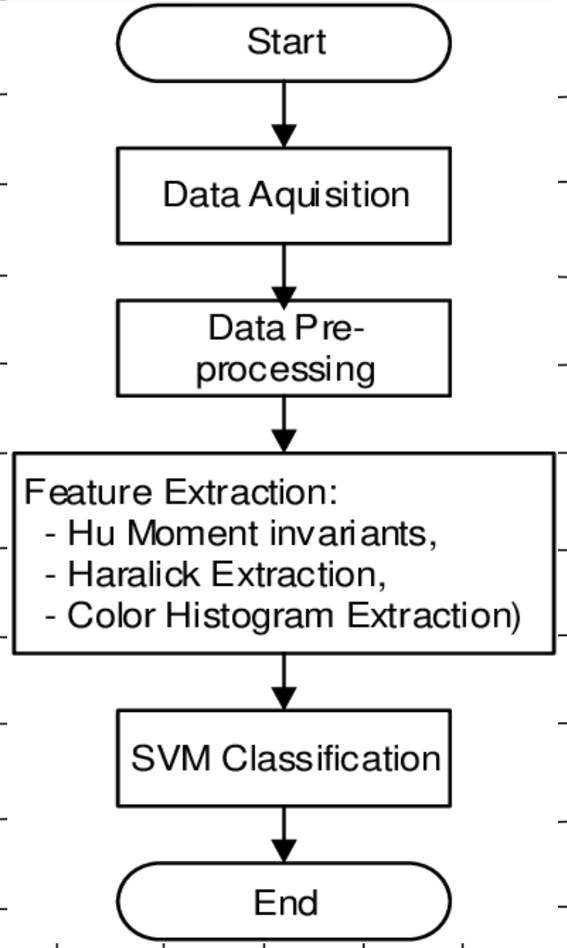
Unlike resizing or filtering, augmentation happens only to the training data, not the entire dataset. It's a separate step aimed at improving model performance.

Algorithms

SVM

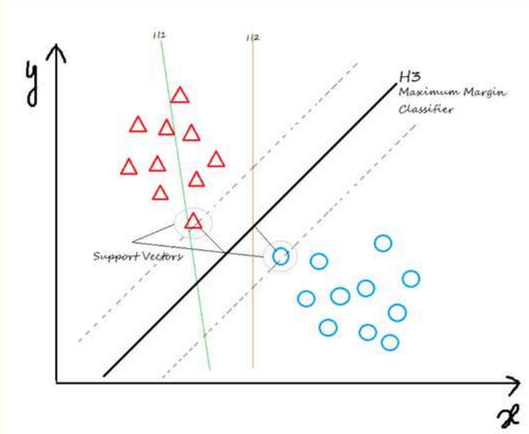
Support vector machines





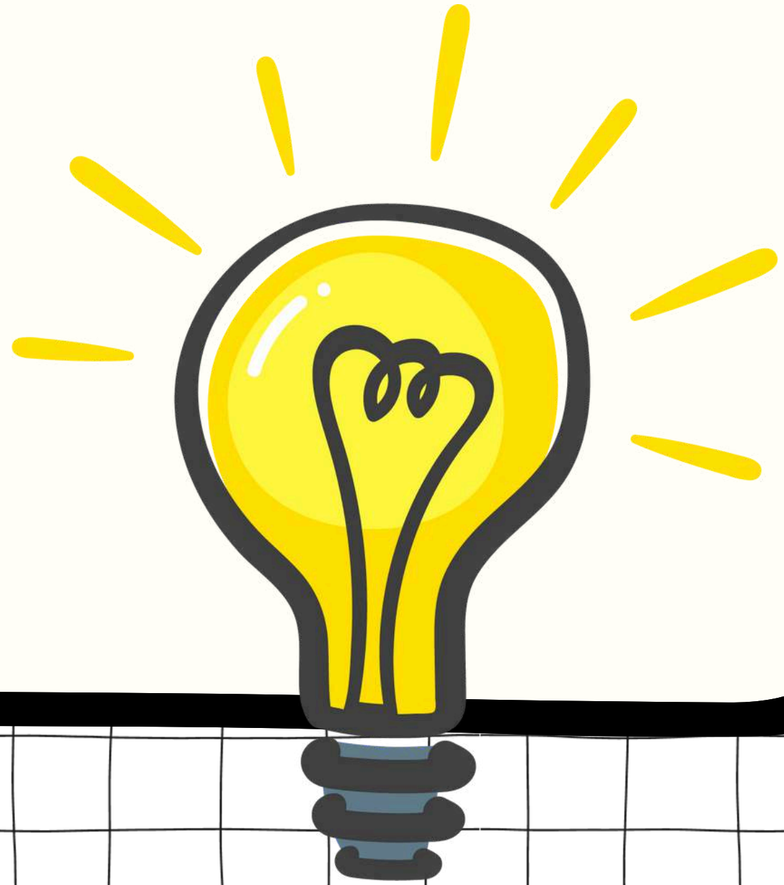
The steps for the algorithm are as follows:

i) Load the imported libraries
ii) Import dataset data point to be classified.
iii) Divide dataset into testset and trainset.
iv) Fitting the SVM Classifier.
v)Code snipped created to DenseNet201 model with specific input and output configurations, suitable for extracting features from images for further processing in a custom CNN architecture.
vi) for each image in the testing set we need to find the index of the label with corresponding largest predicted probability



KNN

K-Nearest Neighbours Algorithm



The steps for the algorithm are as follows:

i) Choose a K value (Eg: 5)
ii) Calculate the distance of the K (5) nearest data points (neighbours) from the data point to be classified. There are two methods to calculate ths distance between the data points: Euclidean Distance and Manhattan Distance.

Euclidean Distance = ((x2-x1)^2 + (y2-y1)^2)^1/2
Manhattan Distance = (a+b)

iii) Once the K nearest data points have been found, the class to which
the data points belong to is observed. The class containing the majority
of the datapoints is then chosen as the label class for the unclassified
data point.

Finding the value of K

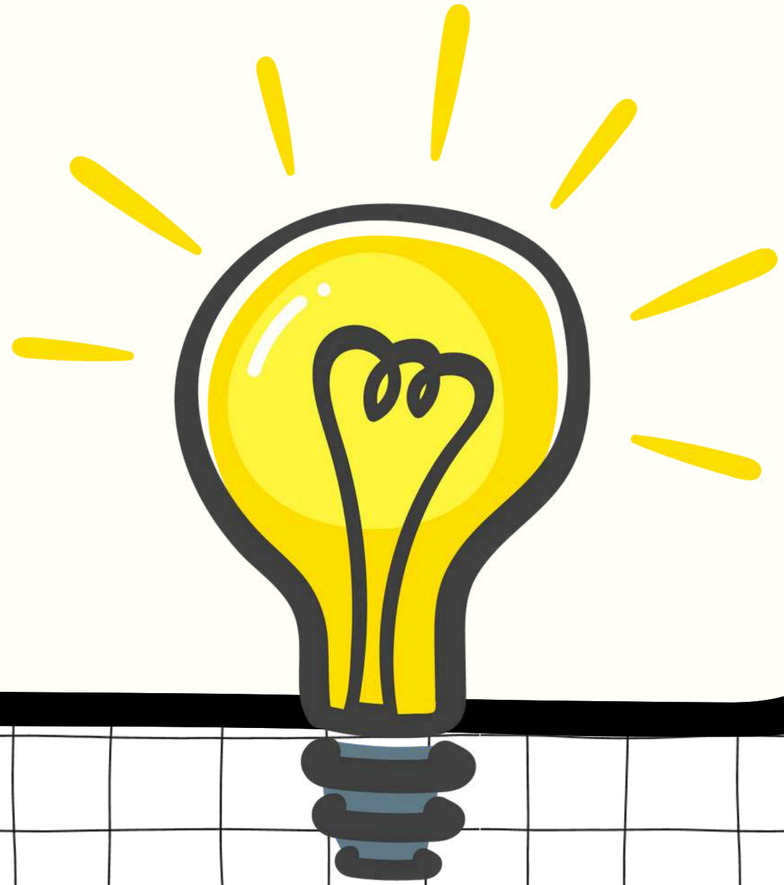
The suitable value of K found out by iterating over a range of K-values :
(Eg: 1-40) and creating a model with the K value for each iteration.

The model is then trained on the training data and the mean error is calculated at the end using the testing data by summing up the incorrectly classified datapoints.

The error for each iteration is stored in an array and is plotted after the end of the loop. The K value resulting in the lowest error is then chosen for our final model.

RF

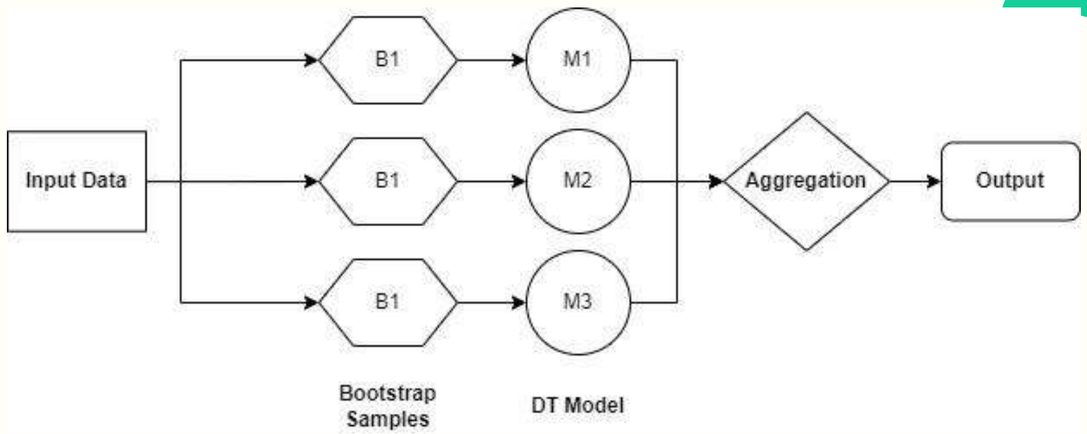
Random Forest Algorithm



The Random Forest, a widely adopted machine learning algorithm, was developed to merge the outputs of multiple decision trees into a unified result. Its popularity stems from its user-friendly nature and versatility, capable of effectively addressing both classification and regression challenges.

A Random Forest is like a group decision-making team in machine learning. It combines the opinions of many “trees” (individual models) to make better predictions, creating a more robust and accurate overall model.

- One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification. It performs better for classification and regression tasks.
- Random Forest Algorithm is one of the ensemble technique elgorithms which comes under Bagging (Bootstrap Aggregation), which makes use of multiple models to give an aggregate output from the output of each model. In this case, the model used is Decision Tree.



Transfer Learning

Transfer learning is a technique in deep learning where a model trained on one task is adapted for a second, related task. Instead of training a neural network from scratch for a new task, transfer learning leverages the knowledge gained from solving a different but related problem. This approach is particularly useful when the amount of labeled data for the target task is limited, as it allows the model to benefit from the information learned in the source task.

There are typically two main steps in transfer learning:

- Pre-training on a source task: A neural network is trained on a large dataset and a related task. This initial training helps the model learn general features and patterns from the data. The model can be pre-trained on a diverse and extensive dataset, often using a well-established architecture like a convolutional neural network (CNN) for image-related tasks or a recurrent neural network (RNN) for sequence-related tasks.
- Fine-tuning on a target task: After pre-training, the knowledge gained by the model is transferred to a new task by fine-tuning on a smaller dataset specific to the target task. The parameters learned during pre-training are adjusted or fine-tuned to adapt the model to the nuances of the new task. This step helps the model specialize and improve its performance on the target task with less labeled data.

Transfer learning has proven to be effective in various domains, including computer vision, natural language processing, and speech recognition. It enables the development of accurate models even when the amount of task-specific labeled data is limited, making it a valuable technique in practical applications where data availability is a challenge.

For the purpose of our study, we use tranfer learning with the help of two CNN models: **DenseNet201** and **VGG16**.

These tranfer learning models are used for the feature extraction process from the input images i.e, the neural network is not trained using back-propagation and instead, the features extracted through forward propgation are inputs to the final output (classification) layer: SVM, KNN, Random Forest

Conclusion

1. **Effective Utilization of Transfer Learning**: The research effectively utilized transfer learning with DenseNet201 and VGG16 models to address the task of disease prediction in sugarcane leaves. By fine-tuning these models, the researchers adapted the learned parameters to better suit the specific nuances of the task, demonstrating the versatility and effectiveness of transfer learning in agricultural applications.
2. **Feature Extraction and Model Adaptation**: The extracted features from each image in the dataset were crucial for accurately predicting diseases in sugarcane leaves. Fine-tuning the pre-trained models allowed for the adaptation of learned features to the specific characteristics of the sugarcane leaf images, showcasing the importance of feature extraction and model adaptation in machine learning tasks.
3. **Algorithm Selection and Performance Variation**: The study compared SVM, KNN, and Random Forest algorithms for classification, revealing variations in classification reports and accuracy scores. Additionally, differences in training times were observed, reflecting the distinct workings of each algorithm. This underscores the importance of algorithm selection based on the specific requirements and constraints of the application.
4. **Dataset Quality and Algorithm Performance**: The research highlighted the critical role of dataset quality in determining the most suitable machine learning model for a given application. The performance of SVM, KNN, and Random Forest varied across datasets, with no single algorithm emerging as consistently superior. This emphasizes the need for understanding the characteristics of the dataset and considering the nuanced behavior of each algorithm in making informed choices for classification tasks.
5. **Implications for Agricultural Applications**: The findings have significant implications for agricultural applications, where accurate disease prediction can lead to improved crop management practices, increased yield, and reduced environmental impact. By leveraging transfer learning and selecting appropriate classification algorithms, researchers and practitioners can develop effective tools for disease detection and management in agriculture.

Overall, the conclusion underscores the importance of thoughtful model selection, feature extraction, and dataset quality in achieving accurate disease prediction in sugarcane leaves, with broader implications for machine learning applications in agriculture and beyond.