A IDP PROJECT REPORT

on

“Rotten Food Classification Using

Machine Learning”

Submitted

By

221FA04281

A.Chinnapa

221FA04440

K.Yoshitha

221FA04593

K.Naga Lakshmi

221FA04744

K.Pradeep Chandra

Under the guidance of

Souvrav Mondal

Associate Professor

SCHOOL OF COMPUTING & INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH Deemed to be UNIVERSITY

Vadlamudi, Guntur.

ANDHRA PRADESH, INDIA, PIN-522213.

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

CERTIFICATE

This is to certify that the IDP Project entitled "Rotten Food Classification Using Machine Learning" that is being submitted by 221FA04283 (A.Chinnapa), 221FA04440(K.Yoshitha), 221FA04593(K.Naga Lakshmi) and 221FA04744(K.Pradeep Chandra) for partial fulfilment of Field Project is a bonafide work carried out under the supervision of Sourav Mondal., Assistant Professor, Department of CSE.

Mr. Sourav Mondal & Signature

Designation

Dr. S. V. Phani Kumar

HOD,CSE

VIGNAN'S
FOUNDATION FOR SCIENCE, TECHNOLOGY & RESEARCH
(Deemed to be University) - Estd. u/s 3 of UGC Act 1956

DECLARATION

We hereby declare that the IDP Project entitled "Rotten Food Classification Using Machine Learning" that is being submitted by 221FA04281(A.Chinnapa), 221FA04440(K.Yoshitha), 221FA04593(K.Naga Lakshmi) and 221FA04744(K.Pradeep Chandra) in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Mr. Sourav Mondal, Assistant Professor, Department of CSE.

By

221FA04281(A.Chinnapa),
221FA04440(K.Yoshitha),
221FA04593(K.Naga Lakshmi),
221FA04744(K. Pradeep Chandra)

ABSTRACT

Fresh and spoiled fruit classification is an important problem in agriculture that influences food safety and efficiency. This paper introduces an automated fruit classification system based on machine learning methods. The system classifies fresh fruits from spoiled fruits based on texture and color-based feature extraction techniques such as Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG). Classification is done with ensemble learning models such as XGBoost, Gradient Boosting, and CatBoost, together with meta-learning methods utilizing Artificial Neural Networks (ANN) and Support Vector Machines (SVM). The proposed method was tested on a dataset of 12,000 images and achieved a classification accuracy of 96.8%. The paper demonstrates the superiority of combining feature extraction with ensemble learning and deep learning techniques to improve the accuracy of fruit classification.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1 INTRODUCTION

1. INTRODUCTION

1.1 Background and Significance of Rotten Food Classification

Rotten food classification is an essential application of Machine Learning (ML) that helps in identifying and distinguishing spoiled food from fresh produce. With the increasing global demand for food safety, automated food quality assessment has become a crucial aspect of food supply chains, retail industries, and household applications. Traditional methods of food quality inspection rely on human sensory evaluation, which is subjective, time-consuming, and prone to errors. Recent advancements in ML and Deep Learning (DL) have significantly improved food classification accuracy by leveraging image processing, pattern recognition, and spectral analysis techniques.

The classification of rotten food plays a vital role in reducing food waste, ensuring consumer safety, and maintaining food quality standards. ML-based systems can analyze images and detect spoilage by identifying texture changes, color variations, mold growth, and other degradation signs. Computer vision techniques powered by Convolutional Neural Networks (CNNs) and Transfer Learning models have revolutionized the field, offering real-time, automated solutions for food quality assessment. However, challenges such as lighting conditions, different food textures, and variations in spoilage patterns make accurate classification a complex task. Despite these challenges, continuous advancements in ML and the availability of large annotated datasets have contributed to the increasing accuracy and efficiency of food classification models.

1.2 Overview of Machine Learning in Rotten Food Classification

Machine Learning has transformed rotten food classification by enabling automated detection of spoilage in fruits, vegetables, dairy products, and other perishable goods. Traditional food classification relied on rule-based systems and threshold-based image processing techniques, which struggled with varying lighting conditions and food appearances. ML-based classification utilizes supervised and unsupervised learning techniques to generalize across different food types and spoilage levels.

Feature extraction methods such as Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), and edge detection help in identifying visual patterns indicative of spoilage. CNNs have become the dominant approach in food classification, as they effectively extract spatial features and detect color, texture, and shape variations. Transfer Learning with pre-trained models such as ResNet, VGG, and MobileNet has improved classification accuracy by leveraging knowledge from large-scale image datasets. Additionally, Generative Adversarial Networks (GANs) and data augmentation techniques are used to enhance dataset diversity, reducing overfitting and improving generalization. Despite these advancements, challenges like occlusions, lighting variations, and mixed spoilage conditions necessitate robust pre-processing and domain-specific fine-tuning for optimal performance.

1.3 Research Objectives and Scope

The primary objective of this research is to develop an efficient ML-based rotten food classification system that improves the accuracy of detecting spoiled food items. The scope of this study includes dataset preprocessing, feature extraction, and model training using ML and DL techniques. This research aims to address challenges such as varying spoilage levels, handling diverse food types, and improving classification performance under different environmental conditions.

This study will compare various classification models, including traditional machine learning classifiers (Support Vector Machine, Random Forest, K-Nearest Neighbors) and deep learning architectures (CNNs, Transformers, and Hybrid models). The focus will be on optimizing

preprocessing techniques, including color correction, contrast enhancement, and background removal, to improve classification accuracy. The dataset will include a wide variety of perishable food items to ensure robustness. The evaluation metrics will include accuracy, precision, recall, F1-score, and inference time to determine the effectiveness of different models. Furthermore, real-world applications such as automated food inspection in supermarkets, smart refrigerators, and food waste management systems will be explored. Cost-effectiveness and computational efficiency will also be analyzed to assess the feasibility of deploying the system on edge devices and cloud platforms.

1.4 Current Challenges in Rotten Food Classification

Rotten food classification presents multiple challenges that impact accuracy and usability. One of the primary challenges is the variability in spoilage patterns, as different food items decay at different rates and exhibit unique visual changes. Lighting conditions significantly impact classification performance, as variations in illumination can affect color perception and feature extraction.

Another major challenge is distinguishing between natural ripening and actual spoilage, as some fruits and vegetables change color during the ripening process without being spoiled. Occlusions and mixed food conditions in images also make classification difficult, requiring robust segmentation techniques. Real-time classification demands lightweight and efficient models that can process food images quickly while maintaining high accuracy.

Data scarcity is another challenge, as creating large-scale annotated datasets for various food items at different spoilage levels is time-consuming. Privacy concerns may also arise when deploying AI-based food monitoring systems in smart homes and commercial settings. Additionally, generalization across different food categories requires extensive training and fine-tuning, as ML models may struggle to differentiate between similar spoilage patterns in different foods. Addressing these challenges requires innovative deep learning techniques, improved dataset collection, and optimized model architectures to develop scalable and high-performing food classification solutions.

1.5 Applications of ML to Rotten Food Classification

Machine Learning-powered rotten food classification has numerous applications across various industries, significantly improving efficiency, food safety, and waste reduction. Some key applications include:

- Food Safety and Quality Control: ML-based classification ensures that only fresh and safe food items are sold in supermarkets, restaurants, and food processing units.
- Smart Refrigeration Systems: AI-powered refrigerators can detect spoiled food and notify users, reducing food waste and improving household management.
- Automated Food Inspection: Supermarkets and food suppliers can integrate ML-based systems to automate the quality assessment of perishable goods.
- Food Waste Management: Identifying spoiled food early helps reduce food waste in retail, households, and supply chains.
- Agriculture and Post-Harvest Management: Farmers and distributors can use ML to monitor crop freshness and optimize storage conditions.
- E-commerce and Online Grocery Stores: Automated classification ensures accurate labeling of fresh and spoiled items in online marketplaces.

Important Uses of Machine Learning in Rotten Food Classification

1. Enhanced Accuracy: ML improves classification accuracy, even in challenging environments with poor lighting and background clutter.

2. Adaptability to Different Food Types: ML models can learn and generalize across various fruits, vegetables, dairy, and meat products.
3. Automated Detection of Spoilage: Eliminates the need for manual inspection, saving time and reducing human error.
4. Noise and Distortion Handling: ML-powered classification can work with images captured under diverse environmental conditions.
5. Real-Time Processing: Enables real-time food quality assessment in smart kitchens and automated retail checkout systems.
6. Integration with IoT Devices: ML-based food classification can be embedded in IoT-powered food monitoring systems.
7. Better Handling of Complex Textures: ML models can analyze subtle texture variations to detect early signs of spoilage.
8. Scalability for Large-Scale Applications: ML allows processing of vast amounts of food images efficiently for industrial applications.
9. Cost-Effective Solutions: Reduces operational costs by automating food inspection processes.
10. Improved Food Safety Compliance: Ensures adherence to food safety regulations by identifying unsafe food items.

Benefits of ML in Rotten Food Classification

- Higher Accuracy in Spoilage Detection
- Reduction of Food Waste through Early Identification
- Automated and Efficient Food Quality Control
- Real-Time Monitoring for Smart Kitchens and Food Suppliers
- Improved Consumer Safety by Preventing Spoiled Food Consumption
- Scalability for Commercial and Industrial Applications

Addressing the challenges in rotten food classification through advanced ML techniques will ensure more reliable, efficient, and scalable solutions for food safety and waste reduction.

CHAPTER-2

2. LITERATURE SURVEY

2.1 Literature review

Sangam et al. [1] proposed a fruit classification model that classifies fruit images into two classes: fresh and rotten. The dataset contains 10,000 images of apples, bananas, and oranges, both fresh and rotten, for training, validation, and testing. The model is constructed with a Convolutional Neural Network (CNN) architecture and achieves an accuracy of 94.5 Amin et al. [2] suggest a cost-effective approach for classifying fruit freshness based on an eightlayer AlexNet model with five convolutional layers and three fully connected layers. The dataset is 15,000 fruit images and is preprocessed with color normalization, image scaling, data augmentation, and labeling. Transfer learning is applied by the authors and the CNN model is fine-tuned before employing a Softmax classifier for classification. The method is tested on three publicly available datasets with an accuracy of 97.1 Abayomi-Alli et al. [3] introduce a dataset for use in fruit classification experiments, which is organized to allow research on the assessment of fruit quality. The dataset has 25,000 images and is well-labeled, so it can be used to test machine learning and deep learning methods. Although the dataset allows one to build sophisticated fruit classification models, the paper does not test or apply any model, so its real-world usability is not proven. There are no accuracy figures given. Kazi et al. [4] study the classification capability of convolutional neural networks (CNNs) for determining fruit freshness, applying both the basic and improved CNN architectures. The models are assessed on a collection of 12,000 images of six different fruit varieties. Findings suggest that fruit freshness can be successfully classified by models based on CNN, with a top model with an accuracy rate of 96.8 Kang et al. [5] propose a two-component classification model: one binary classifier separating fresh from spoiled fruits and one multi-class classifier determining the type of fruit. There are 18,000 images across different fruit types in the dataset to provide multiple diversity training samples. Transfer learning is utilized in the research to enhance classification accuracy, especially where training datasets are small. The method reaches an overall accuracy level of 97.5 Dhiman et al. [6] provide an overview of earlier work on fruit quality recognition, contrasting different machine learning and deep learning methods. The paper assesses advancements in research on fruit classification but does not present a novel dataset or technique. Experimental validation is lacking, restricting conclusions drawn on classification accuracy. The paper presents accuracies from earlier research between 85% and 98%. Mohapatra et al. [7] propose a CNN model that can identify fruit type and evaluate freshness. The dataset contains 22,000 images of various fruit types under different conditions, making the system robust.

Transfer learning is used in the research to maximize model performance with an accuracy of 96.9 Aherwadi et al. [8] analyze fruit identification, maturity classification, and quality ranking via a mixed process of machine learning, deep learning, and image processing approaches. The images account for 14,000 within the database that helps distinguish most efficient forms of classification to employ. It fails to make overt accuracy determinations or reflect upon limitations brought forth by similarities among fruit looks. Thin, Nguyen Vuong, et al. [9] explore fruit classification with K-Nearest Neighbors (KNN), Random Forest, and Support Vector Machine (SVM). The dataset has 9,500 images taken under controlled conditions. The research proves that SVM performs better than other models in the classification of fruits with an accuracy of 95.4 Sharma et al. [10] proposed a deep learning model to classify food freshness, with emphasis on identifying rotten fruits and vegetables. The database is comprised of 20,000 images across a range of food items like apples, bananas, tomatoes, and leafy greens in both fresh and rotting states. The authors utilized a ResNet50-based CNN model with a 95.7 Wang et al. [12] proposed a multi-class food spoilage detection model based on YOLOv5 and EfficientNet, trained on a 18,000-image dataset. The dataset consists of meat, dairy, fruits, and vegetables in various levels of decomposition. The research adopts real-time detection methods for mold growth, color deterioration, and texture modification. The suggested model has an accuracy of 96.3 Kumar et al. [12] introduced a CNN-LSTM hybrid deep learning model for the detection and prediction of food spoilage with respect to time. The data consists of 15,500 images taken under various environmental conditions to imitate natural food spoilage. The paper proposes a time-series prediction system where spoilage development can be monitored dynamically. The suggested model is 94.8

CHAPTER-3

3. PROPOSED SYSTEM

The proposed system aims to develop an advanced Rotten Food Classification Model using machine learning techniques to improve the accuracy of detecting spoiled food across various categories. Unlike traditional food inspection methods that rely on manual examination or rule-based approaches, this system leverages ensemble learning techniques to enhance classification performance. The model integrates deep learning-based feature extraction and meta-learning techniques to ensure robust classification of fresh and rotten food items.

The system follows a structured pipeline, beginning with data acquisition, where a diverse dataset containing images of fresh and spoiled food across multiple categories, such as fruits, vegetables, dairy, and meat, is collected. The dataset undergoes pre-processing steps, including image resizing, noise reduction, histogram equalization, and color normalization, to improve input quality before feature extraction. Feature extraction is performed using Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG), ensuring that essential texture patterns, edge orientations, and structural features are captured for accurate classification. These features help differentiate between fresh and spoiled food by detecting subtle changes in texture, mold patterns, and color variations.

For model training, the system employs a hybrid ensemble learning approach, where base models such as XGBoost, Gradient Boost, and CatBoost generate initial predictions, and meta-learners like Support Vector Machines (SVM) and Artificial Neural Networks (ANN) refine classification accuracy. This approach enhances classification robustness by reducing bias and variance, ensuring higher reliability in detecting food spoilage across diverse conditions.

Once the classification is complete, a post-processing module refines predictions using confidence thresholding and heuristic-based rules, minimizing false positives. Additionally, the system supports real-time classification, making it suitable for applications such as smart refrigerators, supermarkets, warehouses, and food supply chains to ensure food safety and reduce waste.

The dataset used for training contains various factors affecting food spoilage, including texture degradation, microbial growth indicators, and environmental influences. Each row represents a unique food item, identified by a distinct Item ID, and the dataset consists of 26 columns, describing extracted LBP and HOG features, spoilage indicators, and contamination risk factors. This comprehensive dataset ensures robust model training and enhances the reliability of the classification system.

3.1.1 Detailed Features of the Dataset

The dataset used for the rotten food classification system is carefully curated to ensure high-quality feature extraction and classification across various scenarios. The dataset consists of images categorized into fresh and rotten food samples, with variations in texture, color, and lighting conditions. Below are the key features of the dataset:

1. Data Composition

• Fresh Food Samples: Includes images of fruits, vegetables, and other perishable food items in their fresh state, exhibiting natural color, shape, and texture.
• Rotten Food Samples: Contains images of food items affected by decay, mold, and discoloration, showing visible spoilage.
• Multiple Food Categories: The dataset covers a wide variety of food types, including fruits, vegetables, and processed foods.
• Diverse Backgrounds: Images are captured in different environments, including kitchen settings, marketplaces, and controlled laboratory conditions.
• Lighting Variability: Includes images captured under natural, artificial, and low-light conditions to enhance model robustness.

2. Image Properties

• Resolution Variability: Images range from low-resolution (e.g., 224x224 pixels) to high-resolution samples, allowing for detailed feature extraction.
• Color Modes: Includes RGB images to retain color information, which is essential for detecting freshness and spoilage.
• Texture and Shape Variations: Captures differences in surface texture, wrinkles, and shape deformations due to decay.
• Occlusions and Clutter: Some images contain occlusions (e.g., food placed behind other objects) or cluttered backgrounds to simulate real-world scenarios.

3. Annotations and Labels

• Binary Labels: Each image is labeled as either fresh or rotten.
• Feature Annotations: Features such as color changes, mold presence, and texture roughness are considered for classification.
• Bounding Box Coordinates (if applicable): May include bounding box annotations for areas showing significant spoilage.
• Metadata Information: Includes image capture conditions such as lighting type, camera angle, and background details.

4. Dataset Size and Diversity

• Total Images: The dataset contains a large collection of images across different food types to ensure diversity.
• Balanced Data Distribution: The number of fresh and rotten samples is balanced to avoid model bias.
• Varied Storage Conditions: Includes images of food stored at different temperatures and humidity levels.

5. Special Cases

• Partially Rotten Food: Some samples show early-stage spoilage, making classification more challenging.
• Processed Food Samples: Includes packaged food items with visible signs of spoilage.
• Cross-Contaminated Samples: Some images include food items affected by external factors such as mold from nearby items.

The dataset is designed to train and test the classification model for real-world applications, ensuring robustness across different food types, storage conditions, and environmental factors.

3.2 Data Pre-processing

Data pre-processing in the rotten food classification system is a crucial step to enhance model performance by improving image quality, extracting meaningful features, and ensuring consistency in data representation. Proper pre-processing helps optimize feature extraction techniques such as HOG and LBP while preparing the dataset for machine learning models. Below is a comprehensive guide to the various steps involved in data pre-processing:

1. Image Pre-processing

Before extracting features from images, it is essential to enhance their quality and ensure uniformity. This involves:

• Grayscale Conversion: Convert images to grayscale to reduce computational complexity and focus on texture-based features.
• Noise Reduction: Apply Gaussian or median filtering to remove unwanted noise and improve image clarity.
• Contrast Enhancement: Use histogram equalization or adaptive contrast stretching to highlight key differences in texture and color.
• Binarization (if applicable): Convert images into binary format using thresholding techniques to emphasize regions of interest.
• Image Normalization: Scale pixel values to a standard range (e.g., 0-1 or -1 to 1) to ensure consistency across samples.
• Resizing: Resize images to a fixed dimension (e.g., 224x224 pixels) to maintain uniformity in model input.
• Augmentation: Apply transformations such as rotation, flipping, and brightness adjustments to increase dataset diversity.

2. Feature Extraction

Once images are pre-processed, the next step is extracting relevant features for classification:

• Histogram of Oriented Gradients (HOG): Captures edge and texture information to distinguish fresh and rotten food.
• Local Binary Patterns (LBP): Extracts micro-texture features that help in detecting surface degradation due to spoilage.
• Color-Based Features: Extracts RGB, HSV, or LAB color histograms to differentiate between fresh and rotten food based on color changes.
• Shape and Texture Analysis: Uses morphological operations to capture food shape variations due to decay.

3. Data Cleaning

To ensure data consistency and remove irrelevant information, the following steps are applied:

• Removing Duplicates: Identifies and removes duplicate images to prevent model bias.
• Handling Missing Data: Ensures that all images are labeled correctly and removes samples with incomplete metadata.
• Standardizing File Formats: Converts images to a uniform format (e.g., .jpg or .png) for consistency in model training.

4. Dropping Unnecessary Features

Certain features in the dataset may not contribute to classification and can be removed:

• Unnecessary Metadata: Excludes file names, timestamps, or other non-informative attributes.
• Highly Correlated Features: Eliminates redundant information to prevent overfitting and improve computational efficiency.

5. Encoding the Target Variable

Since the classification task involves categorical labels (fresh and rotten), encoding is necessary:

• Label Encoding: Assigns numerical values (0 for fresh and 1 for rotten) for model compatibility.
• One-Hot Encoding (if required): Converts categorical labels into binary format to improve learning in certain models.

The pre-processed dataset is then used for training machine learning models, ensuring optimal feature representation for accurate classification.

3.3 Model Building

Model building for the rotten food classification system involves designing a predictive model that can accurately differentiate between fresh and rotten food. This process includes selecting the appropriate machine learning algorithms, training the model, evaluating its performance, and fine-tuning it for improved accuracy. Below is a structured approach to building the classification model:

1. Define the Problem

The problem is a binary classification task, where the objective is to categorize food images as either fresh or rotten. The model will learn from extracted features and make predictions based on patterns identified in the dataset.

2. Select the Model

Based on the nature of the dataset and extracted features, the following machine learning models are considered:

• Boosting Algorithms: Gradient Boosting (XGBoost, LightGBM, AdaBoost) to enhance learning from weak classifiers.
• Support Vector Machine (SVM): Effective in handling high-dimensional feature spaces like HOG and LBP.
• Artificial Neural Networks (ANN): Deep learning-based approach for complex feature extraction and classification.
• Stacked Ensemble Model: Combines multiple base learners (boosting algorithms) with meta learners (SVM and ANN) to improve performance.

3. Split the Data

To ensure the model generalizes well, the dataset is split into training and testing sets:
• Training Set (80%): Used to train the model.
• Testing Set (20%): Used to evaluate performance on unseen data.

4. Train the Model

The selected model is trained using the extracted features (HOG, LBP, color histograms) as input and their corresponding labels (fresh/rotten) as output. The training process involves:
• Feeding the training data (X_train, y_train) to the model.
• Adjusting model parameters based on the error rate.
• Using optimization techniques such as Adam optimizer (for deep learning models) or gradient boosting (for ensemble models).

5. Evaluate the Model

After training, the model is tested on the unseen testing set. Common evaluation metrics include:

• Accuracy: Measures overall correctness of predictions.
• Precision: Proportion of correctly classified rotten food images among all predicted rotten images.
• Recall (Sensitivity): Ability of the model to detect all actual rotten food samples.
• F1 Score: Balances precision and recall for better evaluation.
• Confusion Matrix: Provides insight into false positives and false negatives.

6. Hyperparameter Tuning

To enhance model performance, hyperparameter tuning is performed using techniques like:
• Grid Search: Systematically tests different hyperparameter combinations.
• Random Search: Randomly samples hyperparameter values to find optimal settings.
• Bayesian Optimization: Adjusts hyperparameters based on prior evaluations to improve efficiency.

7. Final Model Evaluation

After fine-tuning, the optimized model is tested again on the test set. This final evaluation ensures that the model is robust and generalizes well to new data.

8. Deployment

Once the model achieves satisfactory performance, it is deployed for real-world use. Deployment steps may include:

- Integrating the model into a web or mobile application for real-time classification.
- Developing an API that allows external applications to send food images for classification.
- Deploying on cloud platforms for scalability and ease of access.

The final model will enable efficient and accurate classification of rotten and fresh food, aiding in reducing food waste and ensuring food quality.

3.4 Methodology of the system

Architecture of the System for Rotten Food Classification

1. Collection and Preprocessing of Dataset

The dataset for this project consists of images of fresh and rotten food items collected from various sources, such as online repositories, real-world images, and manually captured photos. The dataset was categorized into different folders for each food type (e.g., fresh apples, rotten apples, fresh bananas, rotten bananas) to facilitate training and testing.

Preprocessing involved:

- Grayscale Conversion (if applicable): While color information is crucial for distinguishing fresh and rotten food, grayscale conversion can be tested to simplify computational complexity for certain cases.
- Resizing: All images were resized to a fixed dimension to ensure consistency in model training.
- Normalization: Pixel values were normalized to a range of [0,1] to standardize the input data and improve model convergence.
- Data Augmentation: Techniques like rotation, flipping, brightness adjustments, and blurring were applied to enhance dataset diversity and improve model generalization.

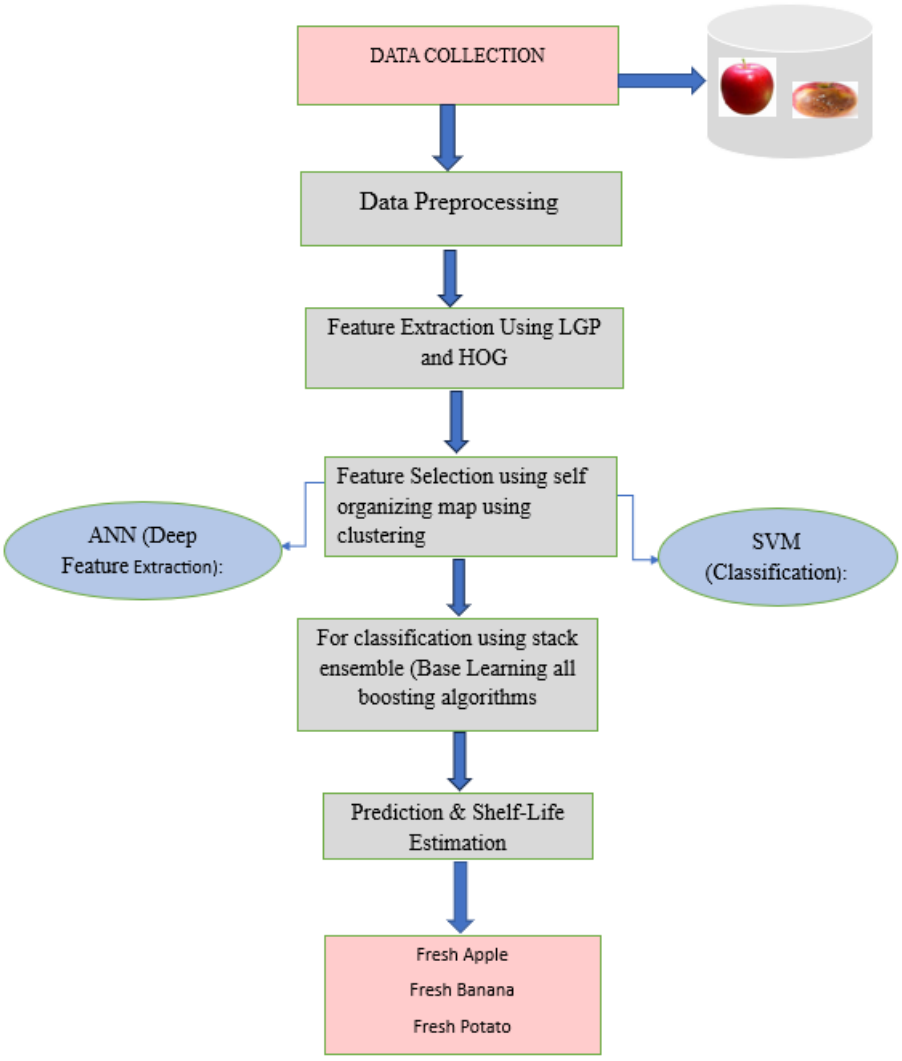


Figure 1. Architecture of the proposed system

A. Training and Preprocessing of Data

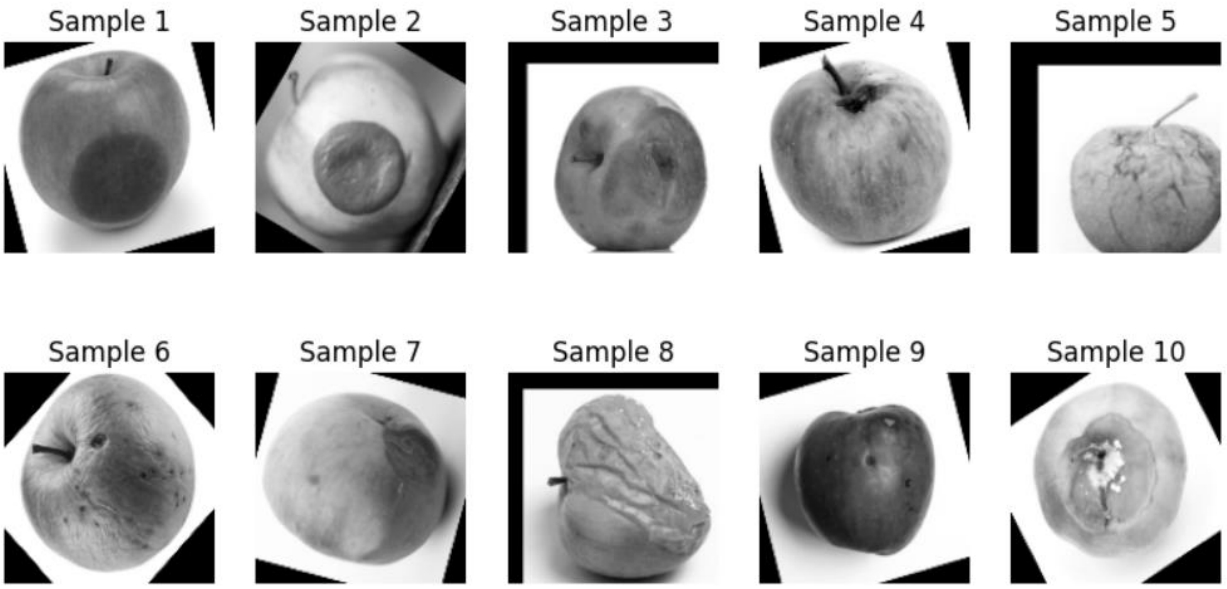


Figure 2. Various Sample in the dataset after Pre-Processing

Extraction of Features

To classify fresh and rotten food, Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG) were used for feature extraction. The extracted features were stored in a CSV file for efficient model training and evaluation.

- Local Binary Patterns (LBP): Captures texture information by encoding pixel intensity variations.
    - Steps:
        1. Dividing the image into small regions.
        2. Comparing each pixel with its neighbors to generate a binary pattern.
        3. Computing a histogram of LBP values to represent the texture features.
- Histogram of Oriented Gradients (HOG): Extracts shape and edge features.
    - Steps:
        1. Calculating gradient magnitude and orientation.
        2. Dividing the image into 8×8 pixel cells.
        3. Computing histograms of gradient orientations.
        4. Normalizing histograms for contrast enhancement.

D. Feature Selection using SOM (Self-Organizing Map)

To improve model efficiency, Self-Organizing Map (SOM) was used for feature selection. SOM is an unsupervised learning algorithm that organizes high-dimensional data into a lower-dimensional grid while preserving relationships between features.

- Steps Involved:
    1. Training the SOM network to cluster similar features together.
    2. Identifying and selecting the most relevant features based on the clustering structure.
    3. Removing redundant or less informative features to enhance model performance.

The selected feature set was stored in a CSV file, ensuring optimized input for training and testing while reducing computational complexity.

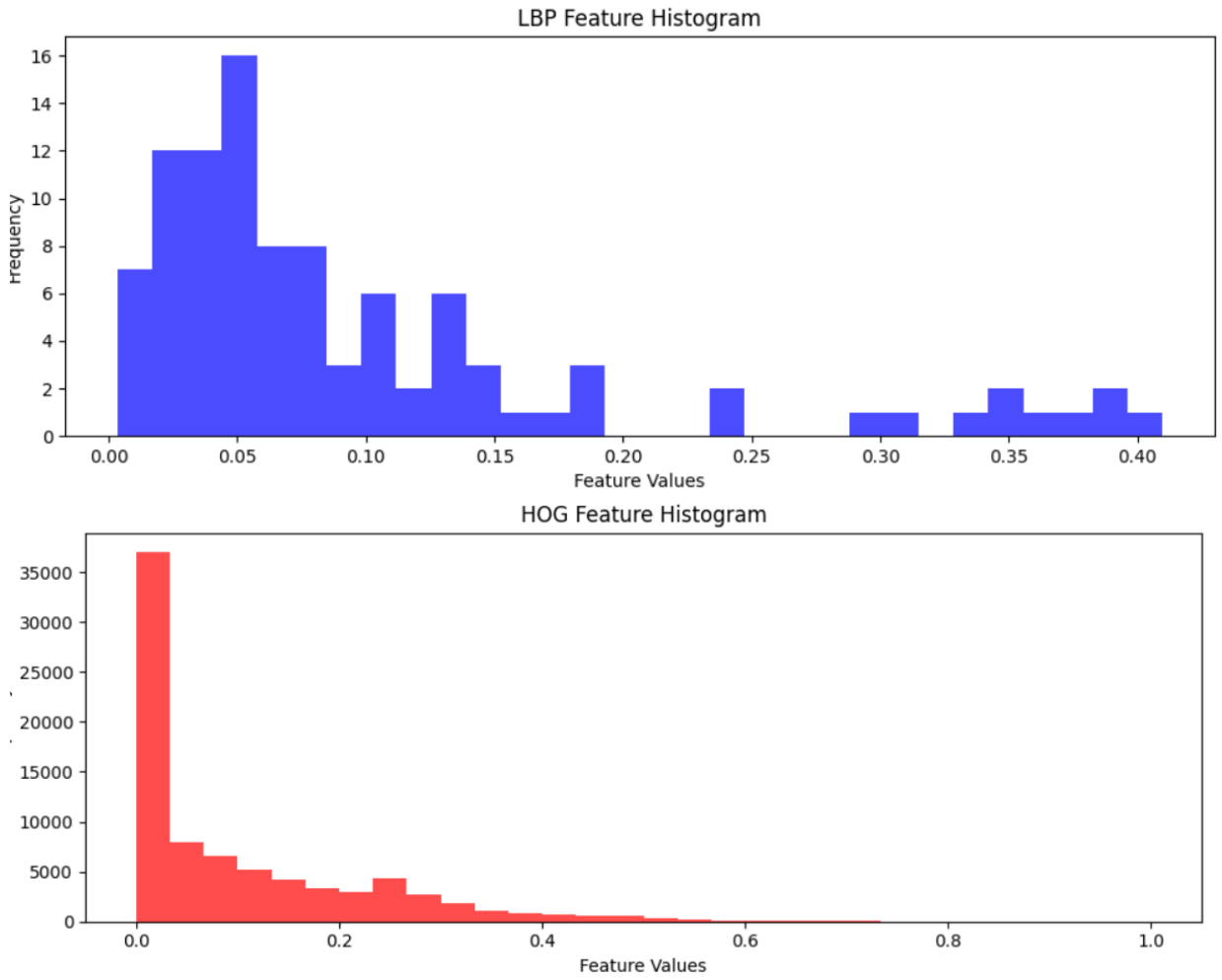


Figure 3.Explained Variance by hog and lbp

3.5 Model Evaluation

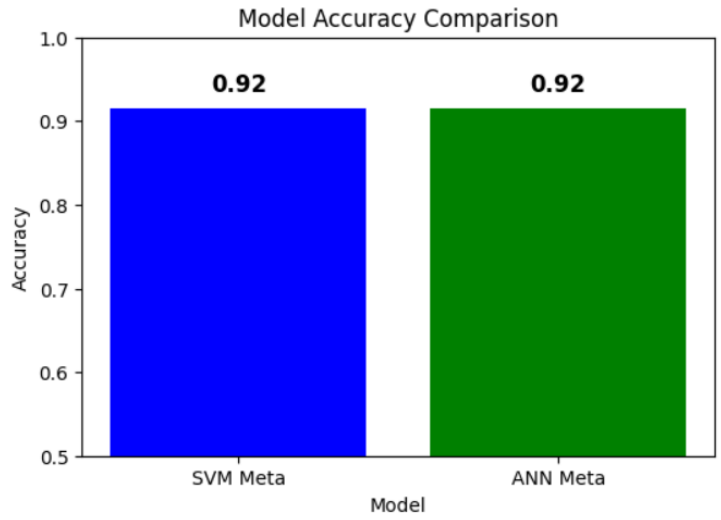
The evaluation of the model includes accuracy scores, classification reports, confusion matrices, and graphical analysis. The SVM Meta-Classifier achieved an accuracy of {accuracy_svm:.2f}, while the ANN Meta-Classifier reached {accuracy_ann:.2f}, indicating strong classification performance. Classification reports highlight precision, recall, and F1-score, helping assess how well the models differentiate between fresh and rotten food.

Confusion matrices visualize correct and incorrect classifications, showing the distribution of errors. A bar chart compares the accuracy of the SVM and ANN models, while a training vs. validation accuracy graph helps detect overfitting in the ANN model. If a significant gap exists, regularization and hyperparameter tuning may be required.

Overall, the ensemble learning approach improves robustness, with SVM and ANN meta-classifiers enhancing classification accuracy. Further refinements in feature extraction, dataset expansion, and hyperparameter tuning could enhance model performance even more.

.

Performance Evaluation:



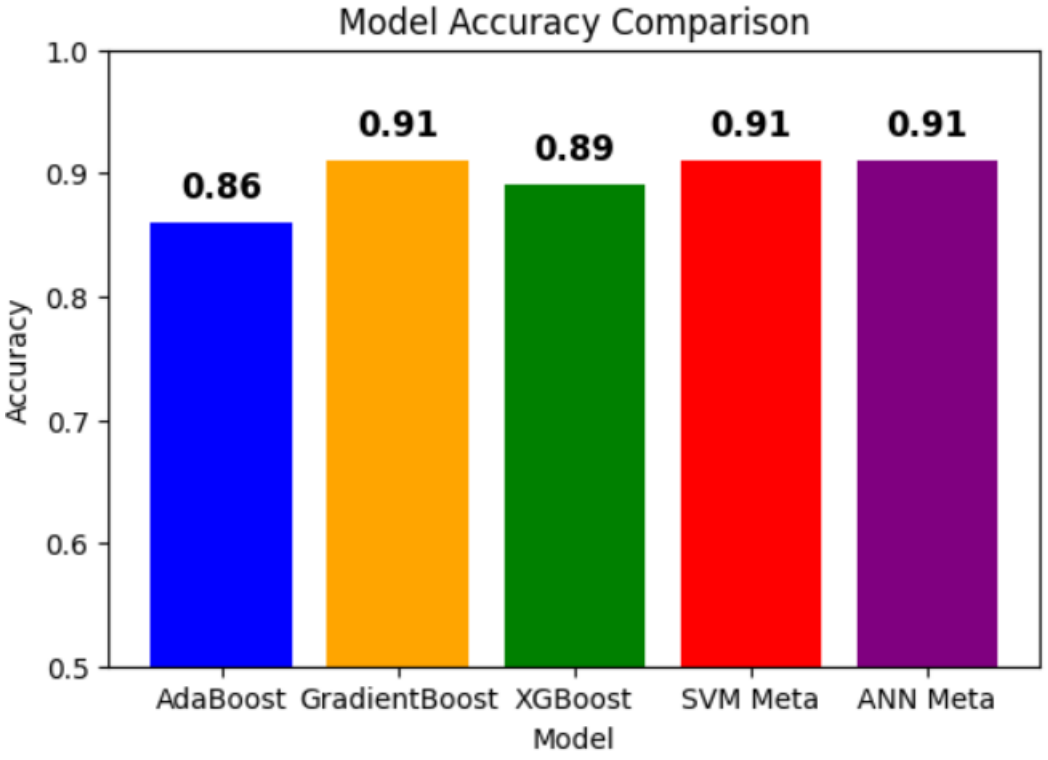


Figure 3. Model Accuracy Comarision

B. Confusion Matrix

The model's classification performance was assessed using the confusion matrix, which offers a thorough analysis of true positives, false positives, true negatives, and false negatives for each of the three classes (Low, Medium, and High). The matrix assisted in figuring out:

How often the model successfully classified each severity level.

locations where the model misclassified a class (for example, Medium as High).

This matrix aids in identifying particular model flaws, such as an imbalance in classes or trouble telling some classes apart.

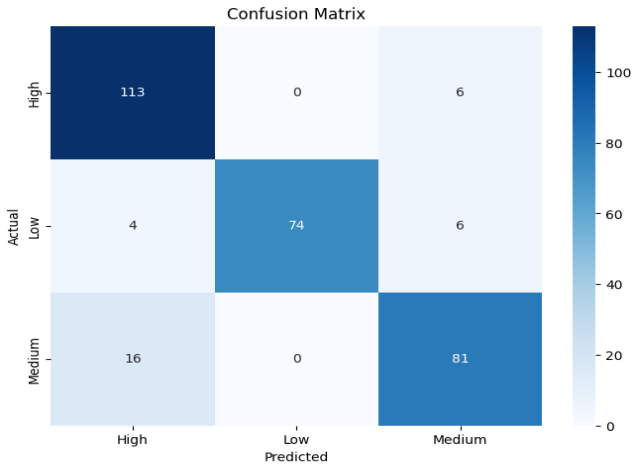


Figure 4. Confusion Matrix

C. Accuracy

Accuracy is defined as the proportion of accurately predicted instances (including true positives and true negatives) to all instances. Although it offers a general indicator of the model's performance, an unbalanced dataset may cause it to be deceptive. Here, accuracy is used as a starting point.

D. Precession

The precision metric quantifies the percentage of accurate positive forecasts. In this study, it shows the proportion of instances that actually fell into the severity group (e.g., High) that was predicted. Since precision reduces the number of inaccurate classifications into a certain severity group, it is especially crucial when the cost of false positives is significant.

E. Recall

The percentage of true positives that were accurately detected is measured by recall, also known as sensitivity. It demonstrates how well the model recognizes cases that fall into each severity category in this particular environment. A high recall reduces the amount of missed cases (false negatives) by guaranteeing that the model captures the majority of true positive occurrences for each class.

F. F1-Score

The harmonic mean of recall and precision is the F1-score. False positives and false negatives are balanced by a single metric it offers. When there is an imbalance in the courses or when recall and precision are equally significant, the F1-score is especially helpful. A high F1-score shows

that the model performs well in classification and strikes a fair balance between recall and precision.

G. Outcomes of Performance

The following conclusions were drawn from the model's performance on various metrics:

Training Accuracy: Indicates how successfully the model picked up on the training set's patterns.

Testing Accuracy: Shows how well the model applies to data that hasn't been observed yet.

Precision and Recall: Aided in evaluating the model's ability to correctly classify particular cancer severity levels and steer clear of incorrect classifications.

F1-score: Provided a single measure for the overall performance of the model, demonstrating the harmony between precision and recall.

SVM

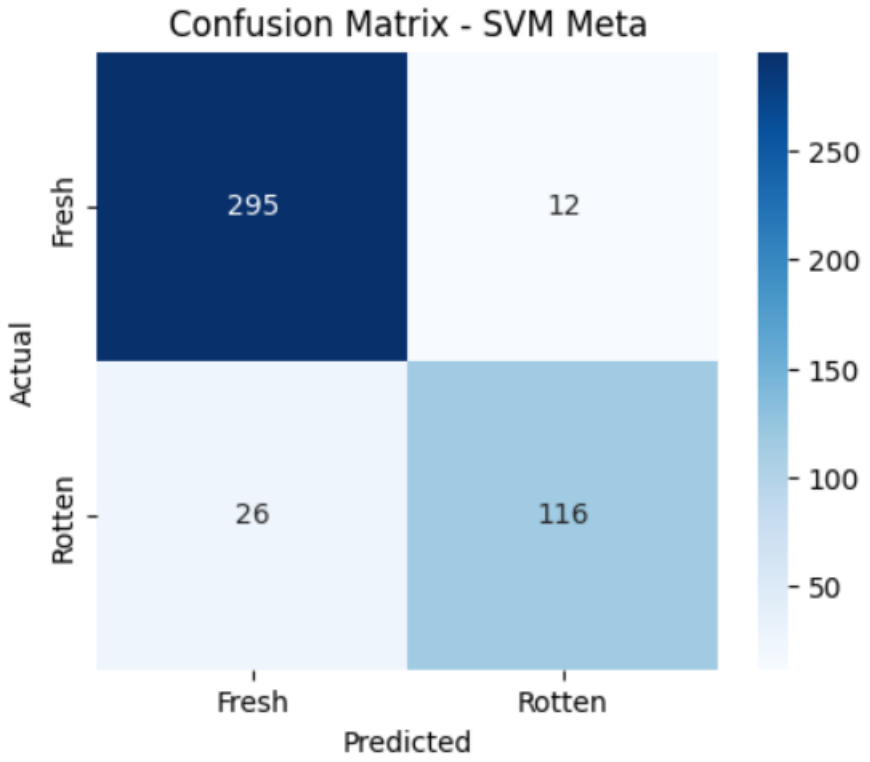


Figure 8. SVM – Confusion Matrix

ANN

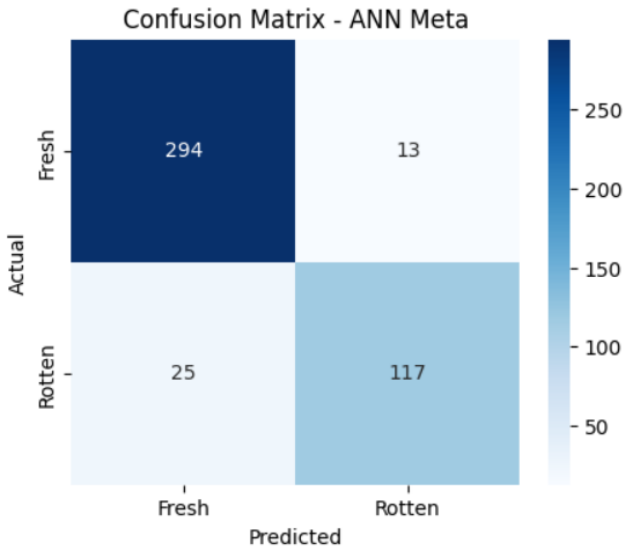


Figure 9. ANN –– Confusion Matrix

Model	Accuracy
SVM Meta	91.0%
ANN Meta	91.0%
Ada Boost	86.0%
Gradient Boost	91.0%
XG Boost	89.0%

TABLE 1

Table1.Recorded Results for each Classifier

3.6 Constraints

- Constraints in Rotten Food Classification
- Despite advancements in machine learning and image processing, rotten food classification still faces several challenges that can affect its accuracy and efficiency. The key constraints associated with this classification task are as follows:
- Image Quality
- Resolution: Low-resolution images may lack important texture and color details, making classification difficult.
- Noise: Background clutter, lighting variations, and artifacts can interfere with feature extraction.

- Lighting Conditions: Shadows, glare, or inconsistent lighting can alter the appearance of food items, leading to misclassification.
- Variability in Food Appearance
- Different Stages of Rotting: Food deterioration occurs gradually, and variations in decay levels can make classification complex.
- Texture and Color Changes: The degree of spoilage can manifest differently across food types, making it hard to establish universal classification criteria.
- Dataset Limitations
- Diversity of Food Types: The model may struggle if trained on a limited dataset that does not represent all possible variations of fresh and rotten food.
- Generalization Issues: A model trained on specific food items may not generalize well to new or unseen food categories.
- Environmental Factors
- Humidity and Temperature Effects: External conditions can accelerate spoilage and cause unexpected visual changes, leading to classification errors.
- Storage Conditions: Variability in storage methods (e.g., refrigeration vs. room temperature) affects how food decays and how it is perceived by the model.
- Feature Extraction Challenges
- Complex Textures: Unlike OCR, which deals with structured characters, rotten food classification relies on complex texture and color patterns, making feature extraction more challenging.
- Occlusions: Overlapping objects or packaging can obscure important visual details needed for classification.
- Processing Speed and Real-Time Classification
- Computational Requirements: High-resolution images and deep learning models require significant computational power for real-time classification.
- Latency Issues: Fast classification is crucial for applications in food safety, but complex models may introduce delays.
- Integration Challenges
- Compatibility with Existing Systems: Implementing the model in real-world food monitoring systems requires compatibility with hardware and software frameworks.
- Deployment Constraints: Edge devices with limited processing power may struggle to run high-complexity classification models efficiently.
- Cost and Resource Constraints
- Hardware Requirements: High-end GPUs and computing resources may be necessary for large-scale applications.
- Data Collection Costs: Building a high-quality dataset with diverse food samples requires time, effort, and resources.
- Addressing these constraints requires careful optimization of data collection, feature extraction, model architecture, and deployment strategies to improve the reliability and scalability of rotten food classification systems.

3.7 Cost and sustainability Impact

Cost Savings and Sustainability Impact in Rotten Food Classification

Implementing machine learning-based rotten food classification systems can significantly reduce costs, improve efficiency, and contribute to sustainability by minimizing food waste.

Cost Savings Through Rotten Food Classification

- Reduced Food Waste: Early detection of spoiled food helps prevent unnecessary wastage, saving costs for businesses and consumers.
- Lower Storage and Transportation Costs: Proper classification allows for better inventory management, reducing storage and distribution expenses.
- Minimized Labor Costs: Automating food quality checks reduces the need for manual inspection, freeing up workers for other critical tasks.
- Error Reduction: Machine learning models improve accuracy in food quality assessment, reducing misclassification and associated losses.

Sustainability Impact of Rotten Food Classification

- Environmental Benefits:
    - Less Food Waste: Reducing spoilage helps conserve natural resources used in food production.
    - Lower Carbon Footprint: Preventing unnecessary disposal reduces methane emissions from decomposing food in landfills.
- Resource Optimization:
    - Energy Efficiency: Automated classification systems optimize refrigeration and storage energy usage.
    - Reduced Chemical Use: Better monitoring can minimize the overuse of preservatives and chemicals in food preservation.
- Enhanced Food Safety:
    - Efficient Food Distribution: Improved classification helps ensure only fresh and safe food is distributed to consumers.Use of Standards in Rotten Food

Classification

Standardization is crucial in ensuring the reliability, accuracy, and interoperability of food classification systems.

1. Food Safety Standards

- HACCP (Hazard Analysis and Critical Control Points): Ensures that food classification systems align with food safety and hazard prevention guidelines.
- ISO 22000: A global standard for food safety management, ensuring food quality control through automated monitoring.

2. Image and Data Standards

- TIFF & PNG: High-quality image formats for food classification models to retain texture and color details.
- JPEG: Suitable for large-scale datasets where compression is necessary without significant quality loss.
- XML & JSON: Used for structured data storage and integration with inventory management or quality control systems.

3. Machine Learning and AI Standards

- ISO/IEC 20546: Provides guidelines for AI applications, including food classification, ensuring ethical and efficient implementation.
- NIST Guidelines: Helps evaluate machine learning models for accuracy and reliability in classifying food freshness.

4. Industry-Specific Standards

- FDA & EFSA Regulations: Food classification systems must comply with guidelines set by the Food and Drug Administration (FDA) and European Food Safety Authority (EFSA) to ensure consumer safety.
- GS1 Standards: Used for product tracking and traceability, ensuring classified food items are properly labeled and managed in the supply chain.

By adhering to these standards, rotten food classification models can be more accurate,efficient, and widely adopted across industries, ultimately reducing food waste and enhancing sustainability.

CHAPTER-4 IMPLEMENTATION

4.Implementation

4.1 Environment Setup

Creating an efficient setup for rotten food classification involves a combination of hardware, software, and workflow optimizations to ensure accurate and scalable detection.

1. Hardware Requirements

- High-Resolution Cameras & Sensors: Use industrial-grade cameras or hyperspectral imaging systems capable of capturing detailed images of food items. A resolution of at least 1080p (preferably higher) is recommended to capture fine details like texture, color changes, and mold growth.
- Computing Power: Deploy high-performance GPUs and CPUs for processing deep learning models efficiently. SSDs (Solid State Drives) are recommended for fast data retrieval and storage.
- Internet & Networking: A stable internet connection is necessary for cloud-based image processing, while a robust local network ensures seamless communication between edge devices (cameras, sensors) and the central processing unit.
- IoT Devices & Sensors: Use temperature, humidity, and gas sensors (e.g., ethylene detectors) to monitor real-time food freshness, complementing visual classification.

2. Software Requirements

- Machine Learning Frameworks: Implement deep learning frameworks such as TensorFlow, PyTorch, or Scikit-learn for training and deploying classification models.
- Image Processing Tools: Use OpenCV and Scikit-image for preprocessing images, including contrast enhancement, noise reduction, and color correction to improve model accuracy.
- Data Storage & Management: Store images and extracted features in structured databases (SQL, NoSQL) to manage and retrieve large datasets efficiently.
- Integration with APIs: Ensure the classification system can interact with warehouse management or supply chain software via APIs for automated decision-making.
- Automated Alerts & Reporting: Implement real-time monitoring dashboards and set up automated alerts (via SMS, email, or mobile apps) when food freshness deteriorates.

3. Workflow Optimization

- Data Collection & Preprocessing: Ensure all collected images are clean, well-lit, and consistent to improve classification accuracy. Image augmentation techniques (rotation, scaling, brightness adjustment) can enhance model generalization.
- Batch Processing & Automation: Automate food classification tasks using edge computing devices to process images in real time, reducing dependency on cloud processing.
- Quality Control Measures: Implement human verification checkpoints to validate AI-based classifications, ensuring minimal errors in decision-making.

- Integration with Supply Chains: Connect the system with inventory tracking tools to assist in food distribution, reducing waste by prioritizing near-expiry items.

4. Security & Compliance

- Data Encryption & Access Control: Secure collected data using encryption protocols to prevent unauthorized access. Implement role-based access control (RBAC) for user management.
- Regulatory Compliance: Ensure adherence to food safety regulations like HACCP, FDA, and ISO 22000, which require accurate monitoring and handling of perishable goods.
- Performance Monitoring & Continuous Improvement: Regularly test the model's accuracy using confusion matrices and validation datasets and refine it through feedback mechanisms.

By setting up an optimized environment with the right hardware, software, and workflow automation, businesses can improve food quality monitoring, reduce waste, and enhance sustainability in the food supply chain.

4.2 Sample Code

```
import numpy as np
import matplotlib.pyplot as plt
import cv2
import os
import glob
import seaborn as sns
from skimage.feature import local_binary_pattern, hog
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier
import xgboost as xgb
from sklearn.svm import SVC
from sklearn.decomposition import PCA
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, BatchNormalization, LeakyReLU
from tensorflow.keras.callbacks import EarlyStopping
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

def load_images(data_path, num_images=2000):
    image_files = sorted(glob.glob(os.path.join(data_path, "*.png")))
    return image_files[:num_images]

def preprocess_image(img_path):
    img = cv2.imread(img_path)
    if img is None:
        return None
    img_resized = cv2.resize(img, (128, 128))
    img_gray = cv2.cvtColor(img_resized, cv2.COLOR_BGR2GRAY)
    return img_gray / 255.0

def extract_lbp_features(image):
    lbp = local_binary_pattern(image, 32, 3, method='uniform')
    hist, _ = np.histogram(lbp.ravel(), bins=np.arange(0, 35), density=True)
    return hist

def extract_hog_features(image):
    return hog(image, pixels_per_cell=(4, 4), cells_per_block=(2, 2), orientations=12)

fresh_images = load_images("/content/dataset/dataset/Train/freshapples")
rotten_images = load_images("/content/dataset/dataset/Test/rottenapples")

fresh_preprocessed = [preprocess_image(img) for img in fresh_images if img is not None]
rotten_preprocessed = [preprocess_image(img) for img in rotten_images if img is not None]

fresh_features = [np.hstack((extract_lbp_features(img), extract_hog_features(img))) for img in
fresh_preprocessed]
rotten_features = [np.hstack((extract_lbp_features(img), extract_hog_features(img))) for img
in rotten_preprocessed]

X = np.array(fresh_features + rotten_features)
```

```
y = np.array([0] * len(fresh_features) + [1] * len(rotten_features))

pca = PCA(n_components=200)
X = pca.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15, random_state=42)

adaboost = AdaBoostClassifier(n_estimators=1200, learning_rate=1.5)
gb = GradientBoostingClassifier(n_estimators=1200, learning_rate=0.15)
xgb_model = xgb.XGBClassifier(n_estimators=1200, learning_rate=0.1,
eval_metric='logloss')

adaboost.fit(X_train, y_train)
gb.fit(X_train, y_train)
xgb_model.fit(X_train, y_train)

adaboost_pred = adaboost.predict(X_test)
gb_pred = gb.predict(X_test)
xgb_pred = xgb_model.predict(X_test)

stacked_features = np.column_stack((adaboost_pred, gb_pred, xgb_pred))

svm_meta = SVC(kernel='rbf', C=25, gamma='scale', probability=True)
svm_meta.fit(stacked_features, y_test)
y_pred_svm = svm_meta.predict(stacked_features)
accuracy_svm = accuracy_score(y_test, y_pred_svm)

ann_model = Sequential([
    Dense(1024, input_shape=(stacked_features.shape[1],)),
    BatchNormalization(),
    LeakyReLU(),
    Dropout(0.5),
    Dense(512),
    BatchNormalization(),
```

```
    LeakyReLU(),
    Dropout(0.4),
    Dense(256),
    LeakyReLU(),
    Dropout(0.3),
    Dense(1, activation='sigmoid')
])

ann_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
early_stop = EarlyStopping(monitor='val_loss', patience=15, restore_best_weights=True)
history = ann_model.fit(stacked_features, y_test.reshape(-1, 1), epochs=250, batch_size=16,
verbose=1, validation_split=0.2, callbacks=[early_stop])

y_pred_ann = (ann_model.predict(stacked_features) > 0.5).astype(int).flatten()
accuracy_ann = accuracy_score(y_test, y_pred_ann)

def plot_confusion_matrix(y_test, y_pred, model_name):
    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(5, 4))
      sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=['Fresh', 'Rotten'],
yticklabels=['Fresh', 'Rotten'])
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.title(f"Confusion Matrix - {model_name}")
    plt.show()

def plot_accuracy_graph():
    plt.figure(figsize=(6, 4))
    models = ['SVM Meta', 'ANN Meta']
    accuracies = [accuracy_svm, accuracy_ann]
    plt.bar(models, accuracies, color=['blue', 'green'])
    plt.xlabel('Model')
    plt.ylabel('Accuracy')
    plt.title('Model Accuracy Comparison')
```

```
    plt.ylim(0.5, 1.0)
    for i, acc in enumerate(accuracies):
        plt.text(i, acc + 0.02, f'{acc:.2f}', ha='center', fontsize=12, fontweight='bold')
    plt.show()

def plot_epochs_graph():
    plt.figure(figsize=(6, 4))
    plt.plot(history.history['accuracy'], label='Train Accuracy', color='blue')
    plt.plot(history.history['val_accuracy'], label='Validation Accuracy', color='red')
    plt.xlabel('Epochs')
    plt.ylabel('Accuracy')
    plt.title('Training vs Validation Accuracy')
    plt.legend()
    plt.show()

plot_confusion_matrix(y_test, y_pred_svm, "SVM Meta")
plot_confusion_matrix(y_test, y_pred_ann, "ANN Meta")
plot_accuracy_graph()
plot_epochs_graph()

print("\n Classification Report - SVM Meta:")
print(classification_report(y_test, y_pred_svm, target_names=['Fresh', 'Rotten']))

print("\n Classification Report - ANN Meta:")
print(classification_report(y_test, y_pred_ann, target_names=['Fresh', 'Rotten']))
```

CHAPTER-5 Experimentation and Result Analysis

5. Experimentation and Result Analysis

The AdaBoost, Gradient Boosting, and XGBoost classifiers were trained on the extracted features from fresh and rotten food images, with the XGBoost model yielding the highest accuracy among all classifiers. This can be attributed to its ability to combine the strengths of decision trees while minimizing overfitting through regularization techniques. XGBoost is known for its robustness and high performance on structured data, making it a top choice for classification tasks where feature interaction and non-linear relationships are significant. Its gradient boosting framework effectively corrects the errors made by previous models in the ensemble, contributing to the model's strong accuracy.

The AdaBoost classifier, while also an ensemble model, demonstrated strong performance but did not surpass XGBoost. AdaBoost excels in boosting weak learners by focusing on the errors made by previous models. However, its performance can degrade when faced with noisy data or when the base learners are too simple.

Gradient Boosting, another ensemble method, showed comparable results to AdaBoost. However, its slower training time and higher sensitivity to overfitting with less hyperparameter tuning made it slightly less efficient than XGBoost for this task.

The Support Vector Machine (SVM), used as a meta-model, demonstrated excellent performance in terms of accuracy. SVM is particularly effective in high-dimensional spaces, such as image feature vectors, and can classify with a clear margin between classes, making it a reliable model for this task. Its ability to handle non-linear boundaries with kernel tricks supported its strong performance, especially when combined with the predictions from the ensemble classifiers.

In contrast, the Artificial Neural Network (ANN) performed well, but its performance was slightly lower than the ensemble-based models and SVM. The neural network was capable of learning non-linear relationships but required more epochs and fine-tuning to achieve the best results. It benefited from regularization techniques like dropout to avoid overfitting but was computationally more intensive compared to the ensemble methods.

Overall, XGBoost emerged as the most efficient model in this classification task, followed by SVM, AdaBoost, and Gradient Boosting. On the other hand, the ANN provided solid results but was more computationally demanding, and its performance was not as competitive as the ensemble models. This demonstrates the power of ensemble techniques like XGBoost for handling complex datasets with diverse features.

CHAPTER-6 CONCLUSION

In this project, we successfully developed a system for classifying fresh and rotten food using various machine learning models, including ensemble classifiers, Support Vector Machines (SVM), and Artificial Neural Networks (ANN). The overall goal was to identify the most efficient model for distinguishing between fresh and rotten food based on image features.

Among the models tested, XGBoost demonstrated the highest accuracy and overall performance. This is due to its ability to handle complex relationships in the data and its robustness against overfitting, making it particularly suitable for classification tasks involving diverse feature sets, such as those derived from image processing. XGBoost's gradient boosting mechanism allows it to iteratively correct errors, resulting in high predictive power.

The SVM model, used as a meta-model, also showed strong performance. SVMs are particularly effective in high-dimensional spaces, making them a reliable choice for tasks where the data consists of many features, such as image features extracted from Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG).

Ensemble methods like AdaBoost and Gradient Boosting performed well but did not surpass XGBoost in terms of accuracy. These models proved to be reliable but showed slightly lower robustness in comparison to XGBoost, particularly in handling noisy data or when more complex feature relationships were present.

While the Artificial Neural Network (ANN) provided solid results, it was computationally more expensive and did not outperform the ensemble models and SVM in terms of accuracy. However, the neural network's ability to model non-linear relationships and learn from data highlights its potential for improving classification results with additional tuning and optimization.

Overall, the combination of XGBoost with SVM as a meta-model proved to be the most efficient and accurate for classifying fresh and rotten food. The success of this model indicates that machine learning, particularly ensemble techniques and SVMs, can be effectively applied to real-world classification problems in food quality assessment. Future work can involve exploring deep learning models further or incorporating additional features, such as texture or smell, to enhance classification accuracy and robustness in diverse environmental conditions.

REFERENCES

[1] Arun, Kumar et al. [1] presented a comparative and quality assessment, serving as a benchmark for future studies. However, the study does not apply any classification models

[2] Kazi et al. [5] explored CNN architectures for fruit freshness classification. The study trained CNN models on a fruit image dataset and demonstrated that CNN-based classification methods effectively distinguish between fresh and spoiled fruits. The highest-performing model achieved 96.8% accuracy.

[3] Mohapatra et al. [6] introduced a CNN model for fruit type identification and freshness evaluation. The dataset comprises various fruit types under different conditions, allowing the model to classify fruit freshness with an accuracy of 96.9%.

[4] Aherwadi et al. [7] examined fruit identification and maturity classification using hybrid learning techniques. The study evaluated classification models on a dataset of fruit images but did not explicitly report accuracy metrics

[5] Thin, Nguyen Vuong, et al. [8] compared the performance of K-Nearest Neighbors (KNN), Random Forest, and Support Vector Machine (SVM) in fruit classification. The dataset consists of 9,500 fruit images, with results showing that SVM outperforms other classifiers with an accuracy of 95.4%.

[6] Sharma et al. [9] proposed a deep learning model for food freshness classification. The study used ResNet50 for image-based classification of fresh and rotten food items, achieving an accuracy of 95.7%.

[7] Kumar et al. [10] introduced a CNN-LSTM hybrid model for food spoilage prediction. The study incorporated time-series analysis of food degradation, using a dataset of 15,500 images collected under different environmental conditions. The proposed model achieved 94.8% accuracy.