**Project Title:** Smart Sorting Transfer Learning for Identifying Rotten Fruits and Vegetables

**Branch Name:** Computer science and Engineering

**Track:** Artificial Intelligence and Machine learning

**Team Lead:** Tirumalasetti Chandrika

**Mail Id:**[chandrika6304827@gmail.com](file:///C:\Users\chigu\AppData\Local\Microsoft\Windows\INetCache\IE\YAR8MGSB\chandrika6304827@gmail.com)

**Team Members:**

* Tammisetti guna Srinivas
* Tejeswari Potnuru
* Tadivalasa Jayasri

**Email Id's:**

* [Gunsrinivas17@gmail.com](mailto:Gunsrinivas17@gmail.com)
* [tejeswaripotnuru2005@gmail.com](mailto:tejeswaripotnuru2005@gmail.com)
* [Sirijayasri08@gmail.com](mailto:Sirijayasri08@gmail.com)

# Submitted By:

* Tirumalasetti Chandrika
* Seshadri Rao Gudlavalleru Engineering College
* Computer Science and Engineering
* Roll No: 22481A05o0

# Submitted To:

SmartBridge

# Abstract:

This project presents Smart Sorting, a computer vision-based solution that leverages Transfer Learning to identify and classify rotten fruits and vegetables. The primary objective is to automate the quality control process in agriculture, reducing human error and improving efficiency using a VGG16-based Convolutional Neural Network (CNN) model.

# Introduction:

Manual sorting of perishable goods like fruits and vegetables is inefficient and prone to errors. By automating this process using deep learning models, particularly those pre-trained on large datasets (transfer learning), this project introduces a scalable, accurate, and cost-effective solution.

# Problem Statement:

Quality control in the food supply chain is a significant challenge. This project addresses:

* + Manual inefficiencies in produce sorting
  + Waste due to late spoilage detection
  + Need for real-time sorting in homes and markets

**ARCHITECTURE:**



**PREREQUISITES :**

* To complete this project, you must require the following software, concepts, and packages
  + Anaconda Navigator:
    - Refer to the link below to download Anaconda Navigator
  + Python packages:
  + Open anaconda prompt as administrator
  + Type “pip install numpy” and click enter.
  + Type “pip install pandas” and click enter.
  + Type “pip install scikit-learn” and click enter.
  + Type ”pip install matplotlib” and click enter.
  + Type ”pip install scipy” and click enter.
  + Type ”pip install seaborn” and click enter.
  + Type ”pip install tenser flow” and click enter.
  + Type “pip install Flask” and click enter.

1. **PRIOR KNOWLEDGE**

* DL Concepts
  + Neural Networks:: [https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/](https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/%20)
  + Deep Learning Frameworks:: [https://www.knowledgehut.com/blog/data-science/pytorch-vs-tensorflow](about:blank)
  + Transfer Learning: [https://towardsdatascience.com/a-demonstration-of-transfer-learning-of-vgg-convolutional-neural-network-pre-trained-model-with-c9f5b8b1ab0a](https://towardsdatascience.com/a-demonstration-of-transfer-learning-of-vgg-convolutional-neural-network-pre-trained-model-with-c9f5b8b1ab0a%20)
  + VGG16: <https://www.geeksforgeeks.org/vgg-16-cnn-model/>
  + Convolutional Neural Networks (CNNs): [https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/ s://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning](https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/%20s:/www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning)
  + Overfitting and Regularization: [https://www.analyticsvidhya.com/blog/2021/07/prevent-overfitting-using-regularization-techniques/](https://www.analyticsvidhya.com/blog/2021/07/prevent-overfitting-using-regularization-techniques/%20)
  + Optimizers: [https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/](https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/%20ttps:/www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/)
* Flask Basics: <https://www.youtube.com/watch?v=lj4I_CvBnt0>

1. **PROJECT OBJECTIVES**

By the end of this project, you will:

* Know fundamental concepts and techniques used for Deep Learning.
* Gain a broad understanding of data.
* Have knowledge of pre-processing the data/transformation techniques on outliers and some visualization concepts.

1. **PROJECT FLOW**

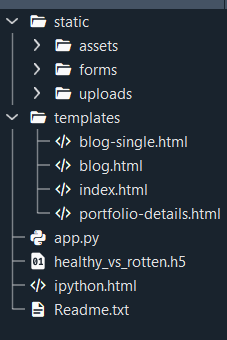
* The user interacts with the UI (User Interface) to choose the image.
* The chosen image is analysed by the model which is integrated with the flask application.
* Once the model analyses the input the prediction is showcased on the UI

To accomplish this, we have to complete all the activities listed below,

* Data Collection: Collect or download the dataset that you want to train.
* Data pre-processing
  + Data Augmentation
  + Splitting data into train and test
* Model building
  + Import the model-building libraries
  + Initializing the model
  + Training and testing the model
  + Evaluating the performance of the model
  + Save the model
* Application Building
  + Create an HTML file
  + Build python code

**Project Structure**

Create the Project folder which contains files as shown below



* We are building a Flask application with HTML pages stored in the templates folder and a Python script app.py for scripting.
* Healthy\_vs\_rotten.h5 is our saved model. Further, we will use this model for flask integration.

**DATA COLLECTION AND PREPARATION**

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

1. **COLLECTING THE DATASET**

It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

Activity 1: Download the dataset

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project, we have used 28 classes of fruits and vegetables data. This data is downloaded from kaggle.com or can be connected by using API. Please refer to the link given below to download the dataset.

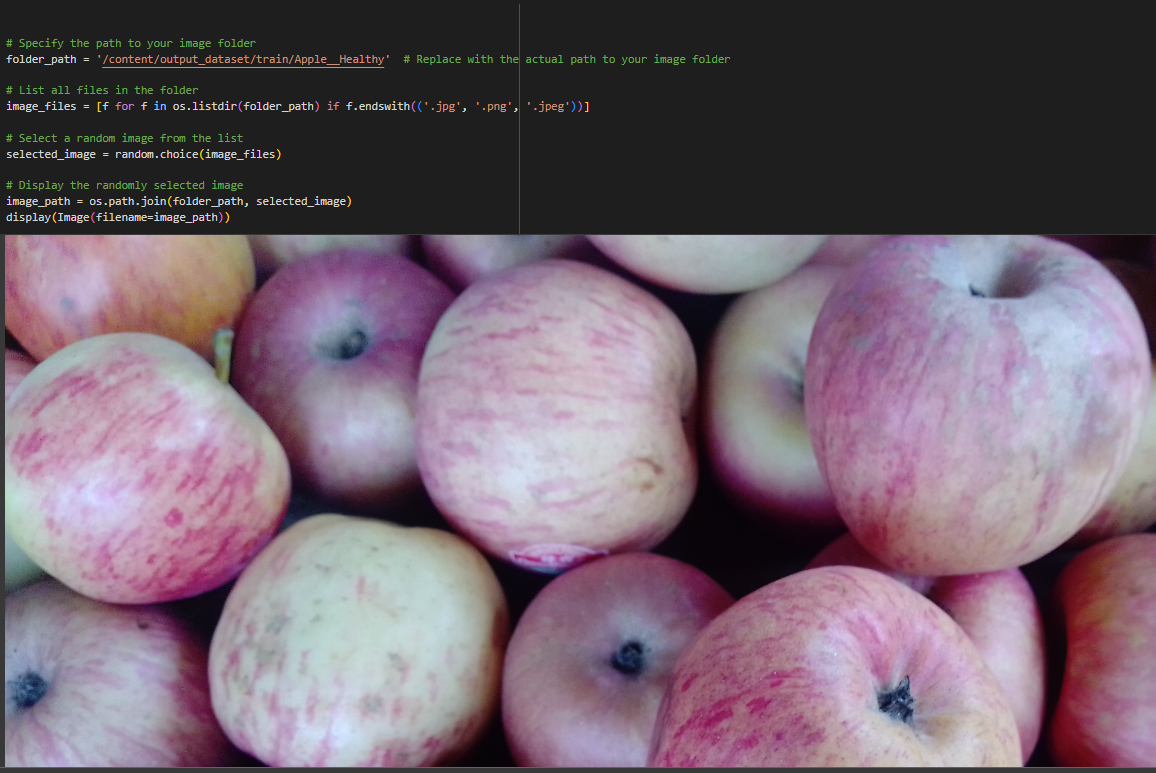
Link:  [Dataset](https://www.kaggle.com/datasets/muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten)

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analyzing techniques.

Note: There are several techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

1. **Data Visualization**

The provided Python code imports necessary libraries and modules for image manipulation. It selects a random image file from a specified folder path. Then, it displays the randomly selected image using IPython's Image module. This code is useful for showcasing random images from a directory for various purposes like data exploration or testing image processing algorithms.



1. **DATA AUGMENTATION**

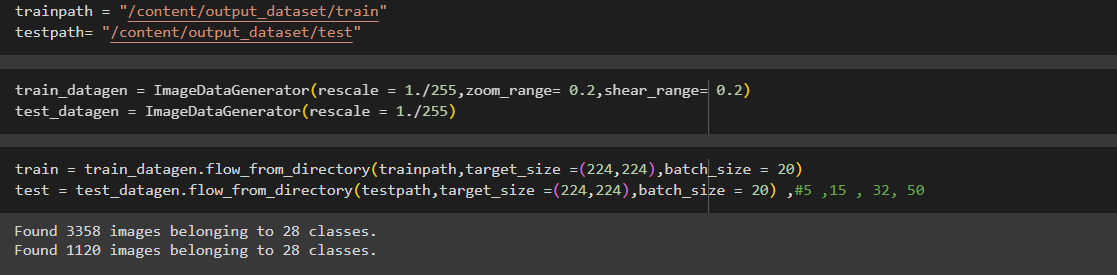
Data augmentation is a technique commonly employed in machine learning, particularly in computer vision tasks such as image classification, including projects like the healthy vs rotten Classification  in fruits and vegetables. The primary objective of data augmentation is to artificially expand the size of the training dataset by applying various transformations to the existing images, thereby increasing the diversity and robustness of the data available for model training. This approach is particularly beneficial when working with limited labeled data.

In the context of the 28  class Classification, data augmentation can involve applying transformations such as rotation, scaling, flipping, and changes in brightness or contrast to the original images of fossils. These transformations help the model generalize better to variations and potential distortions present in real-world images, enhancing its ability to accurately classify unseen data.

This is a crucial step but this data is already cropped from the augmented data so. this time it is skipped accuracy is not much affected but the training time increased.

**SPLIT DATA AND MODEL BUILDING**

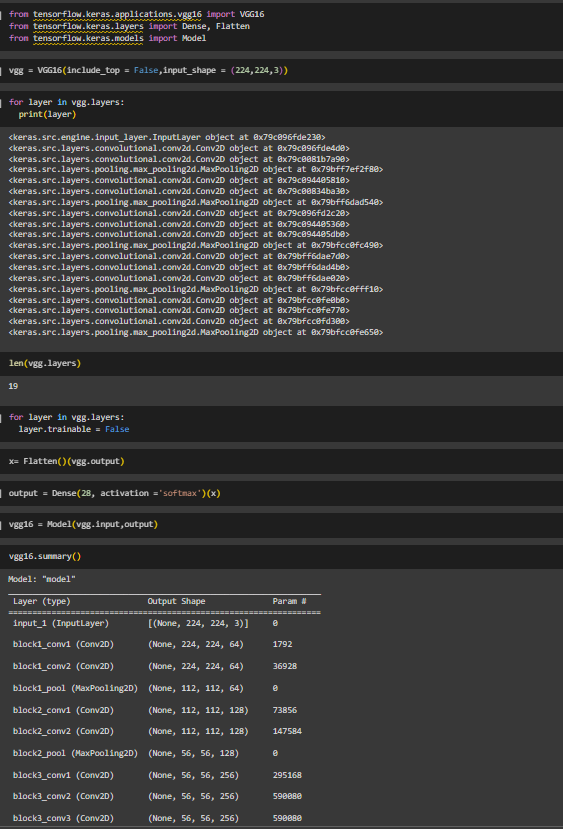
Train-Test-Split: In this project, we have already separated data for training and testing.



A) **Model Building:**

Vgg16 Transfer-Learning Model:

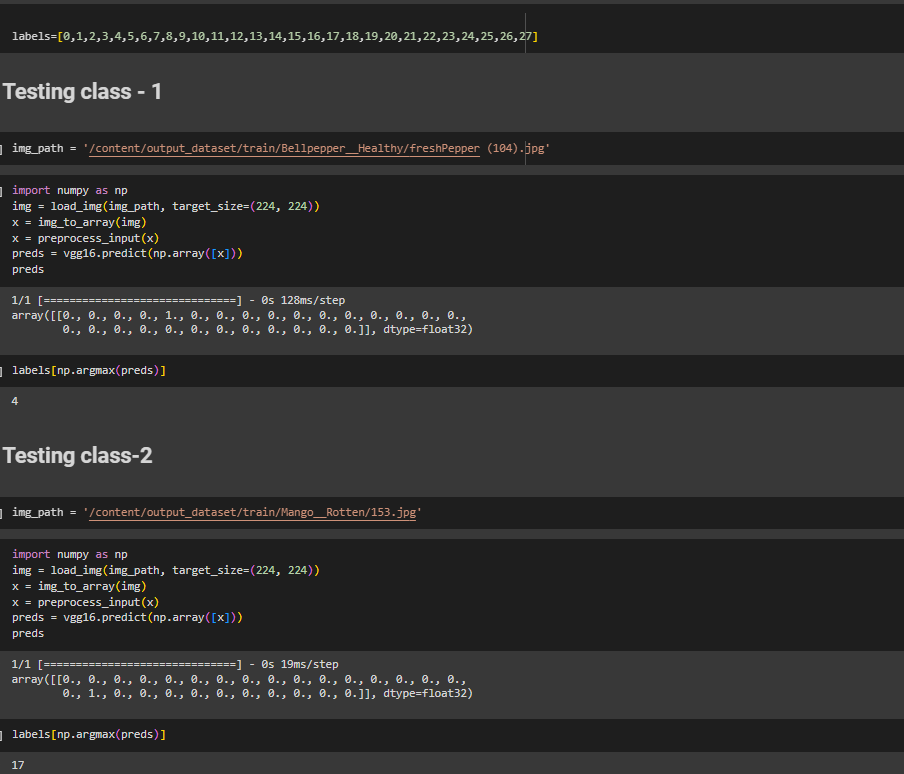
The VGG16-based neural network is created using a pre-trained VGG16 architecture with frozen weights. The model is built sequentially, incorporating the VGG16 base, a flattening layer, dropout for regularization, and a dense layer with SoftMax activation for classification into five categories. The model is compiled using the Adam optimizer and sparse categorical cross-entropy loss. During training, which spans 10 epochs, a generator is employed for the training data, and validation is conducted, incorporating call-backs such as Model Checkpoint and Early Stopping. The best-performing model is saved as "healthy\_vs\_rotten.h5 " for potential future use. The model summary provides an overview of the architecture, showcasing the layers and parameters involved.



**TESTING MODEL & DATA PREDICTION**

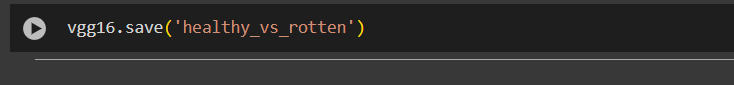
Testing the model

Here we have tested with the Vgg16 Model With the help of the predict () function.



**SAVING THE MODEL**

Finally, we have chosen the best model now saving that model



# Objective:

* + Develop a machine learning model for healthy vs. rotten classification.
  + Implement a web-based application.
  + Reduce waste and human effort.

# Scenarios of Application:

1. Food Processing Plants
2. Supermarkets
3. Smart Home

# Tools and Technologies Used:

Anaconda, Python, Flask

Libraries: NumPy, Pandas, Scikit-learn, TensorFlow, Matplotlib

# Concepts and Prerequisites:

* + Deep Learning & CNN
  + Transfer Learning using VGG16
  + Flask for web deployment

# Methodology:

* + Dataset: 28-class fruit/vegetable images from Kaggle
  + Preprocessing: Normalization, Augmentation
  + Model: VGG16 with SoftMax, trained over 10 epochs
  + Accuracy validated via prediction samples

# System Architecture:

1. UI Input
2. Flask API
3. Model Prediction
4. Output Displayed

**Application Building**

In this section, we will be building a web application that is integrated into the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

* Building HTML Pages
* Building server-side script

# Results and Observations:

The model accurately classified various samples like:

* + Potato rotten
  + Strawberry healthy with correct prediction

# Conclusion:

This project shows how transfer learning enables efficient classification and automation in agriculture. It reduces waste and labor dependency.

# Future Scope:

* + Broader spoilage detection
  + IoT integration
  + Mobile app support

1. **References:** Kaggle Dataset GeeksforGeeks VGG16 Analytics Vidhya Bl