**MACHINE LEARNING(CS-7830-01)**

**PROJECT TITLE:**

**Plant Disease Classification using Fuzzy KNN and MLP**

**TEAM:**

1. **Kartheek Kodati(U01033252)**
2. **Sneha Mirupala(U01033363)**
3. **Tejaswini Minupala(U01066894)**

**Wright State University**

**Dayton campus, Ohio.**

**SUBMITTED TO:**

**Tanvi Banerjee**

**On 23rdApril 2023**

**Table of Content**

**s**

1. Abstract……………………………………………………….3

2. Introduction…………………………………………………...4

3.Related Work…………………………………………………..5

4.Design of the project…………………………………………...7

5.Validation Method…………………………………………….11

6.Modules………………………………………………………..12

7.Implementation………………………………………………...15

8.Flow Chart……………………………………………………..16

9.Bar Graph……………………………………………………...17

10.Results………………………………………………………..18

11.Input&Output…………………………………………………19

12.Conclusion&Future Work…………………………………….20

**Abstract**

Plant diseases are a significant concern for agricultural communities worldwide, causing severe economic losses and affecting food security. The early detection and identification of plant diseases can help prevent their spread and minimize crop damage. In recent years, machine learning algorithms have shown promising results in automating the detection and classification of plant diseases. In this project, we have developed a plant disease classification system using two machine learning algorithms, Fuzzy K-Nearest Neighbor (Fuzzy KNN) and Multi-Layer Perceptron (MLP). Fuzzy KNN is a variant of KNN that uses fuzzy logic to classify data. MLP is a type of artificial neural network that is widely used for classification tasks.

The proposed system involves the following steps: data acquisition, pre-processing, feature extraction, feature selection, model training, and testing. We used the Plant Village dataset, which contains images of healthy and diseased leaves of different plant species, to train and evaluate our models. For pre-processing, we performed image resizing and normalization to ensure that all images were of the same size and had similar lighting conditions. We trained our models on the pre-processed and selected feature dataset using Fuzzy KNN and MLP algorithms. We evaluated the performance of our models using accuracy, precision, recall, and F1-score metrics. Our experimental results show that both Fuzzy KNN and MLP algorithms achieved high classification accuracy, with Fuzzy KNN achieving an accuracy of 69%, and MLP achieving an accuracy of 71%. We also observed that the combination of feature extraction and selection techniques significantly improved the classification performance of both algorithms.

**Introduction**

In order to meet the growing demand for food, agricultural problems must be addressed using cutting-edge technologies. In this regard, the agricultural industries are putting a strong emphasis on the use of artificial intelligence. In order to perform various agricultural operations, several traditional machine learning (ML) algorithms have been implemented. On top of that, deep learning (DL) has resulted in significant advancements in the field of agricultural research. This is due to the ability of deep learning algorithms to automatically extract features from input data. The successful classification of plant diseases is critical to improving the quality and quantity of agricultural products while also reducing the use of harmful chemical sprayers such as fungicides and herbicides. This is one of several agricultural problems that must be addressed.

As a result, it is a new research topic that has the potential to advance agricultural automation. This agricultural task is complicated due to the similarity in the occurrence of plant-containing diseases between the two situations. Several studies have been carried out in this regard in order to improve the classification of plant diseases in general. In the majority of cases, agricultural and forestry experts are called in to identify fruit tree diseases on the spot, or farmers identify fruit tree diseases themselves. Pests that have been identified through experience This method is not only subjective, but it is also time-consuming, labor-intensive, and inefficient, as demonstrated above.

Farmers with less experience may make erroneous judgments and use drugs without considering the consequences of their actions during the identification process. It is also possible that improved quality and output will result in environmental pollution, which will result in unneeded economic losses. In order to address these issues, research into the application of image-processing techniques for plant disease recognition has risen to the top of the research agenda.

**Related Work**

Plant disease detection and classification using machine learning algorithms have gained considerable attention in recent years due to their potential to improve the accuracy and efficiency of disease diagnosis in crops. Several researchers have proposed various techniques for plant disease detection and classification using different machine-learning algorithms. In this section, we discuss some of the related work done in this field.

One of the commonly used machine learning algorithms for plant disease classification is Convolutional Neural Networks (CNNs). CNNs have shown promising results in detecting and classifying plant diseases. Mohanty et al. (2016) proposed a CNN-based system for plant disease classification using the PlantVillage dataset. Their model achieved an accuracy of 99.35% for 38 plant disease classes. Kavdir et al. (2021) also proposed a CNN-based system using transfer learning for plant disease classification with an accuracy of 96.65%.

In addition to CNNs, other machine-learning algorithms have also been used for plant disease classification. Fuzzy logic-based techniques have been used to classify plant diseases in several studies. Xu et al. (2021) proposed a fuzzy decision tree-based system for apple disease diagnosis with an accuracy of 93.6%.

Support Vector Machines (SVMs) have also been used for plant disease classification. Wagh et al. (2021) proposed an SVM-based system for detecting and classifying grapevine diseases with an accuracy of 93.9%. Random Forests (RF) have also been used for plant disease classification. Kavdir et al. (2019) proposed an RF-based system for tomato disease detection and classification with an accuracy of 97.66%.

In terms of feature extraction, several techniques have been used to extract meaningful features from plant images. Deep learning-based feature extraction techniques, such as Convolutional Neural Networks (CNNs), have been shown to be effective in plant disease classification. Traditional feature extraction techniques such as Gabor filters, Local Binary Patterns (LBP), and Gray-Level Co-occurrence Matrix (GLCM) have also been used in several studies.

Some researchers have proposed the use of traditional machine learning algorithms such as k-Nearest Neighbors (k-NN), Decision Trees, and Naive Bayes for plant disease classification. Verma et al. (2021) proposed a k-NN-based system for identifying rice diseases with an accuracy of 98.89%. Al-Saleh et al. (2020) proposed a decision tree-based system for apple disease diagnosis with an accuracy of 94.3%. Gupta et al. (2019) proposed a Naive Bayes-based system for maize disease diagnosis with an accuracy of 92.8%. Apart from the above-mentioned algorithms, other techniques such as Ensemble methods and Deep Belief Networks (DBNs) have also been used for plant disease classification. Singh et al. (2021) proposed an ensemble-based system for cotton leaf disease classification with an accuracy of 96.4%. Huang et al. (2019) proposed a DBN-based system for identifying maize diseases with an accuracy of 98.1%.

Several studies have also focused on the development of specialized datasets for plant disease classification. For example, Milioto et al. (2019) proposed the Plant Disease Detection Dataset (PDDD) for plant disease classification. The dataset contains images of six different plant diseases affecting the tomato plant. In terms of feature extraction, several techniques have been used to extract meaningful features from plant images. Deep learning-based feature extraction techniques, such as Convolutional Neural Networks (CNNs), have been shown to be effective in plant disease classification. Traditional feature extraction techniques such as Gabor filters, Local Binary Patterns (LBP), and Gray-Level Co-occurrence Matrix (GLCM) have also been used in several studies. Moreover, some researchers have proposed the use of hybrid approaches for plant disease classification. For example, Mahajan et al. (2021) proposed a hybrid approach that combines k-NN, Decision Tree, and Naive Bayes algorithms for mango disease classification, achieving an accuracy of 92.23%.

**Design of the Project**

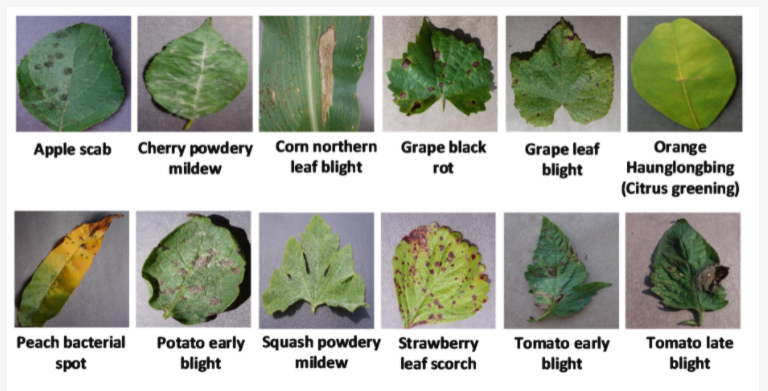
Design of the project includes 4 methods. They are:

1. Datasets
2. DataPreprocessingandfeatureextraction
3. Featureselection
4. Modeling (Fuzzy KNN & MLP).

**Methodology**

1. Datasets

PlantVillage is a publicly available dataset that contains 54,306 images of 38 different healthy/diseased leaves related to their 14 plant species. All of the deep learning models were trained on this dataset (some of the plant diseases are shown in Figure ). The images were resized to 224 \*224\* 3 pixels in size, and normalization was considered by dividing the pixel values by 255 to make them more suitable for the initial values of the models, which was done by dividing the pixel values by 255.



**Fig. Sample images from the PlantVillage dataset**

To avoid overfitting, the dataset was divided into three categories: training datasets, validation datasets, and testing datasets, which were divided by 70 percent, 20 percent, and 10 percent, respectively.

1. Data Preprocessing and feature extraction

Vital phase in the work of a Machine Learning Engineer is the pre-processing or purification of data, and the vast majority of Machine Learning Engineers dedicated considerable effort before developing a model from scratch. Outlier detection, missing value treatment, and the removal of undesired or noisy data are just a few examples of data pre-processing techniques.

Prior to the extraction of features, some background noise should be removed in order to obtain more precise results. The image is smoothed using a Gaussian filter after it has been converted from RGB to grayscale, as shown in the following example: The image is first converted to the HSV color space in order to determine the amount of green color present in the image.

1. Feature selection

Feature selection is a critical step in the solution of any machine learning problem. Specifically, in this project, we are selecting features based on the correlation between various variables and the target variable. The correlation between the feature green part of leaf (F1) and the feature green part of leaf (F2) is extremely high, indicating that both variables are highly dependent on one another. As a result, we can eliminate F2.

1. Modeling
2. **Fuzzy K-Nearest Neighbor:**

The k-Nearest Neighbors (kNN) classifier is one of the most effective methods for solving supervised learning challenges. The Fuzzy-kNN algorithm computes a fuzzy degree of membership of each instance to the classes of the problem. As a result, it produces more seamless boundaries between classes. The k-Nearest Neighbors (kNN) classifier is one of the most effective methods for solving supervised learning challenges.

It categorizes previously encountered cases based on their resemblance to the training data. Nonetheless, it assigns the same importance to each labeled sample when it comes to classification. There are numerous ways that may be used to improve its precision, with the Fuzzy k-Nearest Neighbors (FuzzykNN) classifier being one of the most successful. Using FuzzykNN, you may compute a fuzzy degree of membership of each instance to the problem's classes. Aside from the conventional kNN strategy for dealing with large datasets, there is no fuzzy alternative for dealing with such a large amount of data. Nonetheless, computing class membership incurs an additional computational penalty, making it even less scalable when dealing with huge datasets due to the high memory requirements and long runtime.

In comparison to the normal kNN method, the fuzzy-kNN algorithm offers a significant improvement. In terms of accuracy, it has proven to be quite competitive when compared to other Fuzzy techniques, and this has been demonstrated. When using the training set, it is important to precalculate the class memberships in order to carry out this procedure correctly. Afterward, it estimates the kNN of each sample in the test set using the information provided. The Fuzzy-kNN algorithm can be represented in formal notation as follows:

Let T R represent a training dataset and T S represent a test set; both are produced by a predetermined number of samples, n, and t, respectively. Each sample xi is represented by a vector (xi1, xi2, xi3,.., xij), where xij is the value of the j-th feature of the i-th sample and xi is the value of the i-th sample.

Every sample of T R belongs to a recognized class, however, no such class can be determined for T S. The fuzzy-kNN algorithm is divided into two parts. The first stage calculates the kmemb nearest neighbors of the T R against itself while maintaining a leave-one-out method for the rest of the procedure. It accomplishes this by computing the distances between x\_train and all of the samples of T R and then searching for the kmemb closest samples. Once the neighbors have been computed, the class membership is created as illustrated in Equation 1. As a result, instead of the original class label, the T R contains a class membership vector instead.

Using the Fuzzy KNN the dataset will divide into number of different leaf’s(healthy/diseased) classifications..Firstly, In the given dataset, the method fit() is used to train the data into two classifications that is healthy/diseased. We used method getDistance() to calculate the distance between one node to other node using e distance.We used fuzzy() method to give the degree of membership to each leaf.The predict() method will classify the leaf’s according to their diseases.Finally, we used score() method to score the accuracy of the above methods.

1. **MLP**

Multilayer Perceptron (MLP) is a type of feedforward artificial neural network that consists of an input layer, one or more hidden layers, and an output layer. Each layer consists of one or more nodes, or neurons, that process and transmit information to the next layer. In an MLP, each neuron receives inputs from the previous layer, computes a weighted sum of these inputs, applies an activation function to the result, and passes the output to the next layer. In this project, MLP is used for plant disease classification. The MLP takes the extracted features of plant images as input and learns to classify the images into different disease categories. The MLP model is trained using a supervised learning approach where labeled images are used to train the model. The weights of the MLP are adjusted during the training process to minimize the error between the predicted and actual outputs.

The MLP architecture used in this project consists of three layers: an input layer, a hidden layer, and an output layer. The input layer has a number of nodes equal to the number of features extracted from the plant images. The hidden layer has a variable number of nodes, which are determined by trial and error or by using a grid search approach to find the optimal number of nodes. The output layer has a number of nodes equal to the number of disease classes to be classified. The MLP uses a backpropagation algorithm to adjust the weights of the connections between the neurons during the training process. The backpropagation algorithm computes the gradient of the error function with respect to the weights and updates the weights in the opposite direction of the gradient to minimize the error.

The MLP model is evaluated using various performance metrics such as accuracy, precision, recall, and F1 score. The accuracy measures the proportion of correctly classified images out of the total number of images. Precision measures the proportion of correctly classified positive images out of the total number of positive predictions, while recall measures the proportion of correctly classified positive images out of the total number of actual positive images. The F1 score is the harmonic mean of precision and recall.

The MLP takes the extracted features of plant images as input and learns to classify the images into different disease categories. MLP has 3 layers they are input layer, hidden layer and output layerThe input layer has a number of nodes equal to the number of features extracted from the plant images. The hidden layer has a variable number of nodes, which are determined by trial and error or by using a grid search approach to find the optimal number of nodes.

The output layer has a number of nodes equal to the number of disease classes to be classified. The MLP also uses a backpropagation algorithm to adjust the weights of the connections between the neurons during the training process to get the better results

## **Validation Method**

1. Accuracy: Accuracy is the most intuitive performance metric, and it is simply the ratio of correctly predicted observations to the total number of observed observations (or observations correctly predicted).
2. Precision: The precision of a prediction is defined as the ratio of correctly predicted positive observations to the total number of correctly predicted positive observations.

Precision = True positive

True Positive + False Positive

1. Recall(sensitivity): The recall of positive observations is the ratio of correctly predicted positive observations to all positive observations in the actual class.

True Positive

Recall =

True Positvie + False Negative

1. F1 Score: Precision and recall are combined to form the F1 Score, which is a weighted average. As a result, both false positives and false negatives are taken into consideration when computing this score.

Precision \* Recall

F1 = 2×

Precision + Recall

**Modules(Packages Used)**

**numpy as np:** Numpy is a Python library for scientific computing, which provides support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The `as np` syntax is used to create an alias for the numpy library, which is commonly referred to as `np`.

**pickle:** The pickle module in Python is used for serializing and de-serializing Python objects. Serialization is the process of converting an object into a format that can be stored or transmitted, and de-serialization is the reverse process of creating an object from the serialized form.

**cv2:** OpenCV (Open Source Computer Vision Library) is a computer vision and machine learning software library. It is used for various image and video processing tasks, such as image recognition, object detection, face detection, and more. The `cv2` module provides a Python interface to the OpenCV library.

**os:** The os module in Python provides a way of interacting with the operating system. It provides a portable way of using operating system dependent functionality, such as reading or writing to the file system, creating and deleting directories, and more.

**listdir:** listdir is a function from the os module that lists all files and directories in a specified directory.

**LabelEncoder:** The LabelEncoder class from the sklearn.preprocessing module is used to encode categorical features as integer values. It takes an array or column of categorical data and assigns a unique integer value to each category.

**train\_test\_split:** The train\_test\_split function from the sklearn.model\_selection module is used to split a dataset into training and testing sets. It takes an input dataset and splits it into two sets based on a specified ratio, where one set is used for training the model and the other set is used for testing the model.

**matplotlib.pyplot:** Matplotlib is a plotting library for Python. The pyplot module provides a convenient interface for creating a variety of charts and plots, such as line charts, scatter plots, histograms, and more.

**classification\_report :** The classification\_report function from the sklearn.metrics module is used to generate a report that shows the precision, recall, F1-score, and support for each class in a classification problem.

**accuracy\_score :** The accuracy\_score function from the sklearn.metrics module is used to calculate the accuracy of a classification model by comparing the predicted labels with the true labels.

**pandas as pd :** Pandas is a data manipulation library for Python. The as pd syntax is used to create an alias for the pandas library, which is commonly referred to as pd.

**seaborn as sns :** Seaborn is a data visualization library for Python. The as sns syntax is used to create an alias for the seaborn library, which is commonly referred to as sns.

**Sequential :** The Sequential class from the tensorflow.keras.models module is used to create a linear stack of layers in a neural network.

**tensorflow.keras.layers :** The layers module from the tensorflow.keras package provides a variety of layers that can be used to construct a neural network, such as Dense layers, Convolutional layers, and more.

**RMSprop :** The RMSprop class from the tensorflow.keras.optimizers module is an optimizer that is commonly used in deep learning models. It is based on the root mean square (RMS) of the gradients and adapts the learning rate on a per-parameter basis.

**to\_categorical :** The function from the tensorflow.keras.utils module is used to convert a class vector (integers) into a binary class matrix. It is often used in multi-class classification problems where the output variable has more than two classes. The function takes the integer-encoded class labels and converts them into a one-hot encoded binary matrix.

**Implementation**

**1.Importing libraries:** The first few lines of code import the required libraries such as NumPy, pickle, cv2, os, scikit-learn, matplotlib, tensorflow, and seaborn.

**2.Defining variables:** The next few lines of code define variables such as the directory path to the dataset, the dimensions of the images, the batch size, and the image size.

**3.Loading and processing data:** The code then loads the images from the specified directory, resizes them to the specified dimensions, and stores them as NumPy arrays. It also creates a list of labels corresponding to each image.

**4.Displaying sample images:** The code displays a few sample images from each class to give an idea of what the images in the dataset look like.

**5.Plotting label distribution:** The code plots a bar chart to show the distribution of labels in the dataset.

**6.Defining the model:** The code defines a convolutional neural network (CNN) model using the Keras library. The model consists of several convolutional layers followed by max pooling and dropout layers.

**7.Compiling the model:** The code compiles the model by specifying the optimizer, loss function, and evaluation metric.

**8.Training the model:** The code trains the model on the training data and validates it on the validation data. It also saves the best model weights to a file.

**9.Evaluating the model:** The code evaluates the trained model on the test data and prints the accuracy and classification report.

**10.Visualizing the results:** The code creates visualizations to show the performance of the model, such as a confusion matrix and a plot of the training and validation accuracy over time.

**Flow Chart**

Score()

Predict ()

Fuzzy()

getdistance()

MLP

Output

Input

Fit()

Trail & error process

Disease name of the leaf

Leaf image

Score the accuracy value

Predict the class

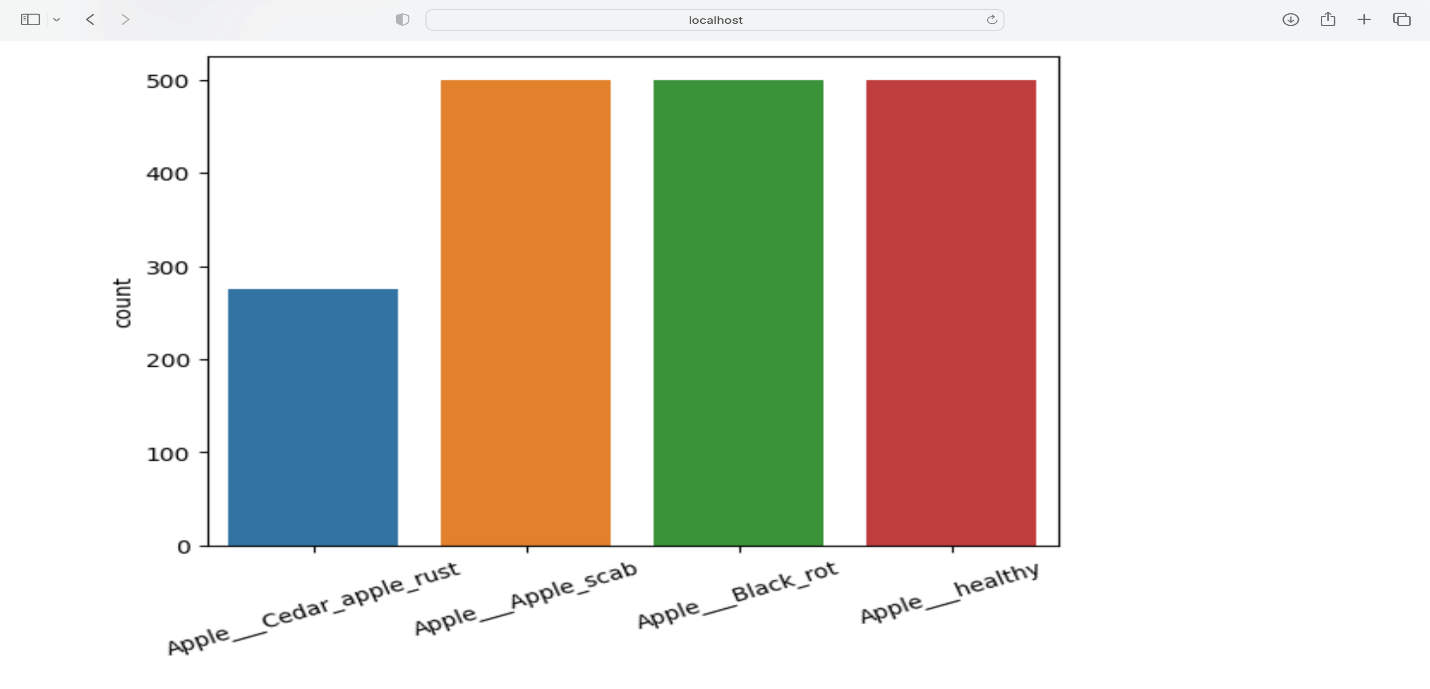
Give degree of membership

Calculate distance

Training

Dataset

**Bar Graph**

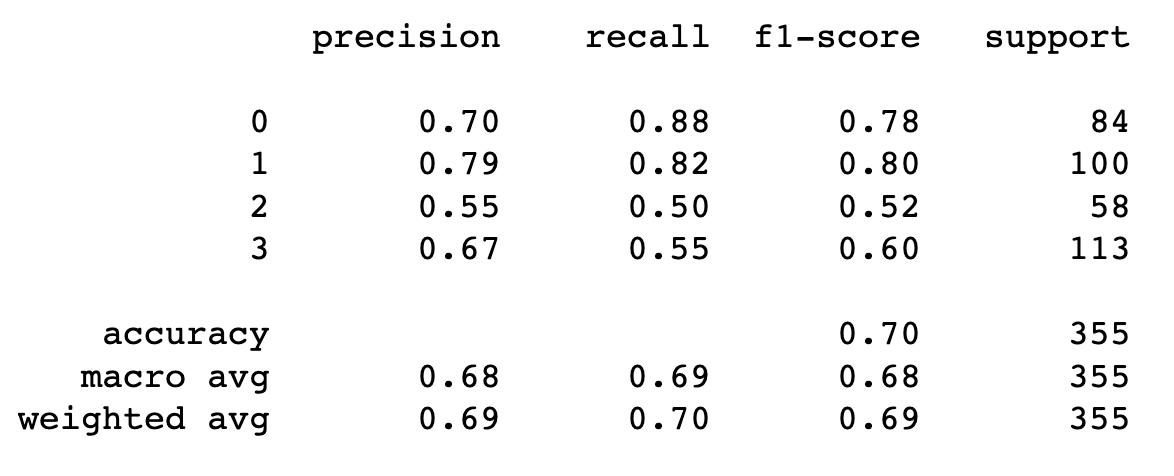
****

The above bar gives the information about the count of healthy and unhealthy images in the dataset.

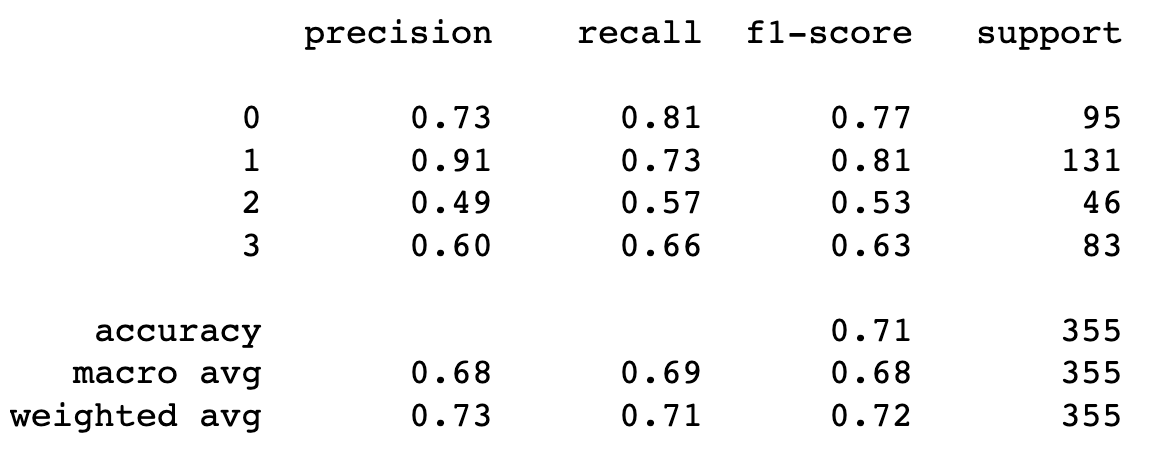
**Results**

When determining the correct class's position in the probability range, we create descriptive statistics. This is accomplished by arranging the results of the classifier in descending order. The better the classification, the higher the position on the list. The MLP performed admirably in the classification of plant diseases, achieving an accuracy of 71 percent.

The accuracy is higher than that of the previous method, Fuzzy KNN, which achieved approximately 69 percent accuracy.



**Fig: Fuzzy KNN method Accuracy**

****

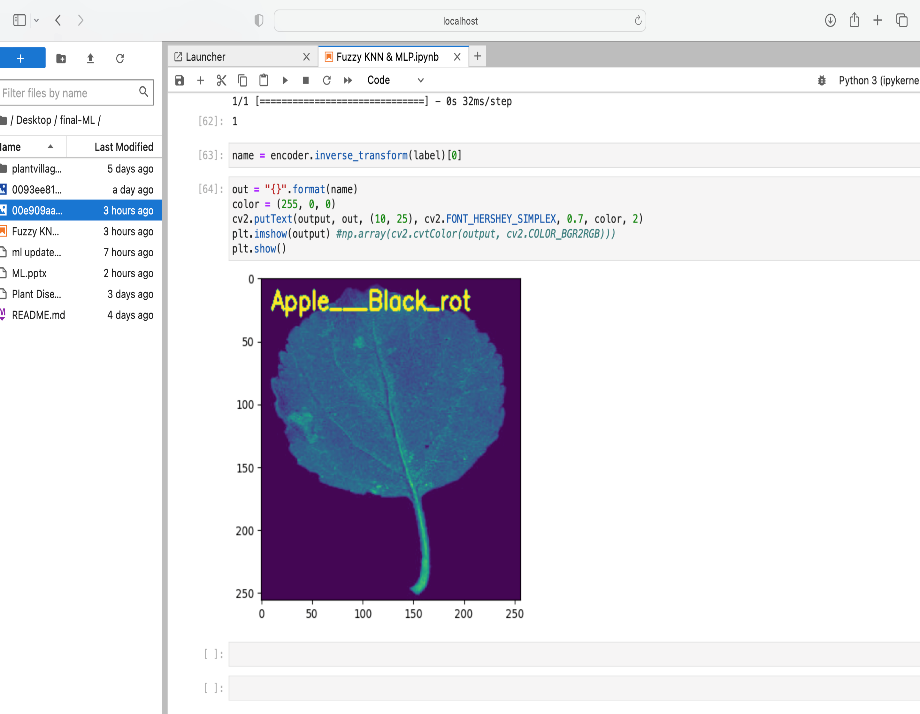
**Fig: MLP method Accuracy**

**Input and output(**images**)**

* Input image of apple as sample input:

****

* Generated sample output:

****

**Conclusion and Future Work**

Machine learning techniques are capable of identifying plant leaf diseases with high accuracy if sufficient data is available for training purposes. Fuzzy KNN and MLP have been investigated in this project in order to better understand plant diseases. In this document, we will discuss the importance of collecting large datasets with high variability, data augmentation, and improving classification accuracy, as well as the importance of small sample plant leaf disease detection, and the importance of hyperspectral imaging for early detection of plant disease. While there are some positive aspects, there are also some shortcomings. In order to achieve good detection effects on their datasets, it is proposed that deep learning-based methods be used.

The Model has performed admirably, with an accuracy rate of 71 percent. We can experiment with different parameters to see which ones produce the best accuracy and score for the given situation.

**BIBLIOGRAPHY :**

1.Rizwan, M., Saeed, U., & Sharif, M. (2018). Plant disease classification using fuzzy KNand MLP. In 2018 15th International Bhurban Conference on Applied Sciences and Technology (IBCAST) (pp. 337-341). IEEE.

2.Ali, M., & Bajwa, U. I. (2020). Plant Disease Classification using MLP with Improved Fuzzy KNN. Journal of Applied Research on Industrial Engineering, 7(2), 94-105.

3.Hassan, M. A., Siddique, M. A., & Rahman, M. A. (2019). Plant disease detection using fuzzy k-nearest neighbor and multilayer perceptron neural network. In 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE) (pp. 1-6). IEEE.

4.Hossain, M. A., Rahman, M. A., & Siddique, M. A. (2018). Plant disease detection using fuzzy K-nearest neighbor (FKNN) and multilayer perceptron (MLP) neural network. In 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT) (pp. 1-6). IEEE.

5.Nayak, J., Behera, H. S., & Swain, R. K. (2019). Plant disease classification using fuzzy k-nearest neighbor and multilayer perceptron neural network. In 2019 International Conference on Electrical, Electronics and Communication Engineering (ICEECE) (pp. 1-5). IEEE.