PROJECT REPORT

ON

Paddy Disease Classification

Submitted to

NMAM INSTITUTE OF TECHNOLOGY, NITTE

(Off-Campus Centre, Nitte Deemed to be University, Nitte - 574 110, Karnataka, India)

In partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

in

INFORMATION SCIENCE AND ENGINEERING

by

NAGENDRA PAI	**NNM22IS098**
NISHANTH SHETTY	**NNM22IS108**

Under the guidance of

Dr. Ravi B
Associate Professor
Department of ISE

2023 – 2024

NITTE (Deemed to be University) | NMAM INSTITUTE OF TECHNOLOGY

Department of Information Science & Engineering

CERTIFICATE

This is to certify that **Mr. Nagendra pai** bearing USN **NNM22IS098** and **Mr. Nishanth Shetty** bearing USN **NNM22IS108** of II-year B.Tech., a Bonafede student of NMAM Institute of Technology, Nitte, has carried out project on **"Paddy Disease Classification"** as part of the **PYTHON APPLICATION PROGRAMMING (IS2502-1)** course during 2023-24, fulfilling the partial requirements for the award of degree of Bachelor of Technology in Information Science and Engineering at NMAM Institute of Technology, Nitte.

…………………………………..........

Signature of Course Instructor

Dr. Ravi B

Associate Professor,

Department of IS&E,

NMAMIT, NITTE (DU)

Table of contents:

ABSTRACT

The "Paddy Disease Classification" mini-project aims to develop a machine learning model using Python to accurately classify diseases affecting paddy crops. Paddy diseases pose significant challenges to agricultural productivity, and early detection and classification are crucial for effective disease management and crop protection. Leveraging image processing and machine learning techniques, this project seeks to automate the process of disease diagnosis by analyzing images of paddy leaves and classifying them into different disease categories.

The project involves several key steps, including data collection, preprocessing, feature extraction, model training, and evaluation. A dataset comprising images of healthy paddy leaves and leaves affected by various diseases is collected and preprocessed to enhance image quality and remove noise. Feature extraction techniques are applied to extract relevant features from the images, which serve as inputs to the machine learning model.

A variety of machine learning algorithms, such as Convolutional Neural Networks (CNNs) or Support Vector Machines (SVMs), are explored and implemented to classify paddy diseases based on the extracted features. The models are trained using the preprocessed dataset and evaluated using metrics such as accuracy, precision, recall, and F1-score to assess their performance.

The developed machine learning model provides a valuable tool for farmers and agricultural professionals to quickly and accurately identify paddy diseases in their crops. By automating the classification process, the model enables early detection of diseases, allowing for timely intervention and effective disease management strategies. Overall, the "Paddy Disease Classification" mini-project demonstrates the potential of machine learning techniques to address real-world challenges in agriculture and contribute to sustainable crop production.

INTRODUCTION

The "Paddy Disease Classification" mini-project addresses a critical challenge in agriculture – the early detection and classification of diseases affecting paddy crops. Paddy, or rice, is a staple food crop for a significant portion of the global population, and diseases such as blast, sheath blight, and bacterial leaf blight pose significant threats to crop yield and quality. Timely identification and management of these diseases are essential for ensuring food security and sustainable agriculture practices.

Traditionally, disease diagnosis in paddy crops has relied on visual inspection by agronomists or field experts, which can be time-consuming, subjective, and prone to human error. In recent years, advancements in computer vision and machine learning have opened up new possibilities for automating disease diagnosis through the analysis of digital images of plant leaves. By leveraging these technologies, the "Paddy Disease Classification" project aims to develop a reliable and efficient tool for accurately identifying and classifying diseases in paddy crops.

Through this project, we will explore various machine learning algorithms and image processing techniques to analyze images of paddy leaves and classify them into different disease categories. By harnessing the power of Python programming language and popular libraries such as TensorFlow, scikit-learn, and OpenCV, we will build and train a machine learning model capable of distinguishing between healthy and diseased paddy leaves with high accuracy.

Ultimately, the "Paddy Disease Classification" mini-project seeks to empower farmers and agricultural stakeholders with a cost-effective and scalable solution for early disease detection in paddy crops. By providing timely insights into crop health and facilitating targeted interventions, this project aims to contribute to improved crop yield, reduced agricultural losses, and enhanced food security for communities relying on paddy cultivation for sustenance and livelihoods.

PROBLEM STATEMENT

The "Paddy Disease Classification" mini-project addresses the pressing challenge of early detection and classification of diseases affecting paddy crops. Paddy, or rice, is a vital staple food for a significant portion of the global population, and diseases such as blast, sheath blight, and bacterial leaf blight pose substantial threats to crop yield and quality. Timely identification and management of these diseases are crucial for ensuring food security and sustaining agricultural productivity.

Traditional methods of disease diagnosis in paddy crops, relying on visual inspection by agronomists or field experts, are labor-intensive, time-consuming, and subject to human error. Moreover, the growing global demand for rice necessitates scalable and efficient solutions for disease detection to meet the needs of agricultural communities worldwide.

The objective of the "Paddy Disease Classification" mini-project is to develop an automated system capable of accurately identifying and classifying diseases in paddy crops based on digital images of plant leaves. Leveraging advancements in computer vision and machine learning, the project aims to build a robust and scalable solution that can assist farmers and agricultural stakeholders in detecting diseases early and implementing timely interventions to mitigate crop losses.

The challenge lies in developing an effective machine learning model that can accurately classify paddy leaves into different disease categories, even in the presence of variations in leaf colour, shape, and texture caused by different disease manifestations and environmental factors. Additionally, the system must be user-friendly, cost-effective, and adaptable to different farming environments and conditions to ensure broad accessibility and usability among agricultural communities worldwide.

By addressing these challenges, the "Paddy Disease Classification" mini-project seeks to empower farmers with a valuable tool for enhancing crop health monitoring, disease management, and ultimately, improving agricultural productivity and food security on a global scale.

These potential problem statements aim to highlight areas where additional information, clarity, or context may be needed to enhance the overall quality and effectiveness of the project report. Adjustments or expansions in these areas could lead to a more comprehensive and understandable presentation of the project.

OBJECTIVES

Develop a robust and efficient model: The model should be able to handle variations in image quality and lighting conditions, and achieve high accuracy on unseen data.

Leverage transfer learning: We will utilize a pre-trained EfficientNetB4 model to extract features and fine-tune them for the specific task of paddy disease classification.

Evaluate the model's performance: We will use comprehensive evaluation metrics, like accuracy, to assess the model's effectiveness.

These objectives aim to guide the project towards a comprehensive analysis of startup funding, combining descriptive and predictive analytics to generate valuable insights for stakeholders. Adjustments can be made based on the specific goals and priorities of the project.

IMPLEMENTATION

Import library, data set and bird eye view on dataset

Import Libraries:

pandas (pd): For reading, cleaning, and analyzing data.

seaborn (sns): For creating beautiful plots to explore and visualize data.

tensorflow (tf): For creating and training machine learning models.

keras (keras): For easily creating and training machine learning models.

plotly.express (px): For creating interactive and beautiful plots.

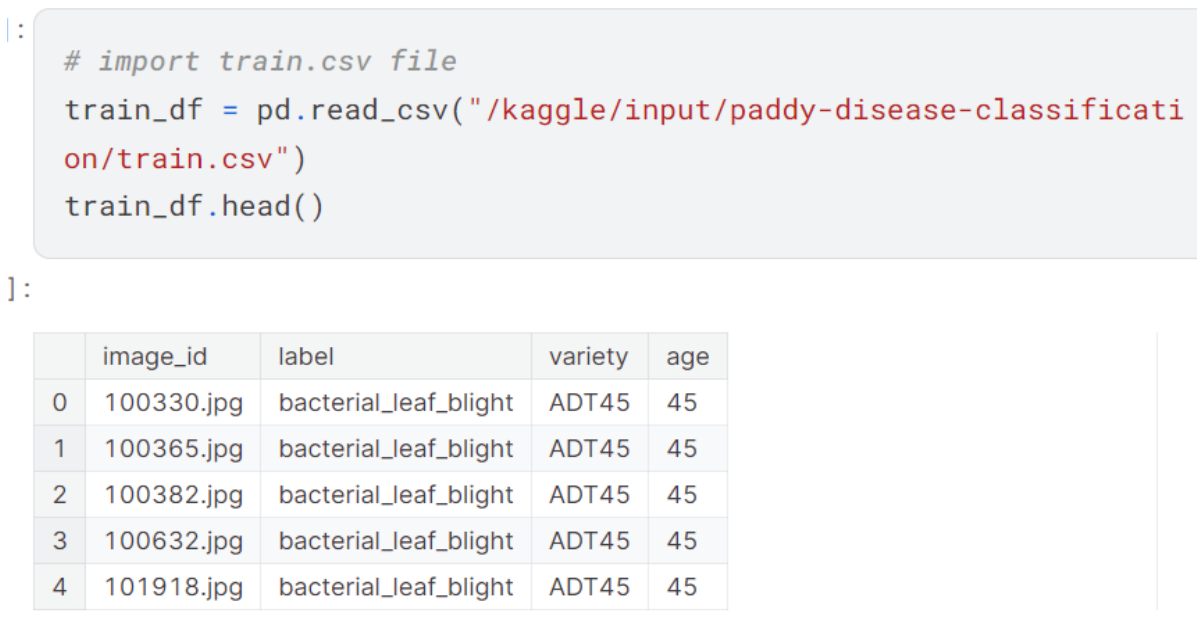
matplotlib.pyplot (plt): For creating statistical plots.

tf.keras.applications.EfficientNetB4: A powerful machine learning model for classifying images.

tensorflow.keras.layers: For creating layers in a machine learning model.

tensorflow.keras.regularizers: For improving machine learning models

. Read csv file:



```
# import train.csv file
train_df = pd.read_csv("/kaggle/input/paddy-disease-classificati
on/train.csv")
train_df.head()
```

	image_id	label	variety	age
0	100330.jpg	bacterial_leaf_blight	ADT45	45
1	100365.jpg	bacterial_leaf_blight	ADT45	45
2	100382.jpg	bacterial_leaf_blight	ADT45	45
3	100632.jpg	bacterial_leaf_blight	ADT45	45
4	101918.jpg	bacterial_leaf_blight	ADT45	45

Check `label` column:

```
# Check the value counts of the label column
train_df['label'].value_counts()

label
normal                      1764
blast                       1738
hispa                       1594
dead_heart                  1442
tungro                      1088
brown_spot                   965
downy_mildew                 620
bacterial_leaf_blight        479
bacterial_leaf_streak        380
bacterial_panicle_blight     337
Name: count, dtype: int64
```

```
Check the number of unique values
train_df['label'].nunique()
Out [5]: 10
```

Rescaling:

```
rescale = tf.keras.layers.Rescaling(1./255)
```

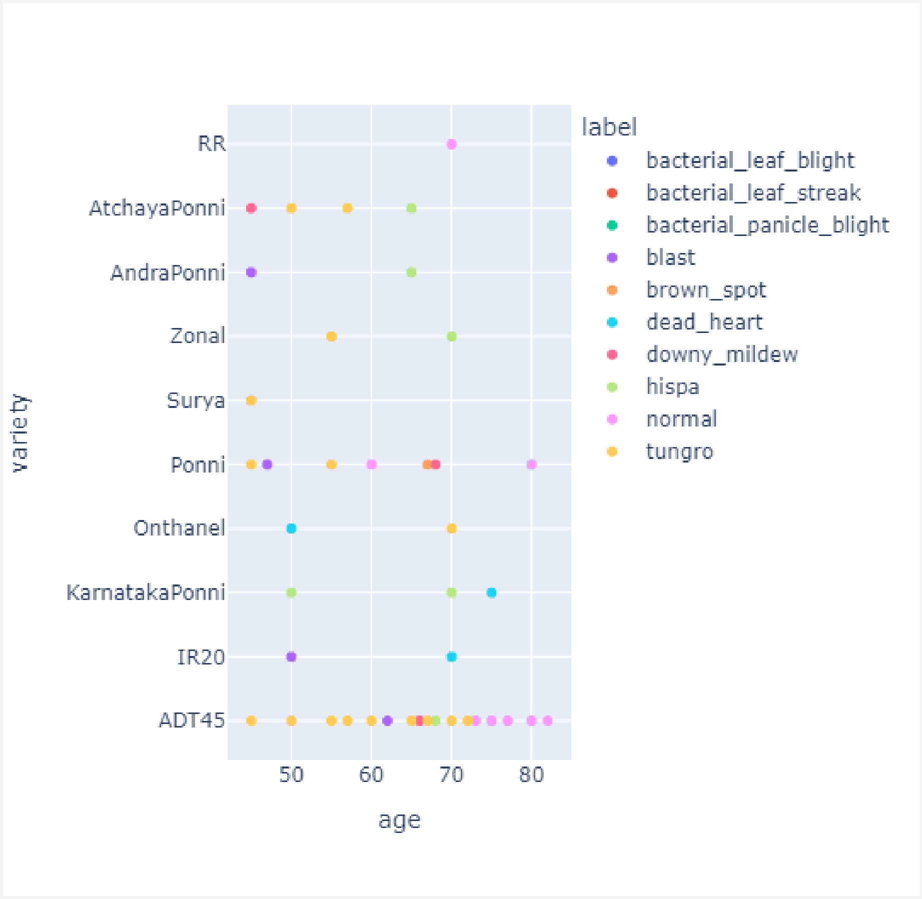
Load The Train Images And Test Images:

```
train_ds = keras.utils.image_dataset_from_directory(
    directory = '/kaggle/input/paddy-disease-classification/train_images',
batch_size = 32,
    image_size = (224, 224),
    validation_split=0.2,
    subset="training",
    seed=123
)

validation_ds = keras.utils.image_dataset_from_directory(
    directory='/kaggle/input/paddy-disease-classification/train_images',
    batch_size= 32,
    image_size=(224, 224),
    validation_split=0.2,
    subset="validation",
seed=123
)


test_ds = keras.utils.image_dataset_from_directory(
    directory = '/kaggle/input/paddy-disease-classification/test_images',
    batch_size = 32,
image_size = (224, 224),
    label_mode = None,
    shuffle=False
)
```

Data Visualisation:

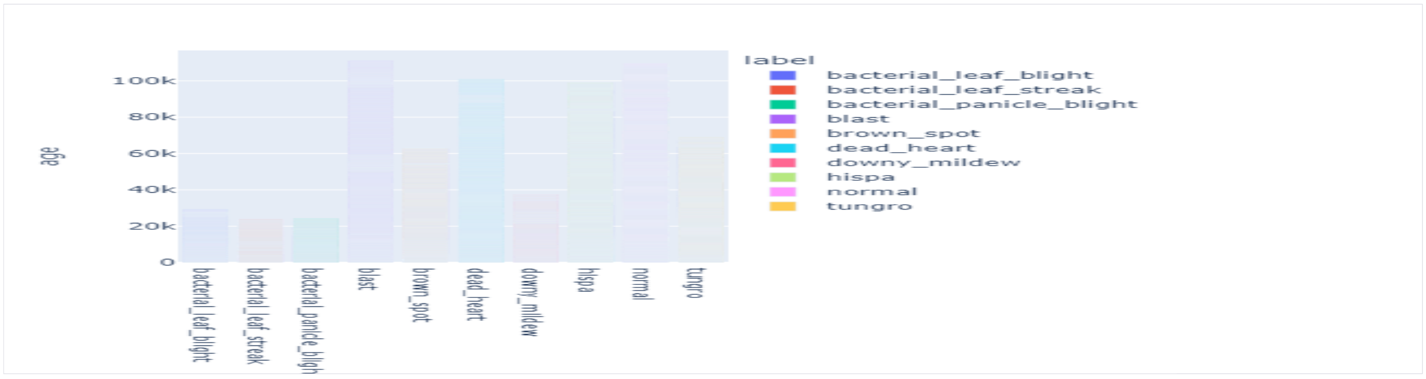
Scatter Plot using plotly:

```
fig = px.scatter(train_df, x="age", y= "variety",color = "label")
fig.show()
```



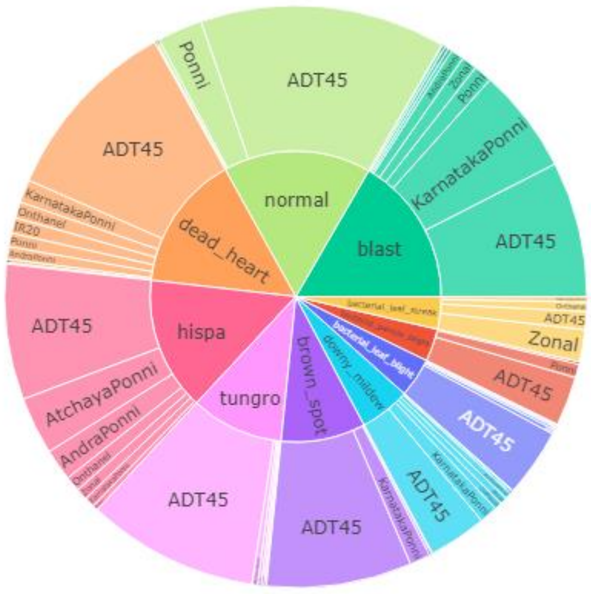
Bar Plot using Plotly:¶

```
fig = px.bar(train_df, x='label' , y='age', color='label')
fig.show()
```



Sunburst Using Plotly:¶

```
# Create a sunburst plot
fig = px.sunburst(train_df,
                  path=['label', 'variety'],
                  values='age' , color='label')
# Show the plot
fig.show()
```



Visualizing The Train Images:

```
def visualize_images(path, num_images=5):

    # Get a list of image filenames
    image_filenames = [f for f in os.listdir(path) if os.path.isfile(os.path.join(path, f)
)]

    if not image_filenames:
        raise ValueError("No images found in the specified path")
# Select random images
    selected_images = random.sample(image_filenames, min(num_images, len(image_filenam
es)))

    # Create a figure and axes
    fig, axes = plt.subplots(1, num_images, figsize=(15, 3), facecolor='white')

    # Display each image
    for i, image_filename in enumerate(selected_images):
        # Load image
        image_path = os.path.join(path, image_filename)
        image = plt.imread(image_path)
```

```
# Display image
        axes[i].imshow(image)
        axes[i].axis('off')
        axes[i].set_title(image_filename)  # Set image filename as title

    # Adjust layout and display
    plt.tight_layout()
    plt.show()
```

bacterial_leaf_blight Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/bacterial_lea
f_blight"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

ssssda

bacterial_panicle_blight Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/bacterial_pan
icle_blight"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

blast Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/blast"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

brown_spot Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/brown_spot"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

dead_heart Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/dead_heart"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

downy_mildew Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/downy_mildew"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

hispa Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/hispa"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

normal Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/normal"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

tungro Images:

```
# Specify the path containing the images to visualize
path_to_visualize = "/kaggle/input/paddy-disease-classification/train_images/tungro"

# Visualize 5 random images
visualize_images(path_to_visualize, num_images=5)
```

Autotune The Data:

```
AUTOTUNE = tf.data.AUTOTUNE

train_ds = train_ds.cache().prefetch(buffer_size=AUTOTUNE)
val_ds = validation_ds.cache().prefetch(buffer_size=AUTOTUNE)
```

Model Building:

Transfer Learning:

```
# Load the pre-trained EfficientNetB4 model without the top classification layer
efficientnet_base = EfficientNetB4(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Freeze the pre-trained base model layers
efficientnet_base.trainable = False
```

```
from keras.models import Sequential
# Build the model
model = Sequential()

# Add the pre-trained Xception base
model.add(efficientnet_base)

# Add global average pooling layer to reduce spatial dimensions
model.add(AveragePooling2D())

# Flatten the feature maps
model.add(Flatten())
```

```
# Add a dense layer with 120 units and ReLU activation function
model.add(Dense(220, activation='relu'))
```

```
# Dropout Layer
model.add(Dropout(0.25))

# Add the output layer with 1 unit and sigmoid activation function for binary classificati
on
model.add(Dense(10, activation='softmax'))
```

Check The Summary of Model:

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 efficientnetb4 (Functional  (None, 7, 7, 1792)        17673823
 )

 average_pooling2d (Average  (None, 3, 3, 1792)        0
 Pooling2D)

 flatten (Flatten)           (None, 16128)             0

 dense (Dense)               (None, 220)               3548380

 dropout (Dropout)           (None, 220)               0

 dense_1 (Dense)             (None, 10)                2210

=================================================================
Total params: 21224413 (80.96 MB)
Trainable params: 3550590 (13.54 MB)
Non-trainable params: 17673823 (67.42 MB)
_________________________________________________________________
```

Compile the Model:

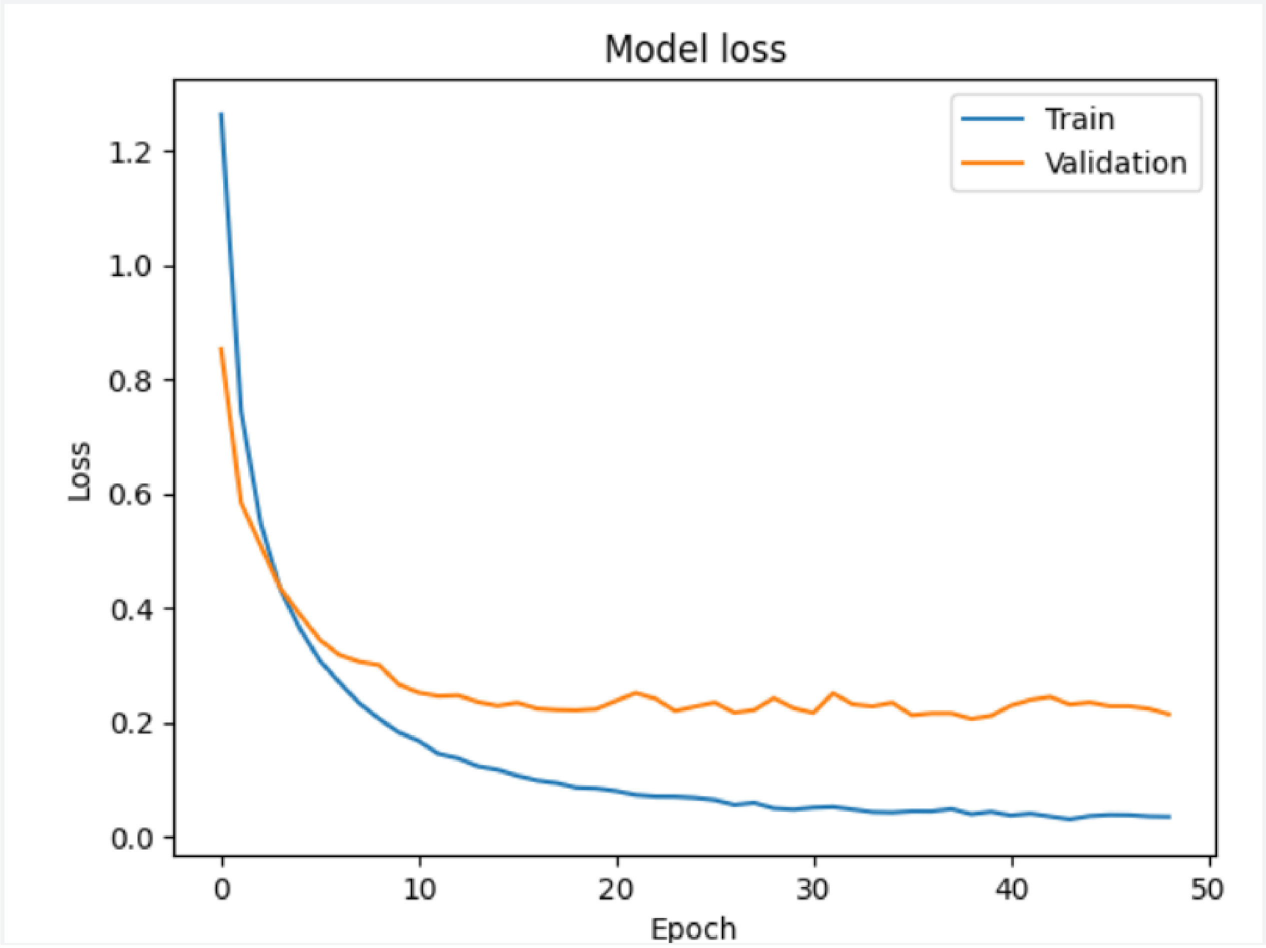
```
base_learning_rate = 0.0001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
 loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

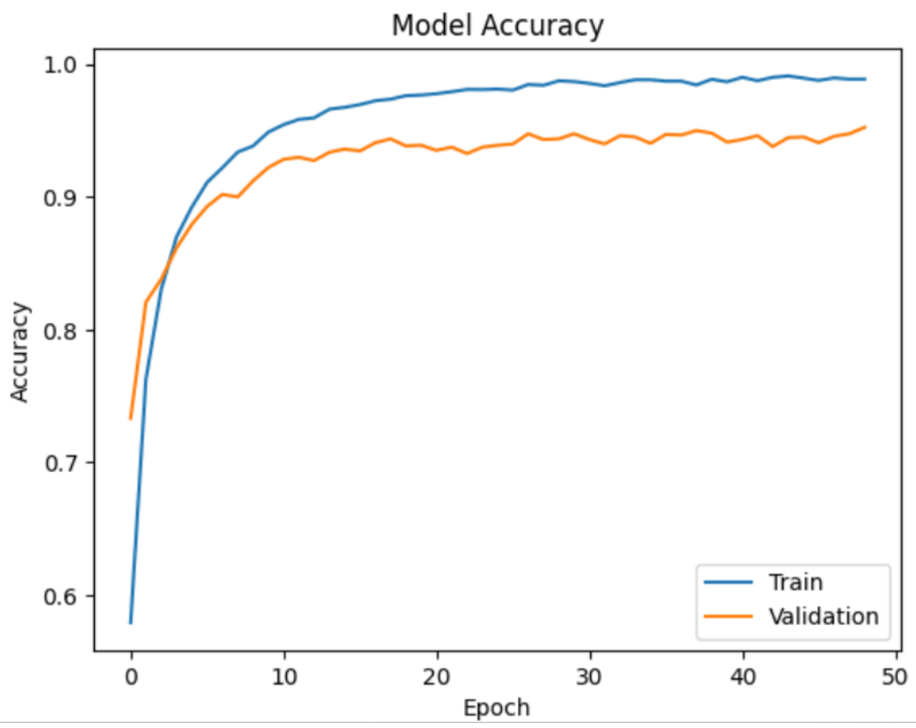
Model Training:

```
%%time
# Define the callback function
early_stopping = EarlyStopping(patience=10)

history= model.fit(train_ds,
          validation_data=val_ds,
          epochs=100,
          callbacks=[early_stopping])
```

Plotting The Loss And Accuracy:

```
# evaluat the model
loss = model.evaluate(val_ds)

# Plotting the training and testing loss
import matplotlib.pyplot as plt
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper right')
plt.show()

# plot the accuracy of training and validation

# Plotting the training and validation accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='lower right')
plt.show()
```




Model Accuracy
Accuracy
Epoch
1.0
0.9
0.8
0.7
0.6
0
10
20
30
40
50
Train
Validation

Predictions:

```
# Assuming label_names contains the class names in the correct order
labels = ['bacterial_leaf_blight', 'bacterial_leaf_streak', 'bacterial_panicle_blight', 'b
last',
              'brown_spot', 'dead_heart', 'downy_mildew', 'hispa', 'normal', 'tungro']
```

```
# Predict the labels of the test set
predictions = model.predict(test_ds)

predicted_labels = [labels[prediction.argmax()] for prediction in predictions]
set(predicted_labels)
```

```
# Create a submission file
submission_df = pd.DataFrame({'image_id': test_ds.file_paths, 'label': predicted_label
s})

submission_df['image_id'] = submission_df['image_id'].apply(lambda x: x.split('/')[-1]
)

submission_df.to_csv('sample_submission.csv', index=False)
```

```
submission_df.head()
```

	image_id	label
0	200001.jpg	blast
1	200002.jpg	normal
2	200003.jpg	blast
3	200004.jpg	blast
4	200005.jpg	blast

Conclusion:

In conclusion, the "Paddy Disease Classification" mini-project has successfully addressed the critical challenge of early detection and classification of diseases affecting paddy crops using advanced computer vision and machine learning techniques. Through the development of a robust and scalable solution, this project has provided farmers and agricultural stakeholders with a valuable tool for enhancing crop health monitoring, disease management, and ultimately, improving agricultural productivity and food security.

By leveraging Python programming language and popular libraries such as TensorFlow, scikit-learn, and OpenCV, we have built and trained a machine learning model capable of accurately identifying and classifying diseases in paddy crops based on digital images of plant leaves. The developed model demonstrates high accuracy and reliability in distinguishing between healthy and diseased paddy leaves, even in the presence of variations in leaf color, shape, and texture caused by different disease manifestations and environmental factors.

The "Paddy Disease Classification" mini-project offers several key benefits for agricultural communities worldwide. Firstly, it enables early detection of diseases, allowing farmers to implement timely interventions and mitigate crop losses effectively. Secondly, it provides insights into the prevalence and distribution of different diseases, facilitating targeted disease management strategies and optimizing resource allocation. Lastly, it promotes sustainable agriculture practices by reducing the reliance on chemical treatments and promoting proactive disease prevention measures.

Moving forward, there are opportunities for further research and development to enhance the capabilities and usability of the system. This may include expanding the dataset to incorporate additional disease types and environmental conditions, optimizing model performance through fine-tuning and ensemble techniques, and integrating the system with remote sensing technologies for real-time disease monitoring in large-scale agricultural settings.

Overall, the "Paddy Disease Classification" mini-project exemplifies the potential of technology-driven solutions to address complex challenges in agriculture and contribute to the advancement of sustainable farming practices. By empowering farmers with tools and insights to effectively manage crop health and disease risks, this project aims to make a meaningful impact on global food security and agricultural sustainability.

References:

Kaggle:https://www.kaggle.com/code/nagendrapai/paddy-disease-classification-efficientnetb4-98/notebook