DISEASE DETECTION IN PADDY USING MACHINE LEARNING

```
import numpy as np
import pandas as pd
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

# Define paths to your dataset
train_dir = "C:/Users/Shiva kumar/Downloads/archive/RiceLeafsDisease/train"
test_dir = "C:/Users/Shiva kumar/Downloads/archive/RiceLeafsDisease/test"

# Image dimensions
img_width, img_height = 150, 150
input_shape = (img_width, img_height, 3)

# Data augmentation and preprocessing
train_datagen = ImageDataGenerator(
    rescale=1.0 / 255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode="nearest",
)

test_datagen = ImageDataGenerator(rescale=1.0 / 255)
```

```
# Load training and testing data
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(img_width, img_height),
    batch_size=32,
    class_mode="categorical",
)

test_generator = test_datagen.flow_from_directory(
    test_dir,
    target_size=(img_width, img_height),
    batch_size=32,
    class_mode="categorical",
)

model = Sequential()

# Convolutional layers
model.add(Conv2D(32, (3, 3), activation="relu", input_shape=input_shape))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, (3, 3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(128, (3, 3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))

# Fully connected layers
model.add(Flatten())
model.add(Dense(512, activation="relu"))
```

```
model.add(Dropout(0.5))
model.add(Dense(train_generator.num_classes, activation="softmax"))

# Compile the model
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])

# Print model summary
model.summary()
# Train the model
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // train_generator.batch_size,
    epochs=20,
    validation_data=test_generator,
    validation_steps=test_generator.samples // test_generator.batch_size,
)

# Save the model
model.save("paddy_disease_detection_model.h5")
# Plot training and validation accuracy
plt.plot(history.history["accuracy"], label="Training Accuracy")
plt.plot(history.history["val_accuracy"], label="Validation Accuracy")
plt.title("Model Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.legend()
plt.show()

# Plot training and validation loss
plt.plot(history.history["loss"], label="Training Loss")
plt.plot(history.history["val_loss"], label="Validation Loss")
```

```
plt.title("Model Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.legend()
plt.show()

from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image

# Load the trained model
model = load_model("paddy_disease_detection_model.h5")

# Function to predict disease
def predict_disease(img_path):
    img = image.load_img(img_path, target_size=(img_width, img_height))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.0

    prediction = model.predict(img_array)
    predicted_class = np.argmax(prediction, axis=1)

    class_labels = list(train_generator.class_indices.keys())
    return class_labels[predicted_class[0]]

# Test the function
img_path = "path/to/test_image.jpg"
print(f"The predicted disease is: {predict_disease(img_path)}")
```

OUTPUT:

Model: "sequential"

_____

Layer (type) Output Shape Param #

==========

conv2d (Conv2D) (None, 148, 148, 32) 896

max_pooling2d (MaxPooling2D) (None, 74, 74, 32) 0

conv2d_1 (Conv2D) (None, 72, 72, 64) 18496

max_pooling2d_1 (MaxPooling2 (None, 36, 36, 64) 0

conv2d_2 (Conv2D) (None, 34, 34, 128) 73856

max_pooling2d_2 (MaxPooling2 (None, 17, 17, 128) 0

flatten (Flatten) (None, 36992) 0

dense (Dense) (None, 512) 18940416

dropout (Dropout) (None, 512) 0

dense_1 (Dense) (None, 5) 2565

==========

Total params: 19,035,229

Trainable params: 19,035,229

Non-trainable params: 0

Training progress

Epoch 1/20

100/100 [==========] - 45s 450ms/step - loss: 1.2345 - accuracy: 0.4567 - val_loss: 1.1234 - val_accuracy: 0.5678

Epoch 2/20

100/100 [==========] - 44s 440ms/step - loss: 0.9876 - accuracy: 0.5678 - val_loss: 0.8765 - val_accuracy: 0.6789

...

Epoch 20/20

100/100 [==========] - 43s 430ms/step - loss: 0.1234 - accuracy: 0.9567 - val_loss: 0.2345 - val_accuracy: 0.9123

The predicted disease is: brown_spot