Assignment No-1

Title-To develop a machine learning model to build a recommendation system for crop disease detection and yield prediction in agriculture.

Objectives-
- To understand the fundamentals of machine learning and its application in agriculture, particularly for crop disease detection and yield prediction.
- To implement a machine learning model that can analyze crop data and make predictions about disease presence and expected yield.
- To evaluate the effectiveness of the model and interpret its predictions in the context of agricultural practices.

Outcomes-Students will be able to
- Grasp key concepts and applications in agriculture, including data handling and preprocessing skills.
- create and select effective features to improve model performance.
- Experience in training models, assessing performance, and analyzing outputs for actionable recommendations.
- Clear documentation of processes and enhanced problem-solving skills through iterative development

Theory-

To develop a machine learning model to build a recommendation system for crop disease detection and yield prediction in agriculture, follow these steps:

1. data collection

you need two types of data:

- **images of crop diseases**: images of healthy and diseased crops for disease detection.
- **agricultural/environmental data**: data on weather conditions, soil properties, water usage, fertilizers, etc., for yield prediction.

2. data preprocessing

- **image data**: prepare images for disease detection using techniques like resizing, normalization, and data augmentation.
- **tabular data**: clean and preprocess agricultural data (handling missing values, scaling, and feature engineering).

3. machine learning models

- **cnn (convolutional neural networks)**: used for disease detection.
- **regression models (random forest, xgboost, etc.)**: used for yield prediction based on environmental data.

4. building cnn for disease detection

- use cnn for image classification to detect diseases in crop images.
- the cnn architecture typically includes convolutional layers, max pooling, flattening, and fully connected layers.

5. building regression model for yield prediction

- use regression models to predict crop yield based on environmental factors such as rainfall, temperature, soil quality, and fertilizer usage.

6. training and evaluation

- train the cnn model for disease detection using labeled image data.
- train the regression model for yield prediction using the preprocessed tabular data.

7. recommendation system

- based on the disease detection and yield prediction results, provide recommendations.
- if a disease is detected, recommend treatments or preventive measures.
- if the yield is predicted to be low, suggest changes in farming practices (fertilizers, irrigation, etc.).

8. deployment

- deploy the trained models as part of a web or mobile application to allow farmers to upload images and input data for real-time recommendations.

Sample code-

```
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from keras.preprocessing.image import img_to_array
import random

# Step 1: Simple CNN for Crop Disease Detection (Using random data for now)
# --------------------------------------
# Generate random image data for CNN training (100 images, size 64x64, 3 channels for RGB)
X_train_images = np.random.rand(100, 64, 64, 3)  # Random 100 RGB images
y_train_images = np.random.randint(2, size=100)  # Random binary labels (0: Healthy, 1:
Diseased)

# CNN Model for Disease Detection
cnn_model = Sequential()
cnn_model.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
cnn_model.add(MaxPooling2D(pool_size=(2, 2)))
cnn_model.add(Flatten())
cnn_model.add(Dense(128, activation='relu'))
cnn_model.add(Dense(1, activation='sigmoid'))  # Output layer for binary classification

# Compile the CNN model
cnn_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the CNN model (using mock data, real images should be used in practice)
cnn_model.fit(X_train_images, y_train_images, epochs=5, batch_size=10, verbose=1)

# Step 2: Random Forest for Yield Prediction (Using random environmental data)
# -------------------------------------

# Create mock environmental data (rainfall, temperature, soil_quality) and yield
X = np.random.rand(100, 3)  # Features: rainfall, temperature, soil_quality
y = np.random.rand(100) * 100  # Yield (arbitrary units, e.g., tons/hectare)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Random Forest Model for Yield Prediction
yield_model = RandomForestRegressor(n_estimators=100, random_state=42)
yield_model.fit(X_train, y_train)

# Step 3: Simple Recommendation System
# -------------------------------------

def recommend(disease_prediction, yield_prediction):
    if disease_prediction >= 0.5:
        return "Disease detected! Recommended action: Apply pesticide."
    elif yield_prediction < 50:
        return "Low yield predicted! Recommended action: Improve irrigation and soil quality."
    else:
        return "Crop is healthy and yield prediction is optimal."
```

```
# Step 4: Testing with Simulated Input Data
# --------------------------------------

# Simulate a new test image (Replace with actual image data in practice)
test_image = np.random.rand(1, 64, 64, 3)  # A new test image
disease_prediction = cnn_model.predict(test_image)[0][0]  # Predict if the crop is diseased

# Simulate new environmental data for yield prediction (rainfall, temperature, soil quality)
test_env_data = np.array([[0.8, 0.6, 0.7]])  # Simulated new data
yield_prediction = yield_model.predict(test_env_data)[0]  # Predict the yield

# Get recommendation based on predictions
recommendation = recommend(disease_prediction, yield_prediction)

# Step 5: Display Output
# --------------------------------------

print(f"Disease Prediction: {disease_prediction:.4f} (0: Healthy, 1: Diseased)")
print(f"Yield Prediction: {yield_prediction:.2f} units")
print(f"Recommendation: {recommendation}")
```

Output-

```
Disease Prediction: 0.3456 (0: Healthy, 1: Diseased)
Yield Prediction: 72.34 units
Recommendation: Crop is healthy and yield prediction is optimal.
```