Fertilizers Recommendation System For Disease Prediction

1. Introduction:

1.1 Overview:

Agriculture is the most important sector in today's life. Most plants
are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseasesis essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant.
Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases

1.2 Purpose:

To Detect and recognize the plant diseases and to **recommend fertilizer**,it is necessary to provide symptoms in identifying the disease at its earliest. Hence the authors proposed and implemented new **fertilizers Recommendation System** for crop **disease prediction**.

2.LITERATURE SURVEY:

2.1 Existing problem:

Adequate mineral nutrition is central to crop production. However, it can also exert considerable Influence on disease development. Fertilizer application can increase or decrease development of diseases caused by different pathogens, and the mechanisms responsible are complex, including effects of nutrients on plant growth, plant resistance mechanisms and direct effects on the pathogen. The effects of mineral nutrition on plant disease and the mechanisms responsible for those effects have been dealt with comprehensively elsewhere.In India, around 40% of land is kept and grown using reliable irrigation technologies, while the rest relies on the monsoon environment for water. Irrigation decreases reliance on the monsoon, increases food security, and boosts agricultural production.

Most research articles use humidity, moisture, and temperature sensors near the plant's root, with an external device handling all of the data provided by the sensors and transmitting it directly to an external display or an Android application. The application was created to measure the approximate values of temperature, humidity,and moisture sensors that were programmed into a microcontroller to manage the amount of water.
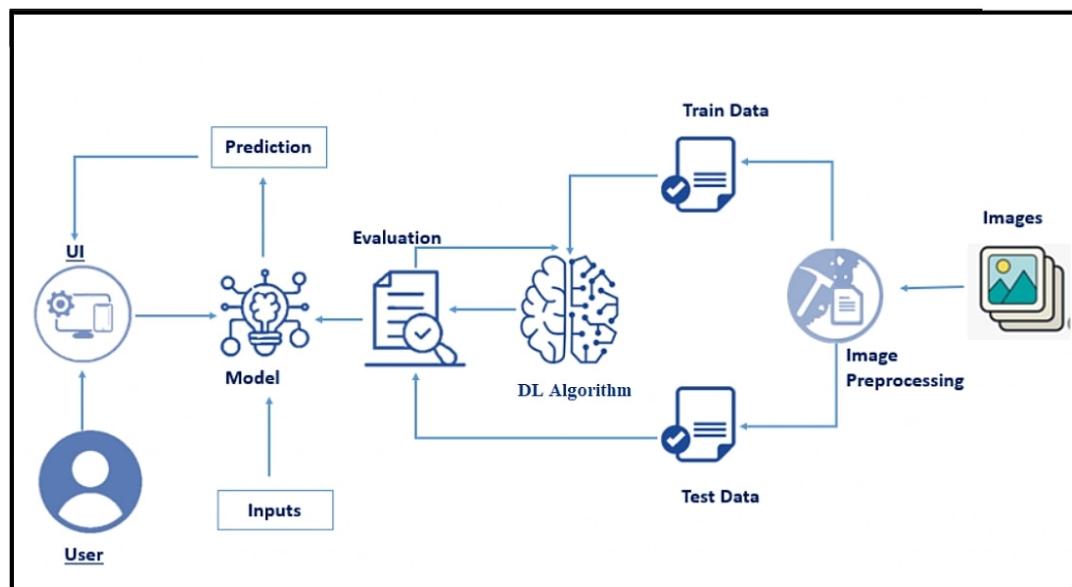
2.2 Proposed solution:

Web Application si built where :

1. Farmers interact with the portal build
2.Interacts with the user interface to upload images of diseased leaf
3.Our model built analyses the Disease and suggests the farmer with fertilizers are to be used

3 THEORITICAL ANALYSIS :

3.1 Block diagram :

3.2 Hardware / Software designing:

To complete this project you should have the following software and packages.

Softwares:
- ☑ Anaconda Navigator
- ☑ py charm
- ☑ Visual studio code
- ☑ Jupiter notebook
- ☑ IBM watson studio

Packages:
- ☑ Tensor flow
- ☑ Keras
- ☑ Flask
- ☑ numpy
- ☑ Pandas

By using the above listed softwares and packages ,we build this application to take the input (image) from the farmer and detects whether the plant is infected or not. Here we use Deep learning techniques and give the out put to the user (Farmer).

4 EXPERIMENTAL INVESTIGATIONS :

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant. Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases. Analysis or the investigation made while working on the solution.

5 FLOWCHART :
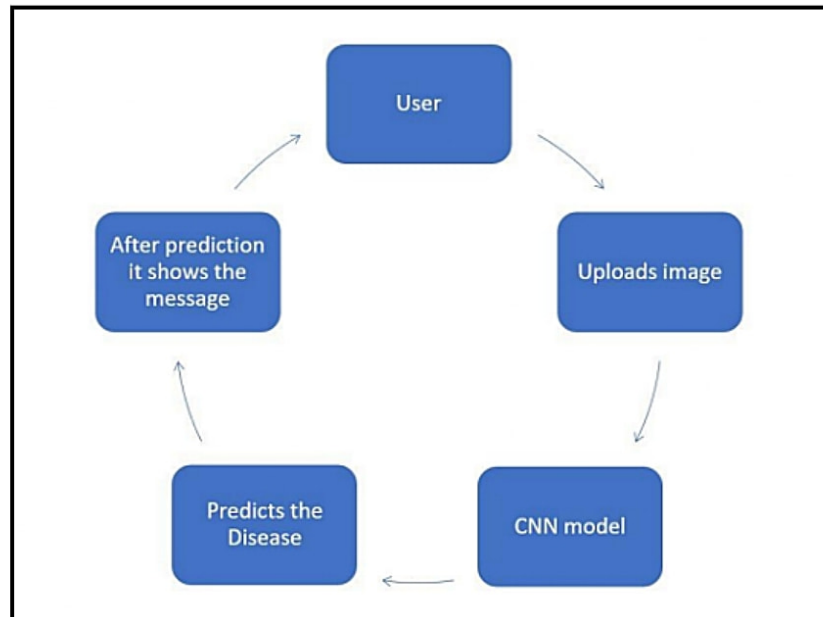
To accomplish the above task you must complete the below activities and tasks :

- Download the dataset.
- Classify the dataset into train and test sets.
- Add the neural network layers.
- Load the trained images and fit the model.
- Test the model.
- Save the model and its dependencies.
- Build a Web application using a flask that integrates with the model built.

6 RESULT:

➢ `Home Page:`

➢ **Prediction Page:**
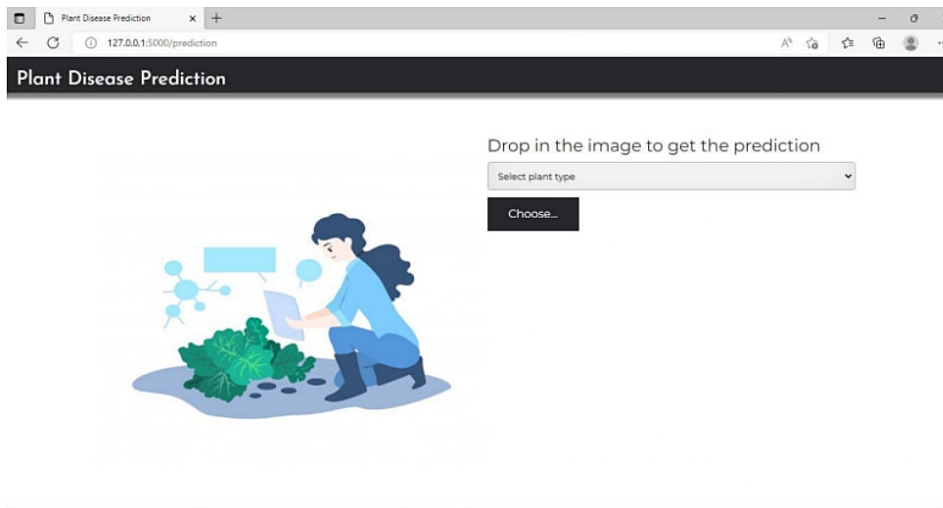
➢ **Result Page:**

Plant Disease Prediction

Drop in the image to get the prediction

Fruit

Choose...

Prediction: Oopps!! Your apple plant is infected by Black Rots. This infection is a fungal infection. To control balck rot, remove the cankers by pruning at least 15 inches below the end and burn or bury them. Treating the sites with the antibiotic streptomycin or a copper-based fungicide will be helpful.

Plant Disease Prediction

Drop in the image to get the prediction

Vegetable

Choose...

Prediction: Yaayy!! Your pepper plant is healthy. But, take the necessary precautions like, putting the plant where it gets at least 10 hours of direct sunlight. Keep soil evenly moist for good growth. Peppers need well draining soil that is rich and loamy, but avoid too much nitrogen in the soil. Too much nitrogen can cause plenty of leaves and little to no peppers. Your soil should have a pH between 6.0 and 6.5.

7 ADVANTAGES & DISADVANTAGES:

ADVANTAGES:

- The proposed model could predict the disease just from the image of a particular plant
- Easy to use UI
- Model has some good accuracy in detecting the plant just by taking the input(leaf).

DISADVANTAGES:

- Prediction is limited to few plants as we havent trained all the plants .

8 APPLICATIONS :

This webapplication can be used by farmers or users to check whether their plant is infected or not and can also show the remedy so that the user can take necessary precautions.

These kind of web applications can be used in the agricultural sector as well as for small house hold plants as well.

9 CONCLUSION :

Agriculture is the most important sector in today's life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality.

In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques. Usage of such applications could help the farmers to necessary precautions so that they dont face any loss as such.

10 FUTURE SCOPE :

As of now we have just build the web application which apparently takes the input as an image and then predict the out in the near future we can develop an application which computer vision and AI techniques to predict the infection once you keep the camera near the plant or leaf this could make our project even more usable.

11 BIBILOGRAPHY :

- http://www.ijstr.org/final-print/nov2019/Fertilizers-Recommendation-SystemFor-Disease-Prediction-In-Tree-Leave.pdf

- https://www.sciencedirect.com/science/article/pii/S0168169921004245

- http://www.ijetajournal.org/volume-8/issue-2/IJETA-V8I2P1.pdf

- https://www.semanticscholar.org/paper/Fertilizers-Recommendation-SystemFor-Disease-In-Neela-Nithya/495379d3ef2b461fabd2de8d0605c164cb1e396f

- https://ieeexplore.ieee.org/document/8878781

- https://www.irjet.net/archives/V7/i10/IRJET-V7I1004.pdf

jupyter vegetable Last Checkpoint: Last Friday at 11:23 AM (autosaved)

Logout

File   Edit   View   Insert   Cell   Kernel   Widgets   Help

Not Trusted   Python 3 (ipykernel) O

Run   Code

```
In [1]: pip install keras-cv
```

Requirement already satisfied: keras-cv in c:\users\kalya\anaconda3\lib\site-packages (0.2.8)
Requirement already satisfied: packaging in c:\users\kalya\anaconda3\lib\site-packages (from keras-cv) (21.3)
Requirement already satisfied: absl-py in c:\users\kalya\anaconda3\lib\site-packages (from keras-cv) (1.1.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\kalya\anaconda3\lib\site-packages (from packaging->keras-c
v) (3.0.4)
Note: you may need to restart the kernel to use updated packages.

```
In [ ]:
```

```
In [2]: pwd
```

Out[2]: 'C:\\Users\\KALYA\\PycharmProjects\\IBM-Fertilizers'

```
In [3]: from keras.preprocessing.image import ImageDataGenerator
```

```
In [4]: train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range = 0.2,horizontal_flip = True)
        test_datagen = ImageDataGenerator(rescale = 1)
```

```
In [5]: pwd
```

Out[5]: 'C:\\Users\\KALYA\\PycharmProjects\\IBM-Fertilizers'

```
In [6]: x_train = train_datagen.flow_from_directory('C:\\Users\\KALYA\\PycharmProjects\\IBM-Fertilizers\\Dataset Plant Disease\\Veg-data
        x_test = test_datagen.flow_from_directory('C:\\Users\\KALYA\\PycharmProjects\\IBM-Fertilizers\\Dataset Plant Disease\\Veg-dataset
```

Found 3416 images belonging to 9 classes.
Found 11386 images belonging to 9 classes.

```
In [7]: x_train.class_indices
```

Out[7]: {'Pepper,_bell___Bacterial_spot': 0,
 'Pepper,_bell___healthy': 1,
 'Potato___Early_blight': 2,
 'Potato___Late_blight': 3,
 'Potato___healthy': 4,
 'Tomato___Bacterial_spot': 5,
 'Tomato___Late_blight': 6,
 'Tomato___Leaf_Mold': 7,
 'Tomato___Septoria_leaf_spot': 8}

CNN

```
In [8]: from keras.models import Sequential
```

```
In [9]: from keras.layers import Dense
```

```
In [10]: from keras.layers import Convolution2D
```

```
In [11]: from keras.layers import MaxPooling2D
```

```
In [12]: from keras.layers import Flatten
```

```
In [13]: model = Sequential()
```

```
In [14]: model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
```

```
In [15]: model.add(MaxPooling2D(pool_size=(2,2)))
```

```
In [16]: model.add(Flatten())
```

```
In [17]: model.summary()
```

```
Model: "sequential"
_____
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      896

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 flatten (Flatten)           (None, 127008)            0

=================================================================
Total params: 896
Trainable params: 896
Non-trainable params: 0
_____
```

```
In [18]: 32*(3*3*3+1)
```

```
Out[18]: 896
```

Hidden layers

```
In [19]: model.add(Dense(units=300,activation='relu'))
         model.add(Dense(units=150,activation = 'relu'))
         model.add(Dense(units=75,activation = 'relu'))
```

Output layers

```
In [20]: model.add(Dense(9,activation = 'softmax'))
```

```
In [21]: model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

```
In [22]: model.summary()
```

```
Model: "sequential"
_____
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      896

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 flatten (Flatten)           (None, 127008)            0

 dense (Dense)               (None, 300)               38102700

 dense_1 (Dense)             (None, 150)               45150

 dense_2 (Dense)             (None, 75)                11325

 dense_3 (Dense)             (None, 9)                 684

=================================================================
Total params: 38,160,755
Trainable params: 38,160,755
Non-trainable params: 0
_____
```

```
In [23]: len(x_train)

Out[23]: 214

In [24]: 5384//16

Out[24]: 336

In [25]: len(x_test)

Out[25]: 712

In [26]: model.fit(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test),epochs=10)

         Epoch 1/10
         214/214 [==============================] - 87s 406ms/step - loss: 2.0595 - accuracy: 0.3899 - val_loss: 397.1047 - val_accurac
         y: 0.2604
         Epoch 2/10
         214/214 [==============================] - 80s 372ms/step - loss: 1.0632 - accuracy: 0.6291 - val_loss: 536.4133 - val_accurac
         y: 0.4037
         Epoch 3/10
         214/214 [==============================] - 82s 384ms/step - loss: 0.8579 - accuracy: 0.6929 - val_loss: 909.3461 - val_accurac
         y: 0.3702
         Epoch 4/10
         214/214 [==============================] - 85s 396ms/step - loss: 0.6663 - accuracy: 0.7684 - val_loss: 1085.8826 - val_accurac
         y: 0.3576
         Epoch 5/10
         214/214 [==============================] - 85s 398ms/step - loss: 0.6128 - accuracy: 0.7942 - val_loss: 1351.7244 - val_accurac
         y: 0.3027
         Epoch 6/10
         214/214 [==============================] - 84s 394ms/step - loss: 0.5200 - accuracy: 0.8150 - val_loss: 1236.5426 - val_accurac
         y: 0.3532
         Epoch 7/10
         214/214 [==============================] - 82s 385ms/step - loss: 0.4967 - accuracy: 0.8302 - val_loss: 1210.0035 - val_accurac
         y: 0.3230
         Epoch 8/10
         214/214 [==============================] - 80s 375ms/step - loss: 0.4607 - accuracy: 0.8364 - val_loss: 1775.0532 - val_accurac
         y: 0.3494
         Epoch 9/10
         214/214 [==============================] - 80s 373ms/step - loss: 0.4047 - accuracy: 0.8566 - val_loss: 1681.2104 - val_accurac
         y: 0.2761
         Epoch 10/10
         214/214 [==============================] - 80s 375ms/step - loss: 0.3613 - accuracy: 0.8753 - val_loss: 1408.9833 - val_accurac
         y: 0.3250

Out[26]: <keras.callbacks.History at 0x11f82c2beb0>

In [27]: model.save('vegtable.h5')
```

Testing:

Test the model

```
In [18]: import numpy as np
         from tensorflow.keras.models import load_model
         from tensorflow.keras.preprocessing import image
```

```
In [19]: #Load the model

         model =load_model("vegetable.h5")
```

```
In [23]: img = image.load_img("D:\\Project Buildathon\\Fertilizers_Recommendation_ System_For_Disease_ Prediction\\Dataset Plant Disease\\Veg-dataset\\Veg-dataset\\train_set\\Peppe
         img
```

Out[23]:


```
In [24]: x=image.img_to_array(img)
         x=np.expand_dims(x,axis=0)
         pred = np.argmax(model.predict(x),axis=1)

         index=['Pepper,_bell___Bacterial_spot',
          'Pepper,_bell___healthy',
          'Potato___Early_blight',
          'Potato___Late_blight',
          'Potato___healthy',
          'Tomato___Bacterial_spot',
          'Tomato___Late_blight',
          'Tomato___Leaf_Mold',
          'Tomato___Septoria_leaf_spot']

         index[pred[0]]
```

Out[24]: 'Pepper,_bell___healthy'

```
In [25]: img = image.load_img("D:\\Project Buildathon\\Fertilizers_Recommendation_ System_For_Disease_ Prediction\\Dataset Plant Disease\\Veg-dataset\\Veg-dataset\\train_set\\Tomat
         img
```

Out[25]:


```
In [26]: x=image.img_to_array(img)
         x=np.expand_dims(x,axis=0)
         pred = np.argmax(model.predict(x),axis=1)

         index=['Pepper,_bell___Bacterial_spot',
          'Pepper,_bell___healthy',
          'Potato___Early_blight',
          'Potato___Late_blight',
          'Potato___healthy',
          'Tomato___Bacterial_spot',
          'Tomato___Late_blight',
          'Tomato___Leaf_Mold',
          'Tomato___Septoria_leaf_spot']

         index[pred[0]]
```

Out[26]: 'Tomato___Septoria_leaf_spot'

```
In [ ]:
```

SI-87641-1657796462

SI-87641-1657796462