**EASYCROP**

Predicting The Fertilizer Type On The Bases Of Environmental Data

A Report

***Submitted by***

ABHINAV CHOUDHRY – 20CS002267

AVI MODI – 20CS002266

T SATHVIK REDDY – 20CS002268

UKANT JADIA – 20CS002286

Bachelor of Technology

*IN*

**Computer Science & Engineering**

**SCHOOL OF ENGINEERING**

****

**Sir Padampat Singhania University, Udaipur**

Under the guidance of

DR. KAMAL KANT HIRAN

**CONTENT TABLE**

|  |  |
| --- | --- |
| Sr. No. | Title |
| 1 | Tech. Used |
| 2 | Introduction |
| 3 | Problem Statement |
| 4 | Project Goals |
| 5 | Existing Situation |
| 6 | Potential Solution |
| 7 | Benefits Of Recommended Solution |
| 8 | Evaluation |
| 9 | Conclusion |
| 10 | Bibliography |
| 11 | Code Design |

**TECH USED**

* Hardware: -

Ryzen 5 3550H 3.2GHz  
8 GB of RAM  
2666 MHz

* Software: -

Apache NetBeans IDE 12.6

AWS

* Programming language: -

Java (GUI)

Python (model)

**INTRODUCTION**

In this age of overpopulation. Our world grows hungry every day. To address this scenario of agricultural necessity we take the case of Bangladesh and its scenario of the agriculture cycle. Bangladesh has a large agricultural sector supporting the majority of the population for their livelihood. Every year huge resources in terms of land and fertilizer are used for the production of food. Most of the fertilizers being used in farms today are being wasted due to the incorrect use of a type of fertilizer. Farmers do not have enough knowledge about soil nutrient contents such as nitrogen, phosphorous, and potassium in the land. However, the balanced dose of fertilizer used for a different crop is very important for obtaining desirable yield which depends on several edaphic and environmental factors.

The elemental composition especially nitrogen, phosphorus, and potassium content of fertilizer needs to be considered in selecting a source fertilizer as these elements play a vital role in plant growth and development.

This requires a development of a system to address the misuse of fertilizers and make sure that the farmers know how to increase the yield by using the correct fertilizers by taking the external factors into consideration.

This is where our project **EasyCrop** comes into the picture. it makes sure that the farmers are following the right steps to increase the yield so that more food is generated which in time feeds many hungry mouths all over the world.

**PROBLEM STATEMENT**

“Farmer not achieving the full potential that can be achieved through the use of fertilizers.”

The farmers usually have little control over the usage of fertilizers. There is a need for proper guidance for the optimal usage of these fertilizers which is required by farmers in order to get more yields and prevent wastage. Research works carried out so far only identify nutrient deficiency and disease detection. The proposed work identifies and determines the area of the region of the deficient region and predicts the amount of fertilizer consumption by these deficient plants. Thus the identification of nutrient deficiency is helpful to the farmer for proper monitoring of crop growth and optimum usage of fertilizers.

There are also many prediction algorithms with different accuracy in different data sets, we will be taking them into consideration to show which is the most optimal of them all.

**PROJECT GOALS**

A fertilizer requirement prediction model would be very useful for obtaining a higher yield of different crops in Bangladesh. The problem statement of Bangladesh has a large agricultural sector supporting the majority of the population for their livelihood. Every year huge resources in terms of land and fertilizer are used for the production of food. Most of the fertilizers being used in farms today are being wasted due to the incorrect use of a type of fertilizer. Farmers do not have enough knowledge about soil nutrient contents such as nitrogen, phosphorous, and potassium in the land.

Our goal in this project is to provide the necessary tools to the farmers so that they don’t make any mistakes when it comes to the part of fertilizing the crops. The vital part of this project would be using suitable fertilizers for its compatible crop by considering external or environmental factors.

We will also be comparing various prediction algorithms to get the most optimal accuracy solution:

* Decision Tree Classifier
* Random Forest Classifier
* Gradient Boosting Classifier
* K Neighbors Classifier
* SVM( Support Vector Machine)
* f. Navie bayes (GuassianNB)

**EXISTING SYSTEM**

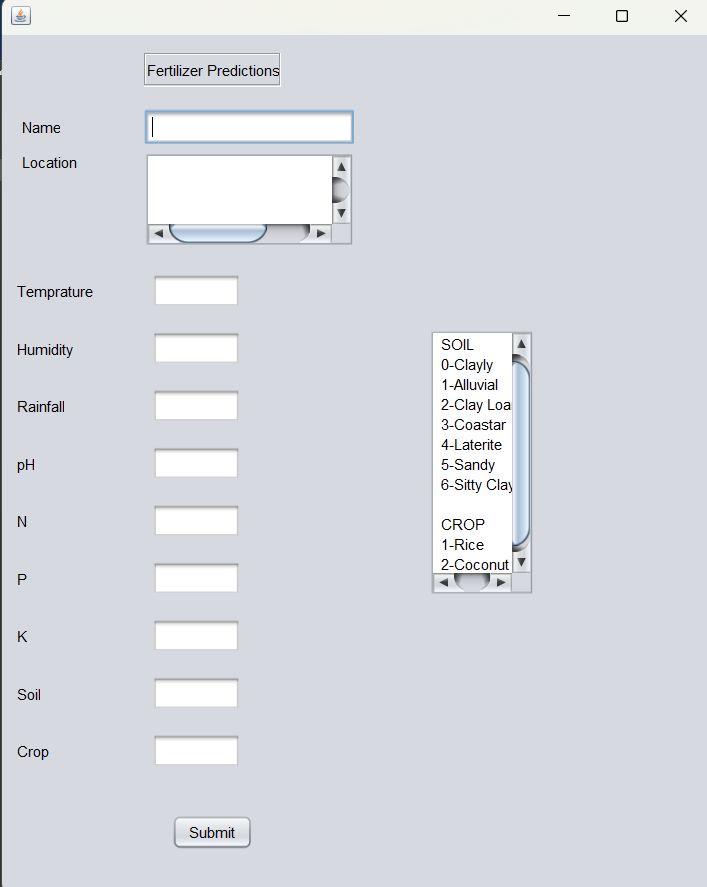
* Niketa et al in 2016 indicated that the yield of the crop depends on the seasonal climate. In India, climate conditions vary unconditionally. In times of drought, farmers face serious problems. So taken into consideration they used some machine learning algorithms to help the farmers to suggest the crop for the better yield. They take various data from previous years to estimate future data. They used SMO classifiers in WEKA to classify the results. The main factors that take into consideration are minimum temperature, maximum temperature, average temperature, and the previous year’s crop information and yield information. Using the SMO tool they classified the previous data into two classes that are high yield and low yield.
* Eswari et al 2018 have indicated that the yield of the crop depends on the perception, average, minimum and maximum temperature. Apart, from that, they have taken one more attribute named crop evapotranspiration. Crop evapotranspiration is a function of both the weather and the growth stage of the plant. This attribute is taken into consideration to get a good decision on the yield of the groups. They all collected the dataset with these attributes and send it as input to the Bayesian network and classify into the two classes named true and false classes and compared it with the observed classifications in the model with a confusion matrix and bring the accuracy. Finally, they concluded that crop yield prediction with Naïve Bayes and Bayesian network gives high accuracy when compared to SMO classifier, and forecasting the crop yield prediction in different climate and cropping scenarios will be beneficial.
* **DISADVANTAGES OF THE EXISTING SYSTEM:**
* The obtained result for the crop yield prediction using SMO classifier gives less accuracy when compared to naïve Bayes, multilayer perceptron, and Bayesian network. Previously yield is predicted on the bases of the farmer's prior experience but now weather conditions may change drastically so they cannot guess the yield.

**POTENTIAL SOLUTION**

In this present project, we will use the standard machine learning models that will analyze various soil features like N, P, K, soil pH, and environmental parameters such as temperature, relative humidity, and annual rainfall of a particular land area to recommend the type of fertilizer to be used for the selected crops. Therefore, the present project has been proposed to analyse available datasets and to predict fertilizer requirements for two different crops based on soil features and environmental parameters using appropriate machine learning models.

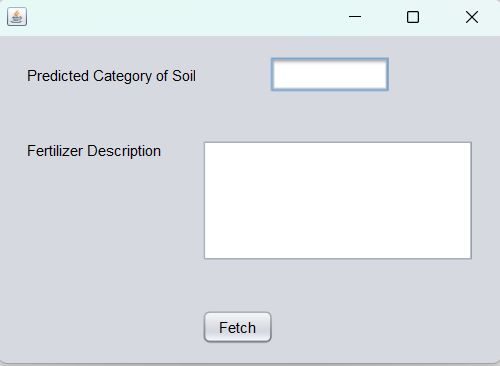
By comparing various prediction algorithms, we will get to know the most optimal accuracy of all of the given classifiers :

* Decision Tree Classifier
* Random Forest Classifier
* Gradient Boosting Classifier
* K Neighbors Classifier
* SVM (Support Vector Machine)
* Navie bayes (GuassianNB)



**BENEFITS OF RECOMMENDED SOLUTION**

* From this project we will be able to find a suitable machine learning model to predict fertilizer that best suits a particular type of soil and crop based on other environmental consequences. Our proposed machine learning model will help growers to decide appropriate fertilizer rapidly and economically to reduce fertilizer losses.
* It also helps in increasing agriculture yield.
* Better accuracy than existing system.
* Achieving the maximum crop at minimum yield is the ultimate Aim of the project.



**EVALUATION**

The data used in this project is made by augmenting and combining various publicly available datasets like weather, soil, etc.

This data is relatively simple with very few but useful features unlike the complicated features affecting the requirement of fertilizer of the crop.

The data have Nitrogen, Phosphorous, Potassium and pH values of the soil. Also, it also contains the humidity, temperature and rainfall required for a particular crop.

**The description of the variables used in this dataset**.

**Feature variables**:

**Temperature**: Temperature in °C

**Humidity**: Relative humidity in %

**Rainfall**: Annual rainfall as in mm

**pH**: Soil pH

**N**: Available Nitrogen at soil

**P**: Available Phosphorus at soil

**K**: Available Potassium at soil

**Soil**: 1. Clayey, 2. Laterite, 3. Silty clay, 4. Sandy, 5. Coastal, 6. Clay loam, 7. Alluvial

**Crop**: 1. Rice and 2. Coconut

**Target Variable**:

**Fertilizer**

**1.DAP and MOP**: Source of P and K  
**2.Good NPK**: NPK are sufficient in soil. No need to apply any fertilizer.  
**3.MOP**: Source of K  
**4.Urea and DAP**: Source of N and P  
**5.Urea and MOP**: Source of N and K  
**6.Urea**: Source of N only  
**7.DAP**: Source of P only

DAP= Diammonium phosphate, MOP= Muriate of Potash

Predicting the final accuracy by comparing various classifiers (prediction algorithm).

**CONCLUSION**

From this project we will be able to find a suitable machine learning model to predict fertilizer that best suits a particular type of soil and crop based on other environmental consequences. Our proposed machine learning model will help growers to decide appropriate fertilizer rapidly and economically to reduce fertilizer losses.

By comparing the various prediction algorithms, we now can give the farmers the most optimal of choice with more accuracy among all the algorithms.

“Happy farmer, happy population”

**BIBLIOGRAPHY**

* <https://jpinfotech.org/crop-yield-prediction-and-efficient-use-of-fertilizers/>
* <https://github.com/tipubari2013/Fertilizer_Prediction_ML/blob/main/Group-1_Fertilizer_Prediction20211118.ipynb>
* C. P. Wickramasinghe, P. L. N. Lakshitha, H. P. H. S. Hemapriya, A. Jayakody and P. G. N. S. Ranasinghe, "Smart Crop and Fertilizer Prediction System," 2019 International Conference on Advancements in Computing (ICAC), 2019, pp. 487-492, doi: 10.1109/ICAC49085.2019.9103422
* K. Hampannavar, V. Bhajantri and S. G. Totad, "Prediction of Crop Fertilizer Consumption," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1-5, doi: 10.1109/ICCUBEA.2018.8697827.
* REFERENCE:
* S.Bhanumathi, M.Vineeth and N.Rohit, “Crop Yield Prediction and Efficient use of Fertilizers”, IEEE International Conference on Communication and Signal Processing, April 4-6, 2019.

**CODE DESIGN**

Python prediction code-

import pandas as pd # to read and manipulating data

import numpy as np # to calculate mean and standard deviations

from sklearn.tree import DecisionTreeClassifier ,plot\_tree

from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier

from sklearn.neighbors import KNeighborsClassifier

# from sklearn.preprocessing import MinMaxScaler # to normalize data

from sklearn.preprocessing import LabelEncoder # to encode object variable to numeric

from sklearn.model\_selection import train\_test\_split # to split data into trainin

from sklearn.metrics import plot\_confusion\_matrix

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn import svm

from sklearn.naive\_bayes import GaussianNB

from xgboost import XGBClassifier

df = pd.read\_csv('Fertilizer\_Prediction.csv')

#Label Encoding

le = LabelEncoder()

df['Fertilizer']= le.fit\_transform(df['Fertilizer'])

df['Soil']= le.fit\_transform(df['Soil'])

df['Crop']= le.fit\_transform(df['Crop'])

X = df.drop(['Fertilizer'], axis=1) #feature variables

y = df[['Fertilizer']] #Target variable

# Create train and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.5, random\_state=10)

result={}

def plotter(x\_model):

plot\_confusion\_matrix(x\_model,

X\_test,

y\_test,

display\_labels=['DAP and MOP','Good NPK','MOP','Urea and DAP','Urea and MOP','Urea','DAP'],

xticks\_rotation ='vertical',

cmap='YlGnBu')

def score\_report(y\_pred\_x,model\_x):

# print("Accuracy and report of model ",model\_x)

# print('Accuracy: ', accuracy\_score(y\_test, y\_pred\_x))

result.update({model\_x:accuracy\_score(y\_test, y\_pred\_x)})

# print(classification\_report(y\_test, y\_pred\_x))

#to build a classification tree

model\_dt = DecisionTreeClassifier(random\_state=42)

model\_dt = model\_dt.fit(X\_train, y\_train)

y\_pred\_dt = model\_dt.predict(X\_test)

# plotter(model\_dt)

score\_report(y\_pred\_dt,model\_dt)

#Random Forest model

model\_RF = RandomForestClassifier(random\_state=42)

model\_RF = model\_RF.fit(X\_train, y\_train)

y\_pred\_RF = model\_RF.predict(X\_test)

# plotter(model\_RF)

score\_report(y\_pred\_RF,model\_RF)

# Gradient Boosting model

model\_GB = GradientBoostingClassifier()

model\_GB = model\_GB.fit(X\_train, y\_train)

y\_pred\_GB = model\_GB.predict(X\_test)

# plotter(model\_GB)

score\_report(y\_pred\_GB,model\_GB)

# Create K-Nearest Neighbors Classifier

model\_knn = KNeighborsClassifier(n\_neighbors=3)

model\_knn = model\_knn.fit(X\_train, y\_train)

y\_pred\_knn = model\_knn.predict(X\_test)

# plotter(model\_knn)

score\_report(y\_pred\_knn,model\_knn)

# using SVM

model\_svm = svm.SVC(kernel='linear')

model\_svm.fit(X\_train,y\_train)

y\_pred\_svm = model\_svm.predict(X\_test)

# plotter(model\_svm)

score\_report(y\_pred\_svm, model\_svm)

# using naive based

model\_nb = GaussianNB()

model\_nb.fit(X\_train,y\_train)

y\_pred\_nb = model\_nb.predict(X\_test)

# plotter(model\_nb)

score\_report(y\_pred\_nb, model\_nb)

# reading the input from the file

lst =[]

with open('input.txt', 'r') as f:

for i in f:

data=i

lst = list(data.split(","))

f.close()

# converting the input list to dataframe rwo

frame = pd.DataFrame([lst])

model = max(result,key=result.get)

y\_pred = model.predict(frame)

if y\_pred == 0:

y\_pred= "DAP"

elif y\_pred == 1:

y\_pred = "DAP & MOP"

elif y\_pred == 2:

y\_pred = "Good NPK"

elif y\_pred == 3:

y\_pred = "MOP"

elif y\_pred == 4:

y\_pred = "Urea"

elif y\_pred == 5:

y\_pred = "Urea & DAP"

else:

y\_pred = "Urea & MOP"

# # writting the output in file

f\_out = open('output.txt', 'w')

f\_out.write(y\_pred)

f\_out.close()

print(y\_pred)

dispout-

import java.io.BufferedReader;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import static java.lang.System.in;

import java.util.Arrays;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

/\*\*

\*

\* @author abhin

\*/

public class dispout extends javax.swing.JFrame {

private Object br;

/\*\*

\* Creates new form dispout

\*/

public dispout() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

t1 = new javax.swing.JTextField();

jLabel2 = new javax.swing.JLabel();

jScrollPane1 = new javax.swing.JScrollPane();

ta = new javax.swing.JTextArea();

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jLabel1.setText("Predicted Category of Soil");

jLabel2.setText("Fertilizer Description");

ta.setColumns(20);

ta.setRows(5);

jScrollPane1.setViewportView(ta);

jButton1.setText("Fetch");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton2.setText("back");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(23, 23, 23)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(60, 60, 60)

.addComponent(t1, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 128, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addGroup(layout.createSequentialGroup()

.addGap(69, 69, 69)

.addComponent(jButton1)

.addGap(88, 88, 88)

.addComponent(jButton2)))

.addContainerGap(23, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(17, 17, 17)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(t1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(38, 38, 38)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel2)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 38, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton1)

.addComponent(jButton2))

.addGap(15, 15, 15))

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

// TODO add your handling code here:

int count =0;

File file = new File("C:\\Users\\abhin\\OneDrive\\Desktop\\project\\output.txt");

FileInputStream fis = null;

try {

fis = new FileInputStream(file);

} catch (FileNotFoundException ex) {

Logger.getLogger(dispout.class.getName()).log(Level.SEVERE, null, ex);

}

byte[] bytesArray = new byte[(int)file.length()];

try {

fis.read(bytesArray);

} catch (IOException ex) {

Logger.getLogger(dispout.class.getName()).log(Level.SEVERE, null, ex);

}

String s = new String(bytesArray);

System.out.println("Contents of the given file are :: " +new String(bytesArray));

t1.setText(s);

if("DAP & MOP".equals(s))

{

ta.setText("Soil is already rich in P and K \n use fertilizer rich in NITROGEN");

}

else if("Good NPK".equals(s))

{

ta.setText("Soil is already rich in nutrients \n no need to use fertilizers");

}

else if("DAP".equals(s))

{

ta.setText("Soil is already rich in P only \n use fertilizer rich in NITROGEN and POTASSIUM");

}

else if("MOP".equals(s))

{

ta.setText("Soil is already rich in K only \n use fertilizer rich in NITROGEN and PHOSPHOROUS");

}

else if("Urea".equals(s))

{

ta.setText("Soil is already rich in N only \n use fertilizer rich in PHOSPHOROUS and POTASSIUM");

}

else if("Urea & DAP".equals(s))

{

ta.setText("Soil is already rich in N and P\n use fertilizer rich in POTASSIUM");

}

else if("Urea & MOP".equals(s))

{

ta.setText("Soil is already rich in N and K\n use fertilizer rich in PHOSPHOROUS");

}

}//GEN-LAST:event\_jButton1ActionPerformed

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton2ActionPerformed

// TODO add your handling code here:

InputFields o = new InputFields();

o.setVisible(true);

dispose();

}//GEN-LAST:event\_jButton2ActionPerformed

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(dispout.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(dispout.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(dispout.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(dispout.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new dispout().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTextField t1;

private javax.swing.JTextArea ta;

// End of variables declaration//GEN-END:variables

}

Inputfield java code-

import java.io.BufferedReader;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.io.UnsupportedEncodingException;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template

\*/

/\*\*

\*

\* @author abhin

\*/

public class InputFields extends javax.swing.JFrame {

/\*\*

\* Creates new form InputFields

\*/

public InputFields() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

jPanel1 = new javax.swing.JPanel();

jScrollPane1 = new javax.swing.JScrollPane();

location = new javax.swing.JTextArea();

jLabel5 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

name = new javax.swing.JTextField();

jPanel2 = new javax.swing.JPanel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel6 = new javax.swing.JLabel();

jLabel7 = new javax.swing.JLabel();

jLabel8 = new javax.swing.JLabel();

jLabel9 = new javax.swing.JLabel();

jLabel10 = new javax.swing.JLabel();

jLabel11 = new javax.swing.JLabel();

jLabel12 = new javax.swing.JLabel();

temprature = new javax.swing.JTextField();

humidity = new javax.swing.JTextField();

rainfall = new javax.swing.JTextField();

ph = new javax.swing.JTextField();

n = new javax.swing.JTextField();

p = new javax.swing.JTextField();

k = new javax.swing.JTextField();

soil = new javax.swing.JTextField();

crop = new javax.swing.JTextField();

jButton1 = new javax.swing.JButton();

jScrollPane2 = new javax.swing.JScrollPane();

jTextArea1 = new javax.swing.JTextArea();

jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setModalExclusionType(java.awt.Dialog.ModalExclusionType.APPLICATION\_EXCLUDE);

jLabel1.setText("Fertilizer Predictions");

jLabel1.setBorder(javax.swing.BorderFactory.createEtchedBorder());

location.setColumns(20);

location.setRows(5);

jScrollPane1.setViewportView(location);

jLabel5.setText("Location");

jLabel4.setText("Name");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(16, 16, 16)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel4)

.addComponent(jLabel5))

.addGap(54, 54, 54)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 168, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(name, javax.swing.GroupLayout.PREFERRED\_SIZE, 168, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel4)

.addComponent(name, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jLabel5)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 76, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(24, 24, 24))

);

jLabel2.setText("Temprature");

jLabel3.setText("Humidity");

jLabel6.setText("Rainfall");

jLabel7.setText("pH");

jLabel8.setText("N");

jLabel9.setText("P");

jLabel10.setText("K");

jLabel11.setText("Soil");

jLabel12.setText("Crop");

p.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

pActionPerformed(evt);

}

});

javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);

jPanel2.setLayout(jPanel2Layout);

jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel12)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(crop, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel11)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(soil, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel10)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(k, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel9)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(p, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(n, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(ph, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(rainfall, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(humidity, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(jPanel2Layout.createSequentialGroup()

.addComponent(jLabel2)

.addGap(47, 47, 47)

.addComponent(temprature, javax.swing.GroupLayout.PREFERRED\_SIZE, 71, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(139, Short.MAX\_VALUE))

);

jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()

.addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel2)

.addComponent(temprature, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel3)

.addComponent(humidity, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel6)

.addComponent(rainfall, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel7)

.addComponent(ph, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel8)

.addComponent(n, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel9)

.addComponent(p, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel10)

.addComponent(k, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel11)

.addComponent(soil, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel12)

.addComponent(crop, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(19, Short.MAX\_VALUE))

);

jButton1.setText("Submit");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jTextArea1.setText("SOIL\n0-Clayly\n1-Alluvial\n2-Clay Loam\n3-Coastar\n4-Laterite\n5-Sandy\n6-Sitty Clay\n\nCROP\n1-Rice\n2-Coconut");

jScrollPane2.setViewportView(jTextArea1);

jButton2.setText("OUTPUT");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(114, 114, 114)

.addComponent(jLabel1))

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(136, 136, 136)

.addComponent(jButton1)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jButton2)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 84, javax.swing.GroupLayout.PREFERRED\_SIZE))))

.addContainerGap(142, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(15, 15, 15)

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 27, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 119, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(63, 63, 63)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 213, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton1)

.addComponent(jButton2))

.addContainerGap(36, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>//GEN-END:initComponents

private void pActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_pActionPerformed

// TODO add your handling code here:

}//GEN-LAST:event\_pActionPerformed

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

// TODO add your handling code here:

String temp = temprature.getText();

String humid = humidity.getText();

String rain = rainfall.getText();

String ph1 = ph.getText();

String nitro = n.getText();

String phos = p.getText();

String pot = k.getText();

String soilt = soil.getText();

String cropt = crop.getText();

String insrt = String.format("%s,%s,%s,%s,%s,%s,%s,%s,%s",temp,humid,rain,ph1,nitro,phos,pot,soilt,cropt);

PrintWriter writer = null;

try {

writer = new PrintWriter("input.txt", "UTF-8");

} catch (FileNotFoundException ex) {

Logger.getLogger(InputFields.class.getName()).log(Level.SEVERE, null, ex);

} catch (UnsupportedEncodingException ex) {

Logger.getLogger(InputFields.class.getName()).log(Level.SEVERE, null, ex);

}

writer.print(insrt);

writer.close();

String dir = null;

String command = "python C:\\Users\\abhin\\OneDrive\\desktop\\project\\Fertilizer\_model.py";

try {

Process p = Runtime.getRuntime().exec(command);

} catch (IOException ex) {

Logger.getLogger(InputFields.class.getName()).log(Level.SEVERE, null, ex);

}

}//GEN-LAST:event\_jButton1ActionPerformed

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton2ActionPerformed

// TODO add your handling code here:

dispout o = new dispout();

o.setVisible(true);

dispose();

}//GEN-LAST:event\_jButton2ActionPerformed

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(InputFields.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(InputFields.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(InputFields.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(InputFields.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new InputFields().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JTextField crop;

private javax.swing.JTextField humidity;

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel10;

private javax.swing.JLabel jLabel11;

private javax.swing.JLabel jLabel12;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

private javax.swing.JLabel jLabel7;

private javax.swing.JLabel jLabel8;

private javax.swing.JLabel jLabel9;

private javax.swing.JPanel jPanel1;

private javax.swing.JPanel jPanel2;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JTextArea jTextArea1;

private javax.swing.JTextField k;

private javax.swing.JTextArea location;

private javax.swing.JTextField n;

private javax.swing.JTextField name;

private javax.swing.JTextField p;

private javax.swing.JTextField ph;

private javax.swing.JTextField rainfall;

private javax.swing.JTextField soil;

private javax.swing.JTextField temprature;

// End of variables declaration//GEN-END:variables

}