A

PROJECT REPORT

On

"SOIL CLASSIFICATION AND CROP SUGGESTION USING ML"

Submitted to:

Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur for

Partial Fulfillment of the Degree of

Bachelor of Engineering in Computer Science & Engineering

Ms. Divya Bawane **Ms. Komal Ambagade**
Ms. Rita Dhoble **Ms. Sanjana Virmuttu**
Ms. Vidya Tulaskar

Name of Guide

Prof. Abhimanyu Dutonde

Computer Science and Engineering

NAAC Accredited with A+ Grade **ISO 9001:2015 Certified**

Tulsiramji Gaikwad-Patil College of Engineering & Technology, Nagpur-441108

(An Autonomous Institute affiliated to RTMNU, Nagpur)

Session 2022-23

Tulsiramji Gaikwad-Patil College
Engineering & Technology, Nagpu
441108

(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur)

CERTIFICATE

This is to certify that project work described in this report entitled, "**Soil Classification And Crop Suggestion Using ML**" was carried out by **Ms. Divya Bawane ,Ms. Komal Ambagade, Ms. Rita Dhoble, , Ms. Sanjana Virmuttu, Ms. Vidya Tulaskar** in Tulsiramji Gaikwad-Patil College of Engineering & Technology, Nagpur under my supervision and guidance in partial fulfillment of the requirement for the degree of Bachelor of Engineering in **Computer Science & Engineering** of Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur.

This work is the own work of the candidate, completed in all respect and is of sufficiently high standard to warrant its submission to the said degree. The assistance and resources used for this work are duly acknowledged.

Prof. Abhimanyu Dutonde
Guide

Prof. Abhay Bagade
Project Coordinator (Branch)

HoD (CSE)

Principal

Date: / / 2023

Tulsiramji Gaikwad-Patil College of Engineering & Technology, Nagpur-441108

(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur)

DECLARATION

I hereby declare that this project titled **"Soil Classification And Crop Suggestion Using ML"** is a bonafide and authentic record of the work done by me under supervision of **Prof. Abhimanyu Dutonde** during academic session 2022-23.

The work presented here is not duplicated from any other source and also not submitted earlier for any other degree/diploma of any university. I understand that any such duplication is liable to be punished in accordance with the university rules. The source material, data used in this research have been duly acknowledged.

Date:

Place:

Name and Signature of Students

Ms. Divya Bawane

Ms. Komal Ambagade

Ms. Rita Dhoble

Ms. Sanjana Virmuttu

Ms. Vidya Tulaskar

Tulsiramji Gaikwad-Patil College of Engineering & Technology, Nagpur-441108

(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur)

ACKNOWLEDGEMENT

With profound feeling of immense gratitude and affection, I would like to thank my guide **Prof. Abhimanyu Dutonde**, **Computer Science & Engineering**, for his continuous support, motivation, enthusiasm and guidance. His encouragement, supervision with constructive criticism and confidence enabled me to complete this project.

We, also wish to extend my reverences to **Prof. Abhimanyu Dutonde, Computer Science & Engineering** for providing necessary facilities to complete my project.

We, are also thankful to all the faculty members and all non-teaching staff of the department & college for their cooperation throughout the project work.

We, also put forth our deepest sense of gratitude towards the Principal, TGPCET for constant motivation and providing necessary infrastructure.

PROJECTEE

Date:

Place:

Ms. Divya Bawane
Ms. Komal Ambagade
Ms. Rita Dhoble
Ms. Sanjana Virmuttu
Ms. Vidya Tulaskar

PUBLICATION BASED ON THIS WORK

Sr. No.	Paper title	Authors	Journal	Publishing Date
1.	Soil Classification and crop suggestion using ML	Abhimanyu Dhutonde, Divya Bawane, Komal Ambagade, Rita Dhoble, Sanjana Virmiuttu Vidya Tulaskar,	International Research Journal of Modernization in Engineering Technology and Science	Volume:05/ Issue:06/ June-2023

CONTENTS

Certificate i

Declaration ii

Acknowledgement iii

Publication based on the present work iv

CHAPTER – I

Overview of Present Work 1-6

1.1Overview of the project 1

1.2 Background 1

1.3 Problem Statement 4

1.4 Research Objectives 4

1.5 Scope of the Present work 4

CHAPTER – II

Literature Review 7-10

2.1 Introduction 7

CHAPTER – III

Formulation of Present Work 11-22

3.1 Soil Classification using Machine Learning 11

3.2 System Overview 11

3.3 Proposed Method 12

3.4 Overview of Convolutional Neural Network in Image Classification 14

3.5 Support Vector Machine (SVM) in Classification 16

3.6 Implementation 22

CHAPTER – IV

Requirements 24-36

4.1 Hardware Requirements 24

4.2 Software Requirements 25

4.3 Python Libraries 31

4.4 Algorithms Used 35

CHAPTER – V

Result and Discussion **37-38**

CHAPTER – VI

Conclusion And Future Scope **39**

6.1 Conclusion 39

6.2 Future Scope 39

REFERENCES **40-41**

PUBLICATION

APPENDIX I **57-64**

FIGURE INDEX

Figure 3.3.1 Typical digital image processing cycle 12

Figure 3.3.2 System Architecture of Soil Classification and Crop Suggestion System 13

Figure 3.4.1 CNN Classification Process 14

Figure 3.4.2 Soil Classification and Crop Suggestion Module Flow 15

Figure 3.5.1.1 computations of data point’s separation depend on a kernel function 17

Figure 3.5.2.1 Three Class Classification 19

Figure 3.5.2.2 Three Class Separation 19

Figure 3.5.2.3 Class distribution by hyperplanes 20

Figure 4.2.1.1 Python Programming Application 27

Figure 4.2.2.1 Features of Pycharm 30

Figure 4.4.1 Working of CNN model 35

Figure 4.4.2 Support Vector Machine 36

Figure 5.1 Soil Classification and Crop Suggestion Module GUI 37

Figure 5.2 Input Soil Image 37

Figure 5.3 Soil Classification in Progress 38

CHAPTER – I

OVERVIEW OF PRESENT WORK

CHAPTER – I
Overview of Present Work

Summary:

This chapter contains the brief overview of the project topic with briefly explained aim, objectives and scope of the study behind the project selection. The contents are arranged point wise to understand it clearly.

1.1 Overview of the project

Changes in land cover will cause the changes in the climate and environmental characteristics, which has an important influence on the social economy and ecosystem. The main form of land cover is different types of soil. Compared with traditional methods, visible and near-infrared spectroscopy technology can classify different types of soil rapidly, effectively, and nondestructively. Based on the visible near-infrared spectroscopy technology, this project takes the soil of seven different land cover types in Alluvial Soil, Arid Soil, Black Soil, Laterite Soil, Mountain Soil, Red Soil, and Yellow Soil as examples and establishes a convolutional neural network classification model. The classification results of different number of training samples are analyzed and compared with the support vector machine algorithm. Under the condition that algorithm divides the calibration set, the classification results of seven different soil types and single seven soil types by convolutional neural network are better than those by the support vector machine. Under the condition of randomly dividing the calibration set according to the proportion of 1/3 and 1/4, the classification results by convolutional neural network are also better. The aim of this study is to analyze the feasibility of land cover classification with small samples by convolutional neural network and, according to the deep learning algorithm, to explore new methods for rapid, nondestructive, and accurate classification of the land cover.

1.2 Background

The soil is composed of gravel, sand, silt, and clay as a result of disintegration or by disintegration, transportation, and deposition of rocks. There are several methods to find the properties of soils. The methods followed in the examination of soils are complementary to each other, it is impossible to obtain information about the behavior of soils without determining the properties and changes of soil characteristics. Geotechnical engineers are able to determine which characteristics have the most impact on soil behavior. The soils are heterogeneous. It can be expected to vary within meters. Soils remain under various influences such as loading, dewatering, drying, and freezing over the years. The reactions of the soil in these cases are important both in the use of the soil as a building material and in the structures to be built upon. Soils can have infinitely different properties due to the composition of its mineral or organic contents. It is difficult to apply probability methods to such a subject. It is also considered that to determine the soil characteristics require long-term and expensive experiments. Therefore, various researchers presented statistical methods in the form of regression analysis in order to determine the soil properties which provide reliable results and also can be obtained rapidly and inexpensively. The buildings that make up the living areas of people are mostly built on soils. Accurate estimation of the properties of the soils on which these buildings will be build will provide economic gain for the design of the buildings and will guarantee their lives and assets for the people living in it. The soil classification system has been one of the communication languages among the engineers in geotechnical engineering applications. The determination of soil classification is not eliminating the need for detailed soil investigations and other laboratory tests on soil samples which we determine the engineering properties. However, an engineer can determine the behavior of the soil in the case of structural loads in the application phase by classified soil. It is an inevitable fact that clays, which are frequently encountered in soil mechanics problems, have a wide range in terms of their engineering characteristics.

The grains forming the soil have a very different geometry and are of a wide variety of sizes. Knowing the grain size distribution in the soil plays an important role in determining the index properties of soils. The grain size distribution is the ratio of the weight of the grains of various diameters to the total dry weight of the soil in percent. Soils are divided into two types: coarse-grained soils (gravel and sand) and fine-grained soils (clay and silt). In order to determine the grain size distribution of the coarse-grained soils according to the in diameters, the sieve analysis is carried out and the hydrometer test is performed to determine the grain size distribution of the fine-grained soils according to the diameters. Research on Machine Learning continues on software and hardware. Today, MACHINE LEARNING applications can be found in many areas such as economics, industrial engineering, automation, electronic circuit design, electronic engineering, computer engineering, medicine, various intelligence problems, optical perception, and object identification. MACHINE LEARNINGs have also been successfully utilized in the field of geotechnical and construction engineering with the advancements in computational sciences and in computational power. MACHINE LEARNINGs are inspired by biological neurons (nerve cells), resulting in artificially simulated studies of the brain's working system. The distinguishing feature of MACHINE LEARNING from other methods of computation is that they perform operations using the learning feature of the human brain. Classical statistical methods recognize that the relationship between dependent and independent variables is linear, which results in insufficiencies as well as inefficiencies in the studies. In geotechnical sciences, parameters are controlled by many variables such as environmental factors, dynamic characteristics, and pore water pressure, where the relationships between these variables may be both linear and non-linear. The interdependent interaction of these features may make it difficult and time-consuming to utilize classical statistical methods. The application of a series of methods developed by MACHINE LEARNING provides alternative solutions to the problems in geotechnical sciences or offers supplementary tools to the classical statistical methods in geotechnical studies.

MACHINE LEARNING change its structure and weight of the neurons throughout its training and development by randomly distributed input parameters. It has a structure that can adapt itself like a nervous system of a living organism. In other words, it can change its structure and learn according to an internal and external stimulus. In the decision-making stage, the connection weights are activated and find the solution by itself. Therefore, it is not known what the system will do under a certain situation. This is the factor that adds an unknown feature to the system. The MACHINE LEARNING generally generates a set of data sets corresponding to an input data set. In this context, the final MACHINE LEARNING model consists of three layers, an input layer where the input data is entered, a hidden layer where the data is processed, and an output layer where the results are obtained. The other important component of the MACHINE LEARNING model is the connections between the layers. Each connection has a weight value. The weights of these connections are altered to develop a successful MACHINE LEARNING tool throughout its training which provides favorable output results for a given set of input values. The weights generated during training are the values in which necessary information is stored. Although MACHINE LEARNING is a proven technology and has a wide variety of usage and implementations in almost all science divisions, it is not entirely known how these weights are calculated and assigned. In this respect, the MACHINE LEARNING content has not been fully solved and is criticized for this reason via various researchers.

1.3 Problem Statement

The aim of this study is to explain the estimation of the desired parameters using the learning method of the CNN with the available data. In cases where classical statistical methods such as multiple linear regression are insufficient and there is no linear relationship between variables, CNN can provide solutions to these type of problems and can be utilized successfully. Similarly, the linear relationship between the values of the sieve analysis and the Atterberg values used in the estimation is insufficient, CNN can be used for such a process. Sieve analysis and hydrometer analysis are required to determine whether the soils are fine-grained or coarse-grained, while Atterberg limits are required for the classification of fine-grained soils. Each process is laborious and expensive. The number of processes can be reduced by using sieve analysis values in estimating Atterberg limits.

1.4 Research Objective

The main objective of this thesis is to estimate the SVM limits from the grain size distribution values by the CNN method and to determine the soil classification from these estimated values. To achieve this goal;

i. Training the model of CNN with sieve analysis and SVM limits obtained from previous projects. ii. Simulate the trained model with another set of data with the same characteristics.
ii. Determination of soil classification with labelling their names.
iii. The comparison of the determined soil classifications with the original soil classification and crop suggestion.

1.5 Scope of the Present Work

Land cover is a direct result of the interaction between natural environment and human activities. It mainly focuses on describing the natural properties of the earth's surface which has specific time and space characteristics. Changes in land cover will cause the changes in the climate and environmental characteristics, which has an important influence on the social economy and ecosystem [1], [2]. The main form of land cover is different types of soil, including cultivated lands, woodlands, grasslands, and bare lands. Therefore, it is of great significance to classify different types of soil quickly and accurately for land cover research, soil investigation, and mapping.

The early classification method is the land use topographic map obtained by combining with the actual ground survey. Now the classification technology of the remote sensing image is mostly used to realize the classification of different types of soil [3] [4]. Visible and near-infrared spectroscopy technology is a fast, nondestructive measurement method. It has been widely used in medicine, agriculture, oil, and other fields [5] [6]. The spectral analysis method indirectly obtains useful information of the substance. Through establishing an effective correction model between the spectrum and the information, the result is obtained [7] [8]. The spectral technology is introduced into the classification of soil, the remote sensing image information is replaced by the spectral information, and different types of soil models are established. It can be fast and nondestructive to realize the classification of soil.

Deep learning is modeled by simulating the neural structure of the human brain and has made breakthroughs in applications such as image recognition and speech recognition [9] [10]. The commonly used classification method support vector machine (SVM) is a machine learning method based on statistical learning theory. The idea behind SVM is that input samples are projected from low-dimensional feature space to high-dimensional space through nonlinear mapping, which allows data in the low-dimensional space that is not linearly separable to transform into linearly separable data in the high-dimensional space [11] [12]. While deep learning is to transform the original signal layer by layer, transform the feature representation in the original space to the new feature space. And it automatically learns to get the hierarchical feature representation, and the classification result is achieved [13]. Convolution neutral network MACHINE LEARNING is a network structure in the deep learning, which has a good effect in the classification of images and makes the MACHINE LEARNING method widely used in many fields [14]. MACHINE LEARNING is a new and nondestructive method for the application of quality monitoring of agricultural products, includes the detection and grading of fruits, vegetables, etc., and has achieved good results [15] [16]. MACHINE LEARNING is usually used for classification modeling with large sample size.

Based on the visible near-infrared spectroscopy technology, this paper took six different types of soil orchards, woodlands, tea plantations, farmlands, bare land, and grasslands in Qingdao, China, as examples, and established a convolutional neural network classification model. The classification results under the conditions of different label samples were analyzed, and the classification results with the shallow network SVM were compared. In this paper, the aim is to analyze the feasibility of land cover classification with small samples by MACHINE LEARNING and explore new methods for rapid, nondestructive, and accurate classification of soil according to the MACHINE LEARNING.

CHAPTER – II

LITERARTURE REVIEW

CHAPTER – II
Literature Review

Summary:

In this chapter various research studies are discussed below with their research results and the methods are also discussed with the respective research gaps.

2.1 Introduction

Jianzhou Gong et al [1] the results gave some affirmative answers. Landscape pattern exhibited obviously scale-dependent to grain size and extent. The landscape metrics with gradient analysis could quantitatively approach spatial pattern of urbanization. A precise location for urbanized area, like city center and sub-center, could be identified by multiple landscape metrics. Multiple adjunctive centers occurred as indicated by analysis of radiation zones around the city center. Directional differences of landscape pattern along the two transects (N-S and W-E) came into being.

Lu, Dengsheng et al [2] this paper explores approaches to improve urban land-cover classification with QuickBird imagery. Traditional per-pixel spectral-based supervised classification, incorporation of textural images and multispectral images, spectral-spatial classifier, and segmentation-based classification are examined in a relatively new developing urban landscape, Lucas do Rio Verde in Mato Grosso State, Brazil. This research shows that use of spatial information during the image classification procedure, either through the integrated use of textural and spectral images or through the use of segmentation-based classification method, can significantly improve land-cover classification performance.

F. Gao et al [3] this paper evaluates remote sensing approaches for mapping crop phenology using vegetation index time-series generated by fusing Landsat and MODIS (Moderate Resolution Imaging Spectroradiometer) surface reflectance imagery to improve temporal sampling over that provided by Landsat alone. The results show that detailed spatial and temporal variability in vegetation development across this landscape can be identified using the fused Landsat-MODIS data.

Jinwei Dong et al [4] this study evaluated the potential of Landsat 8 images on annual paddy rice mapping in NE Asia which was dominated by single cropping system, including Japan, North Korea, South Korea, and NE China. The cloud computing approach was used to process all the available Landsat 8 imagery in 2014 (143 path/rows, ~ 3290 scenes) with the Google Earth Engine (GEE) platform. The results indicated that the Landsat 8, GEE, and improved PPPM algorithm can effectively support the yearly mapping of paddy rice in NE Asia.

N. Noman et al [5] determine the optimal feature-combination for classification of functional near-infrared spectroscopy (fNIRS) signals with the best accuracies for development of a two-class brain-computer interface (BCI). Using a multi-channel continuous-wave imaging system, mental arithmetic signals are acquired from the prefrontal cortex of seven healthy subjects.

S. Liu et al [6] developed the CFSSL, which will become an important supplement to the soil spectral libraries of China and those of the world, via large-scale, standardized soil sampling and spectral measurements. Second, this study confirms vis-NIR spectroscopy technique is cost- and time-efficient for SOC monitoring, and provides an important methodological reference for large-scale SOC spectroscopic.

M. Paradelo et al [7] models used to evaluate leaching of contaminants to groundwater are very sensitive to sorption coefficients (K_d). These models need reliable K_d data at the field scale, but the number of samples required makes the classic batch sorption experiments inappropriate for this purpose. Since visible–near infrared (vis–NIR) spectroscopy is an inexpensive and fast method, it has been used for predicting soil properties related to soil sorption capacity. In this study, authors aimed to predict the spatial variation of K_d from vis–NIR spectra for two contaminants: phenanthrene (sorbed on organic fractions) and glyphosate (sorbed on mineral fractions).

D. Cozzolino et al [8] the aim of this review is to provide with an overview of different applications of near infrared (NIR) spectroscopy addressing issues related with contamination in soil, sediments and water. A discussion on the main factors or variables that affect the results of this type of applications is provided.

Hassan et al [9] these are not only confined to the classification process but they also extend to the pre-processing steps. Pre-processing steps include segmentation of unwanted regions in images, selection of features for big data and degrading textual reports to their abstract form. Regardless of the fact that this model can be used for different types of medical data, it is highly suitable for cancer medical data. This is due to the fact that cancer medical data contain different datasets; namely; numbers, images and text medical reports. The Particle Swarm Optimization (PSO) and Artificial Neural Networks (ANN) were used to develop an algorithm to classify breast cancer datasets. The algorithm effectively differentiated between malignant and cancer free datasets with a 93.2% accuracy. The model provides an efficient tool for processing breast cancer datasets.

J. Zhou et al [12] identifying functional effects of noncoding variants is a major challenge in human genetics. To predict the noncoding-variant effects *de novo* from sequence, we developed a deep learning–based algorithmic framework, DeepSEA (http://deepsea.princeton.edu/), that directly learns a regulatory sequence code from large-scale chromatin-profiling data, enabling prediction of chromatin effects of sequence alterations with single-nucleotide sensitivity. We further used this capability to improve prioritization of functional variants including expression quantitative trait loci (eQTLs) and disease-associated variants.

J.Gao et al [11] proposed a classification method support vector machine (SVM) with spectral information and texture features (ST-SVM), which incorporates texture features in remotely sensed images into SVM. Using kernel methods, the spectral information and texture features are jointly used for the classification by a SVM formulation. Then, the texture features were calculated based on segmented block matrix image objects using the panchromatic band. A comparison of classification results on real-world data sets demonstrates that the texture features in this paper are useful supplement information for the spectral object-oriented classification, and proposed ST-SVM classification accuracy than the traditional SVM method with only spectral information.

Jie Xu et al [12] analyze the excess misclassification error of SVMC based on u.e.M.c. samples, and obtain the optimal learning rate of SVMC for u.e.M.c. samples. We also introduce a new Markov sampling algorithm for SVMC to generate u.e.M.c. samples from given dataset, and present the numerical studies on the learning performance of SVMC based on Markov sampling for benchmark datasets. The numerical studies show that the SVMC based on Markov sampling not only has better generalizatio++n ability as the number of training samples are bigger, but also the classifiers based on Markov sampling are sparsity when the size of dataset is bigger with regard to the input dimension.

T. Serre et al [13] show that the model can perform recognition tasks on datasets of complex natural images at a level comparable to psychophysical measurements on human observers during rapid categorization tasks. In sum, the evidence suggests that the theory may provide a framework to explain the first 100–150 ms of visual object recognition. The model also constitutes a vivid example of how computational models can interact with experimental observations in order to advance our understanding of a complex phenomenon.

Ying Li et al [14] the proposed method views the HSI cube data altogether without relying on any pre-processing or post-processing, extracting the deep spectral–spatial-combined features effectively. In addition, it requires fewer parameters than other deep learning-based methods. Thus, the model is lighter, less likely to over-fit, and easier to train. For comparison and validation, we test the proposed method along with three other deep learning-based HSI classification methods namely, stacked auto encoder (SAE), deep brief network (DBN), and 2D-CNN-based methods on three real-world HSI datasets captured by different sensors. Experimental results demonstrate that our 3D-CNN-based method outperforms these state-of-the-art methods and sets a new record.

J. Steinbrener [15] show an easy way to classify hyper-spectral images with state of the art convolutional neural networks pre-trained for RGB image data. A small, custom dataset of hyper-spectral images was recorded from staged but realistic scenes. With this dataset, an ImageNet pre-trained convolutional neural network was fine-tuned to obtain a classifier. An additional data compression layer has been added to be able to classify the hyper-spectral images with the RGB pre-trained network. To isolate the benefit of increased spectral resolution for the classification, the same analysis was also performed with pseudo-RGB images calculated from the hyper-spectral images. The results show that the hyper-spectral image data increases the average classification accuracy from 88.15% to 92.23%. The approach can easily be extended to other applications.

CHAPTER – III

FORMULATION OF PRESENT WORK

CHAPTER – III
Formulation of Present Work

3.1 Soil Classification using Machine Learning

Methods and Crop Suggestion Based on Soil Series This project creates a model that can predict soil series with land type and according to prediction it can suggest suitable crops. It makes use machine learning algorithms such as weighted K-Nearest Neighbor (KNN), Bagged Trees, and Gaussian kernel-based Support Vector Machines (SVM) to classify the soil series. The Soil classification philosophies used, follows the existence knowledge and practical circumstances. On the land surfaces of earth, classification of soil creates a link between soil samples and various kinds of natural entity. Based on these classifications and the mapped data, the suitable crops were suggested for a particular region.

3.2 System Overview

The group of soils which is formed from the same kind of parent materials and remains under the similar conditions of drainage, vegetation time and climate forms the soil series. It also has the same patterns of soil horizons with differentiating properties. Each soil series were named based on its locality. The main purpose of this system is to create a suitable model for classifying various kinds of soil series data along with suitable crops suggestion for certain regions of India. In this project, have worked on 7 soil series datasets obtained from kaggle. The dataset considered had 383 samples with 11 classes. The crop database was created by considering the Upazilla codes, map units and class labels. The method involved two phases: training phase and testing phase. Two datasets were used: Soil dataset and crop dataset. Soil dataset contains class labelled chemical features of soil. The system follows an architecture as seen in Fig 3.3.1. The machine learning methods were used to find the soil class. Three different methods used were: weighted K-NN, Gaussian Kernel based SVM, and Bagged Tree.

Weighted KNN:

KNN uses all training data, since weighting is used. The nearest neighbours are given more weightage than the farther ones. The distance is calculated and the majority of such classification is taken as the final classification. The obtained accuracy of the classification using KNN was 92.93 %

Gaussian Kernel based SVM:

SVM separates the objects of classes into different decision planes. A decision boundary separates the objects of one class from the object of another class. Support vectors are the data points which are nearest to the hyper-plane. Kernel function separates the non-linear data by transforming the inputs to higher dimensional space. In this project they have used gaussian kernel function. The accuracy obtained using SVM was 94.95%

Bagged Trees:

Bagging generates a set of models that classifies the random samples of data and predicts the classes. When given a new instance, the predictions of these models are aggregated to find the final prediction of class. The accuracy obtained using this algorithm was 90.91%.

After classifying the soil series, the crops, that are suitable for that series for the given map unit of corresponding upazilla were suggested. The results inferred that K-NN and Bagged tree showed comparative accuracy, but SVM outperforms the other two algorithms and produces an accuracy of 94.95%. The system was able to achieve an average accuracy of 92.93%.

3.3 Proposed Method

Data of different parameters affecting soil like temperature, humidity, air quality and PH of soil are stored in excel sheet. According to value of different parameters of soil, a list of best suited crop is selected from all crops. Values of monitoring parameters are adjusted according to optimal condition required for particular crop. All the data is stored on database. A GUI is developed in Python to suggest a crop which can be grown in farm seasonal wise. Geometric progression algorithm is used for predicting crop.

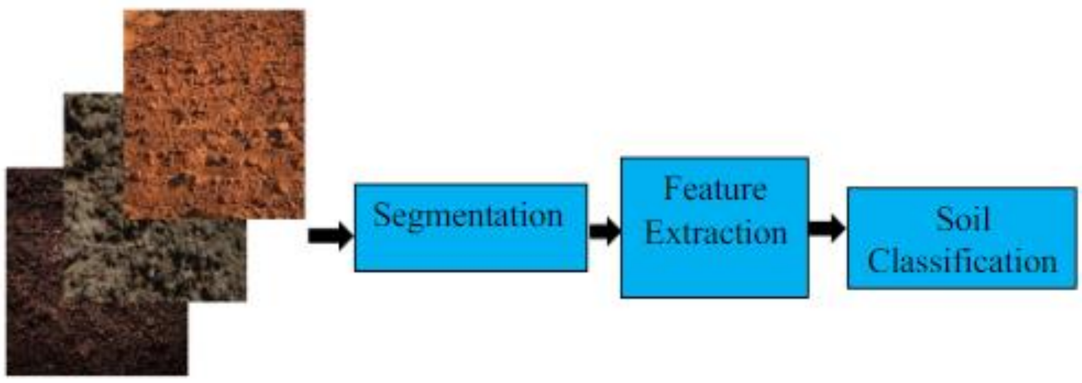


Figure 3.3.1 Typical digital image processing cycle

Soil classification can be performed keeping focus on different features of soil, such as soil particle size (clod/aggregate), texture, color or combination of features. Digital image processing and computer vision approaches can be employed on soil images for classification. Color is an extensive measure of the physical characteristics and chemical compositions of soil, so a substantial measures of soil statistics can be productively acquired by analyzing the soil color.

Soil texture in another property which has been used in various approaches to determine the soil type, mainly clay, silt and sand. Work has also been done on prediction of pH value for soil characterization and classification. On that account, the process of soil classification and qualitative distinction basing on color texture of the soil is very often used method. There is an expansion in the interest amongst the researchers for evolving automated processes for topography division based on soil images into soil spatial entities which aimed to replace conventional, expensive manual procedures for classifying and delineating soils. Various algorithms have been proposed in the recent past based on conventional image processing and computer vision techniques. Figure 3.3.2 shows a process with an input and an output, according to which the techniques can be decided. This section gives an elaborate review of such techniques used to classify soil proposed by various researchers and with different success rates.

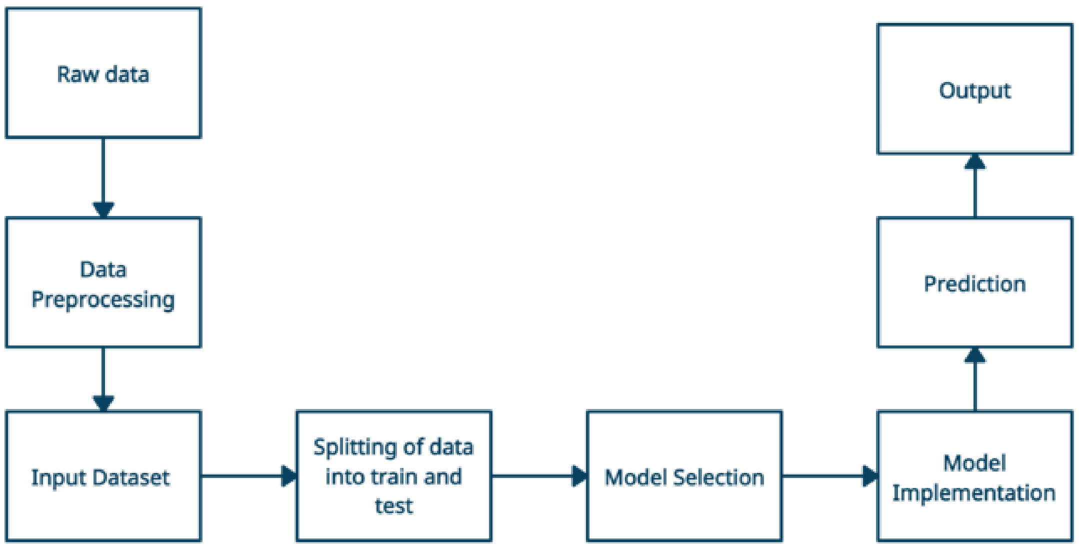
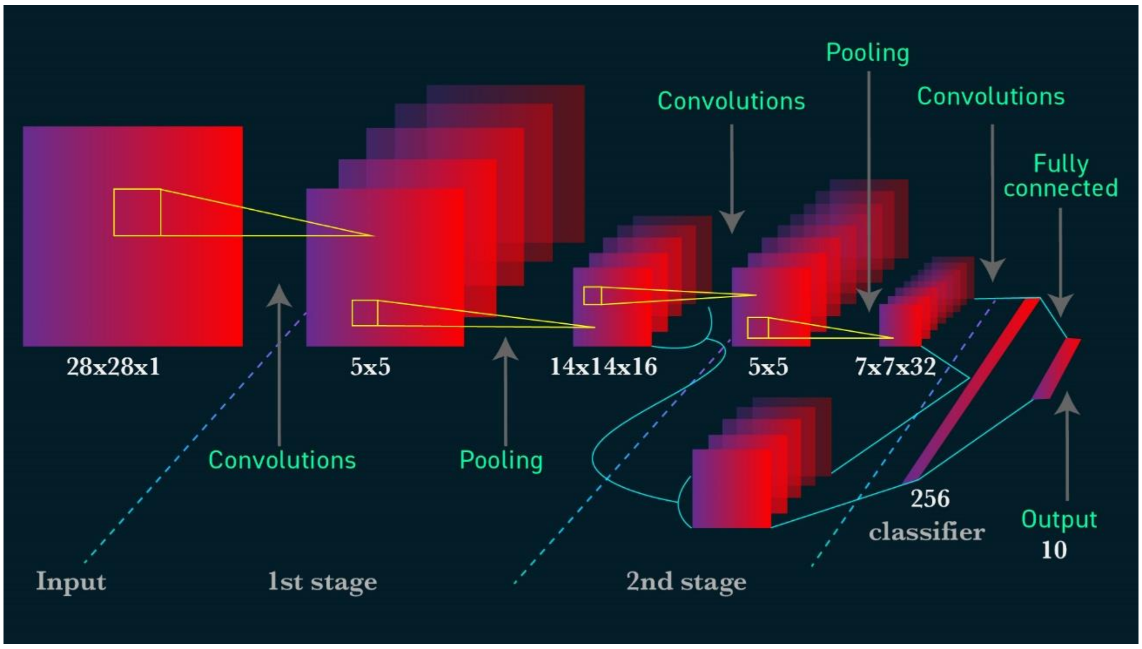


Figure 3.3.2 System Architecture of Soil Classification and Crop Suggestion System

The block diagram depicts the basic stages involved in creating the machine learning model above. The system's first stage involves cleaning and processing raw data. Crop data and fertilizer data are combined in this process to produce a final dataset. This dataset is then separated into training and testing data and utilized as input to the system. Model selection is the process of comparing various classification models in order to find the best appropriate one. The most suited crops are predicted, and the output is used for the following.

3.4 Overview of Convolutional Neural Network in Image Classification

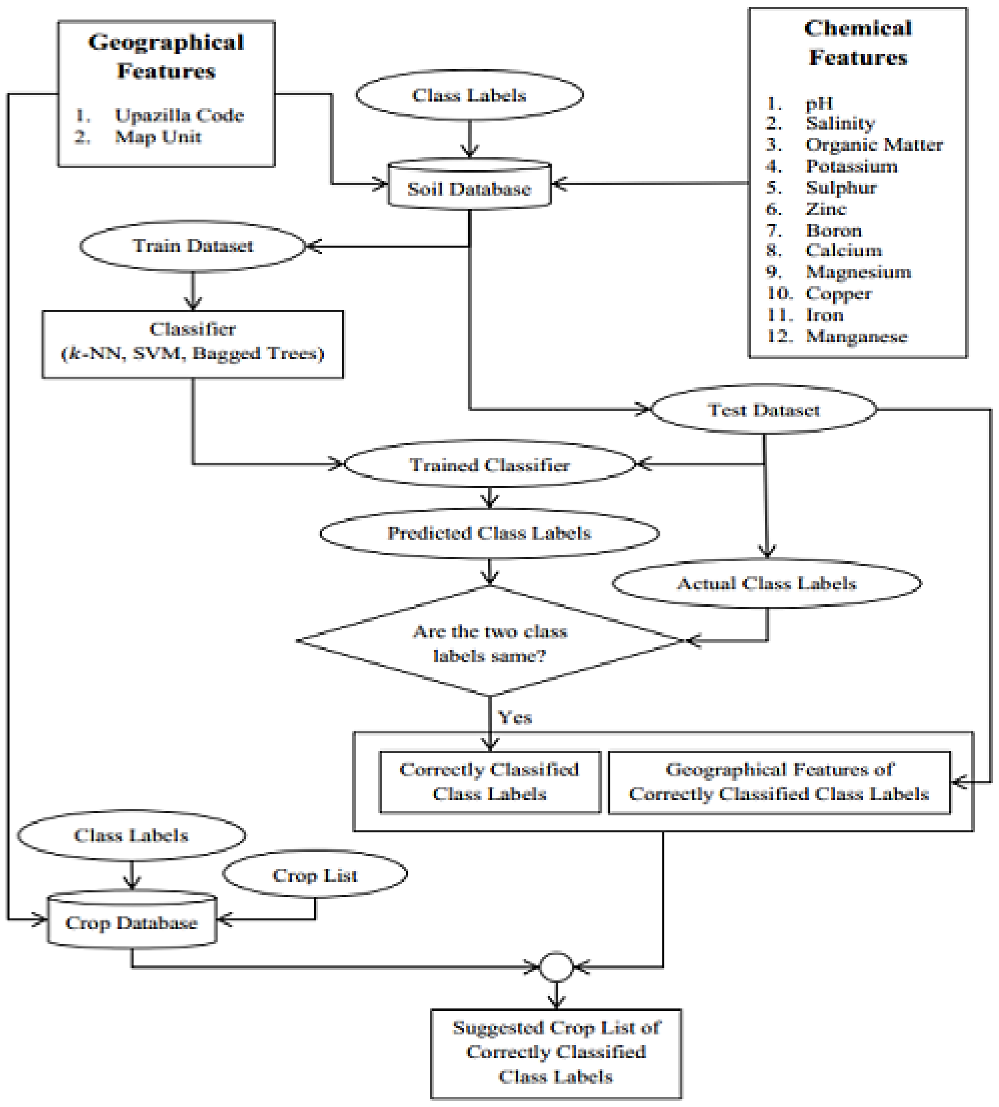
In machine learning, Convolutional Neural Networks (CNN or ConvNet) are complex feed forward neural networks. CNNs are used for image classification and recognition because of its high accuracy. It was proposed by computer scientist Yann LeCun in the late 90s, when he was inspired from the human visual perception of recognizing things. The CNN follows a hierarchical model which works on building a network, like a funnel, and finally gives out a fully-connected layer where all the neurons are connected to each other and the output is processed.



3.4.1 CNN Classification Process

We constructed a new ConvNet step-by-step, we will be implementing the datset data set for image classification. We will be building a ConvNet made of two hidden layers or convolutional layers. Now let us look at one of the images and the dimensions of the images. The archive contains two folders, a test folder and train folder with subdirectories corresponding to the possible soil types. The images are found within each subdirectory.

The photos are imported into the Python session. The first step is to process the images into a format that 1) makes the data readable to the model, and 2) provides more training material for the model to learn. For example, the *train_processor* variable scales the data so that it can be a feature (input) for the model, but also takes each images and augments it so that the model can learn from multiple variations of the same picture. It flips it horizontally, rotates it, and shifts it, and more to make sure the model learns from the shape of the bit rather than the orientation or size.



3.4.2 Soil Classification and Crop Suggestion Module Flow

The next step is to build the Convolutional Neural Network (CNN) model. Options are number of convolutional layers, fully connected dense layers, the number of nodes in each layer, and the number of training epochs.

The image is divided into pieces and classification occurs on the individual tiles. Below is the discussion about how the CNN Classification works:

- Image classifiers rely on Convolutional Neural Networks (CNNs) to process an image. CNNs are a special form of neural network with a specific architecture of layers.
- The four types of CNN layers are the convolutional layer, ReLU layer, pooling layer, and fully connected layer. An image classifier passes an image through these layers to generate a classification.
- The convolutional layer extracts the features of an image by scanning through the image with filters.
- The ReLU layer rectifies all negative values to zero. Its objective is to add non-linearity to the model.
- Pooling layers are applied to downsize an image and increase computational speed. The most common approach is max pooling, which takes the maximum value of each subregion.
- The fully connected layer concludes the CNN. It aggregates and weights all information and generates the final classification.

Two-third of the samples are used to train the model(s), rest one-third of the samples are used to test the models. For correctly classified samples, crop is suggested for the corresponding map unit of corresponding soil classes.

3.5 Support Vector Machine (SVM) in Classification

SVM is a supervised machine learning algorithm that helps in classification or regression problems. It aims to find an optimal boundary between the possible outputs.

SVM does complex data transformations depending on the selected kernel function and based on that transformations, it tries to maximize the separation boundaries between your data points depending on the labels or classes you’ve defined.

3.5.1 How SVM Works in Classification:

In the base form, linear separation, SVM tries to find a line that maximizes the separation between a two-class data set of 2-dimensional space points. To generalize, the objective is to find a hyperplane that maximizes the separation of the data points to their potential classes in an dimensional space. The data points with the minimum distance to the hyperplane (closest points) are called *Support Vectors*.

In the image below, the Support Vectors are the 3 points (2 blue and 1 green) laying on the scattered lines, and the separation hyperplane is the solid red line:

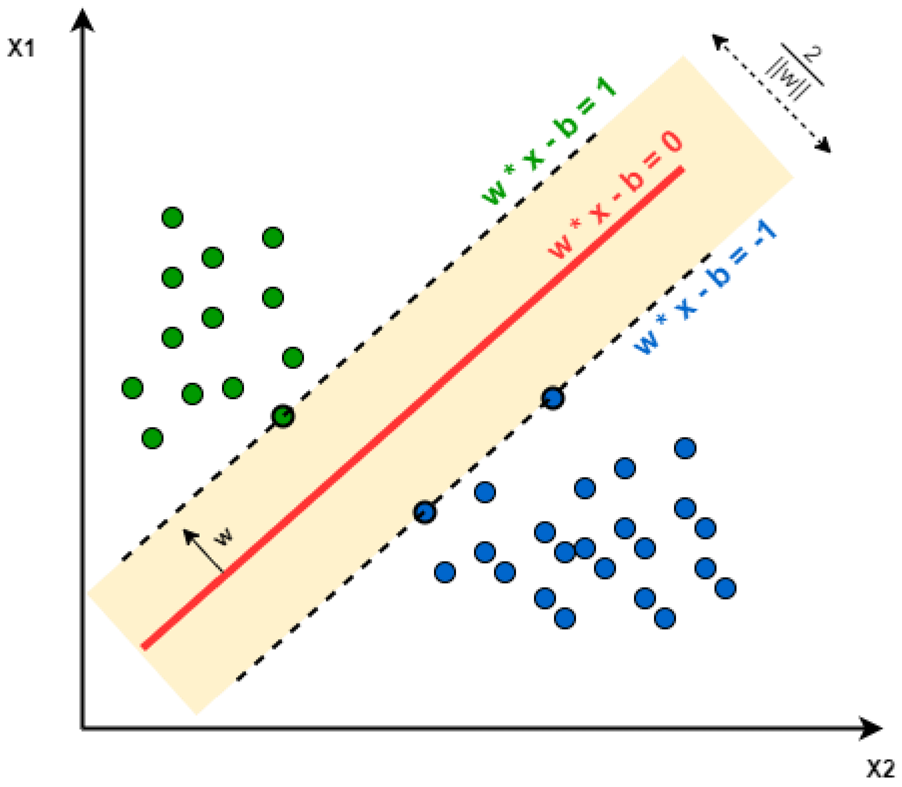


Figure 3.5.1.1 computations of data point's separation depend on a kernel function.

There are different kernel functions: Linear, Polynomial, Gaussian, Radial Basis Function (RBF), and Sigmoid. Simply put, these functions determine the smoothness and efficiency of class separation, and playing around with their hyperparameters may lead to overfitting or underfitting.

3.5.2 Multiclass Classification Using SVM

SVM doesn’t support multiclass classification natively. It supports binary classification and separating data points into two classes. For multiclass classification, the same principle is utilized after breaking down the multi classification problem into multiple binary classification problems.

The idea is to map data points to high dimensional space to gain mutual linear separation between every two classes. This is called a ***One-to-One*** approach, which breaks down the multiclass problem into multiple binary classification problems. A binary classifier per each pair of classes.

Another approach one can use is ***One-to-Rest***. In that approach, the breakdown is set to a binary classifier per each class.

A single SVM does binary classification and can differentiate between two classes. So that, according to the two breakdown approaches, to classify data points from classes’ data set:

- In the *One-to-Rest* approach, the classifier can use ***m*** SVMs. Each SVM would predict membership in one of the ***m*** classes.
- In the *One-to-One* approach, the classifier can use SVMs.

Let’s take an example of 3 class classification problem; green, red, and blue, as the following image:

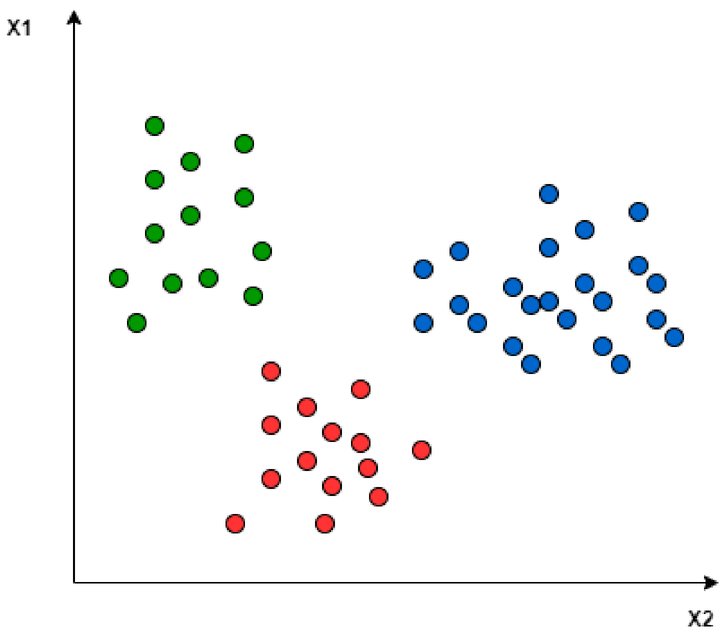


Figure 3.5.2.1 Three Class Classification

Applying the two approaches to this data set results in the followings:

In the *One-to-One* approach, we need a hyperplane to separate between every two classes, neglecting the points of the third class. This means the separation takes into account only the points of the two classes in the current split. For example, the red-blue line tries to maximize the separation only between blue and red points. It has nothing to do with green points:

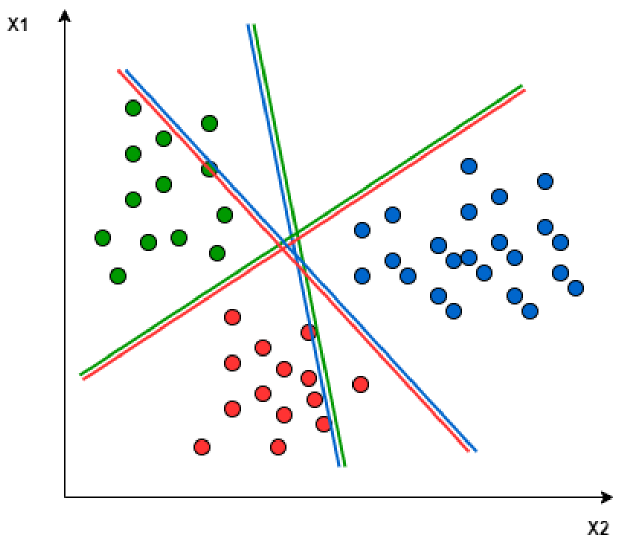


Figure 3.5.2.2 Three Class Separation

In the *One-to-Rest* approach, we need a hyperplane to separate between a class and all others at once. This means the separation takes all points into account, dividing them into two groups; a group for the class points and a group for all other points. For example, the green line tries to maximize the separation between green points and all other points at once:

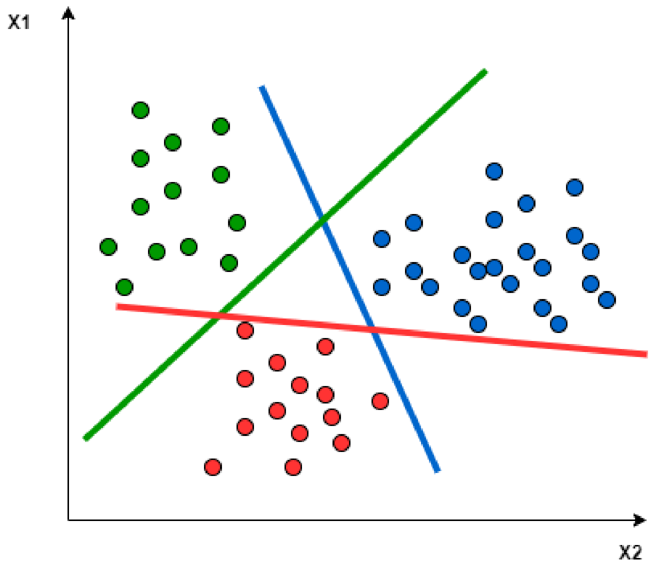


Figure 3.5.2.3 Class distribution by hyperplanes

3.5.3 SVM Multiclass Classification in Python

The following Python code shows an implementation for building (training and testing) a multiclass classifier (3 classes), using Python 3.10.1 and Scikitlean library.

We developed two different classifiers to show the usage of two different kernel functions; Polynomial and RBF. The code also calculates the accuracy and f1 scores to show the performance difference between the two selected kernel functions on the same data set.

In this code, we use the soil data set. That data set contains class classes of 50 instances each, where each class refers to a type of Iris plant.

We'll start our script by importing the needed classes:

```
from sklearn import svm, datasets

import sklearn.model_selection as model_selection

from sklearn.metrics import accuracy_score

from sklearn.metrics import f1_score
```

```
iris = datasets.load_iris()
```

Now we need to separate features set from the target column (class label), and divide the data set to 80% for training, and 20% for testing:

```
X = iris.data [:, :2]

y = iris.target

X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y,
train_size=0.80, test_size=0.20, random_state=101)
```

We'll create two objects from SVM, to create two different classifiers; one with Polynomial kernel, and another one with RBF kernel:

```
rbf = svm.SVC(kernel='rbf', gamma=0.5, C=0.1).fit(X_train, y_train)

poly = svm.SVC(kernel='poly', degree=3, C=1).fit(X_train, y_train)
```

To calculate the efficiency of the two models, we'll test the two classifiers using the test data set:

```
poly_pred = poly.predict(X_test)

rbf_pred = rbf.predict(X_test)
```

Finally, we'll calculate the accuracy and f1 scores for SVM with Polynomial kernel:

```
poly_accuracy = accuracy_score(y_test, poly_pred)

poly_f1 = f1_score(y_test, poly_pred, average='weighted')

print('Accuracy (Polynomial Kernel): ', "%.2f" % (poly_accuracy*100))

print('F1 (Polynomial Kernel): ', "%.2f" % (poly_f1*100))
```

In the same way, the accuracy and f1 scores for SVM with RBF kernel:

```
rbf_accuracy = accuracy_score(y_test, rbf_pred)
rbf_f1 = f1_score(y_test, rbf_pred, average='weighted')
print('Accuracy (RBF Kernel): ', "%.2f" % (rbf_accuracy*100))
print('F1 (RBF Kernel): ', "%.2f" % (rbf_f1*100))
```

That code will print the following results:

```
Accuracy (Polynomial Kernel): 70.00
F1 (Polynomial Kernel): 69.67
Accuracy (RBF Kernel): 76.67
F1 (RBF Kernel): 76.36
```

3.6 Implementation

The proposed methodology of SVM (Support Vector Machine) classifier. The proposed approach is divided into three phase.

3.6.1 Data Pre-Processing:

Data pre-processing is deemed to be the main step in the data mining method and machine learning projects. In effective data collection can subsequently lead to improper combination, weak control and incorrect values. It moreover provides misleading results if the identification of the data is not carried out clearly. Therefore, it is essentially important to carry over quality and representation of data at the initial stage.

3.6.2 Improving Crop Productivity through a Crop Recommendation System Using Ensembling Technique

The ensembling technique is used to build a model that combines the predictions of multiple machine learning models together to recommend the right crop based on the soil specific type and characteristics with high accuracy. The independent base learners used in the ensemble model are Random Forest, Naive Bayes, and Linear SVM. Each classifier provides its own set of class labels with an acceptable accuracy. The class labels of individual base learners are combined using the majority voting technique. The crop recommendation system classifies the input soil dataset into the recommendable crop type, Kharif and Rabi. The dataset comprises of the soil specific physical and chemical characteristics in addition to the climatic conditions such as average rainfall and the surface temperature samples. The average classification accuracy obtained by combining the independent base learners is 99.91%.

A brief step by step procedure of designing the crop recommendation system is explained as follows:

Step 1: Input The input dataset is a comma separated values file containing the soil dataset, which has to be subjected to pre-processing.

Step 2: Pre-processing of input data Input dataset is subject to various pre-processing techniques such as filling of missing values, encoding of categorical data and scaling of values in the appropriate range

Step 3: Splitting into training and testing dataset the pre-processed dataset is then split into training and testing dataset based on the specified split ratio. The split ratio considered in the proposed work is 75:25, which means 75% of the dataset is used for the training the ensemble model and the rest 25% is used as test dataset.

Step 4: Building individual classifiers on the training dataset the training dataset is fed to each of the independent base learners and the individual classifiers are built using the training dataset.

Step 5: Testing the data on each of the classifiers the testing dataset is applied on each of the classifiers, and the individual class labels are obtained.

CHAPTER – IV

REQUIREMENTS

CHAPTER – IV
Requirements

Summary:

In this chapter the project requirements like tools, technologies, algorithms, and the methods used in this project is discussed in brief:

4.1 Hardware Requirements

Requirement	Minimum	Recommended
RAM	4 GB of free RAM	8 GB of total system RAM
CPU	Any modern CPU	Multi-core CPU. PyCharm supports multithreading for different operations and processes making it faster the more CPU cores it can use.
Disk space	3.5 GB	SSD drive with at least 5 GB of free space
Monitor resolution	1024×768	1920×1080
Operating system	Officially released 64-bit versions of the following: • Microsoft Windows 10 1809 or later • Windows Server 2019 or later • macOS 10.15 or later • Any Linux distribution that supports Gnome, KDE , or Unity DE. • PyCharm is not available for the Linux	Latest 64-bit version of Windows, macOS, or Linux (for example,

	distributions that do not include GLIBC 2.27 or later. Pre-release versions are not supported.	

4.2 Software Requirements:

4.2.1 Python

Python is commonly used for developing websites and software, task automation, data analysis, and data visualization. Since it’s relatively easy to learn, Python has been adopted by many non-programmers such as accountants and scientists, for a variety of everyday tasks, like organizing finances.

a) Python Features:

- Python is an interpreter-based language, which allows the execution of one instruction at a time.
- Extensive basic data types are supported e.g., numbers (floating point, complex, and unlimited-length long integers), strings (both ASCII and Unicode), lists, and dictionaries.
- Variables can be strongly typed as well as dynamic typed.
- Supports object-oriented programming concepts such as class, inheritance, objects, module, namespace etc.
- Cleaner exception handling support.
- Supports automatic memory management.
- Various built-in and third-party modules, which can be imported and used independently in the Python application.

b) Python Uses:

- **Data Science**

Today Python has become the language of choice for data scientists. Python libraries like NumPy, Pandas, and Matplotlib are extensively used in the process of data analysis, including the collection, processing and cleansing of data sets, applying mathematical algorithms, and generating visualizations for the benefit of users. Commercial and community Python distributions by third-parties such as Anaconda and ActiveState provide all the essential libraries required for data science.

- **Machine Learning**

This is another key application area of Python. Python libraries such as Scikit-learn, Tensorflow and NLTK are widely used for the prediction of trends like customer satisfaction, projected values of stocks, etc. Some of the real-world applications of machine learning include medical diagnosis, statistical arbitrage, basket analysis, sales prediction, etc.

- **Web Development**

This is another application area in which Python is becoming popular. Web application framework libraries like django, Pyramid, Flask, etc. make it very easy to develop and deploy simple as well as complex web applications. These frameworks are used extensively by various IT companies. Dropbox, for example, uses Django as a backend to store and synchronize local folders.

- **Image Processing**

The OpenCV library is commonly used for face detection and gesture recognition. OpenCV is a C++ library but has been ported to Python. Because of the rapid development of this feature, Python is a very popular choice from image processing.

- **Game Development**

Python is a popular choice for game developers. The PyGame library is extensively used for building games for desktop as well as for mobile platforms. PyGame applications can be installed on Android too.

- **Embedded Systems and IoT**

Another important area of Python application is in embedded systems. Raspberry Pi is a very popular yet low-cost single-board computer. It is extensively used in automation products, robotics, IoT, and kiosk applications. Popular microcontrollers like Arduino are used in many IoT products and are being programmed with Python. A lightweight version of Python called Micropython has been developed, especially for microcontrollers. A special Micropython-compatible controller called PyBoard has also been developed.

- **Android Apps**

Although Android apps are predominantly developed using Android SDK, which is similar to Java, Python can also be used to develop Android apps. Python's Kivy library has all the functionalities required for a mobile application.

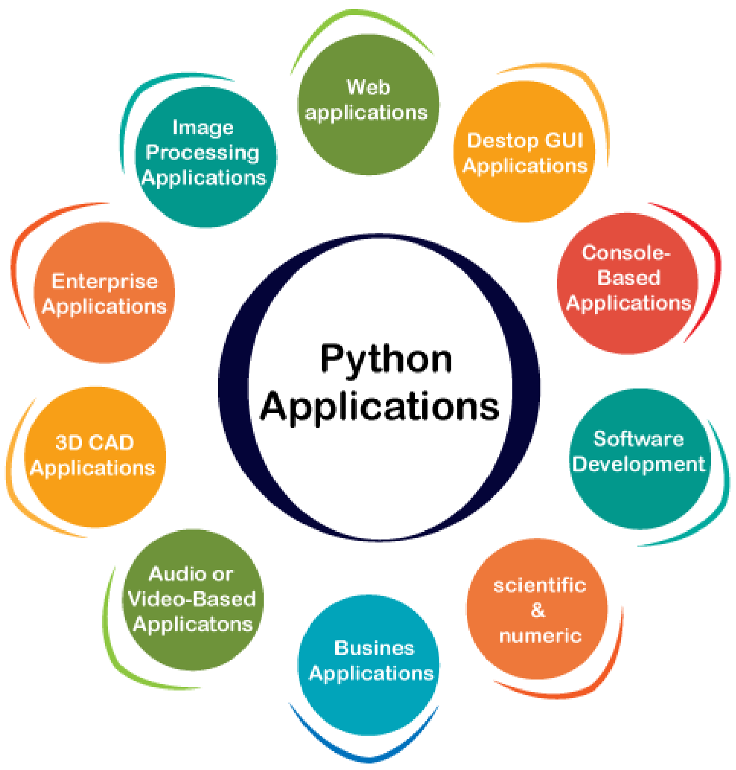


Figure 4.2.1.1 Python Programming Application

c) Python Tools and Frameworks

The following lists important tools and frameworks to develop different types of Python applications:

- Web Development: Django, Pyramid, Bottle, Tornado, Flask, web2py
- GUI Development: tkInter, PyGObject, PyQt, PySide, Kivy, wxPython
- Scientific and Numeric: SciPy, Pandas, IPython
- Software Development: Buildbot, Trac, Roundup

- System Administration: Ansible, Salt, OpenStack

4.2.2 Pycharm IDE:

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development. PyCharm is considered to be one of the most integrated Python IDEs, offering a range of modules and tools which make coding a lot faster and easier for programmers. One of the reasons for its popularity is the credentials of its developer, Jetbrains, a Czech who is renowned for creating some of the most popular Java and Javascript IDEs. It offers support for both Python 2 (2.7) and Python 3 (3.5 and above) versions and can be used on a multitude of platforms including Windows, Linux, and macOS.

PyCharm can be used for code analysis, debugging, and testing, among other things. It is particularly useful for web creation using web application frameworks like Django and Flask. Python plugins can be built by programmers using various APIs. It also allows programmers to access a range of databases without integrating with other tools. While designed specifically for programming with Python, it can also be used to create HTML, CSS, and Javascript files. It also comes with a great user interface that can be modified based on applications using plugins.

PyCharm is available in two editions: Professional, and Community. The Community edition is an open-source project, and it's free, but it has fewer features. The Professional edition is commercial, and provides an outstanding set of tools and features. For details, see the editions comparison matrix

a) PyCharm Features:

- **Smart Editor**

PyCharm offers a smart code editor which improves the readability of code using a variety of color schemes and error highlighting. It also has a smart code completion feature.

PyCharm provides support for a variety of integration tools which include:

- **Anaconda:** A distribution of Python geared towards scientific computing
- **IPython:** An interactive Python command terminal
- **Kite:** An AI-driven autocomplete plugin
- **Pylint:** A bug and quality checker
- **Pytest:** A test writing framework
- **WakaTime:** A dashboard with automatic time tracking and productivity metrics

PyCharm provides support for a variety of integration tools which include:

- **Anaconda:** A distribution of Python geared towards scientific computing
- **IPython:** An interactive Python command terminal
- **Kite:** An AI-driven autocomplete plugin
- **Pylint:** A bug and quality checker
- **Pytest:** A test writing framework
- **WakaTime:** A dashboard with automatic time tracking and productivity metrics

- **Data Science and Machine Learning**

PyCharm supports various libraries, including Matplotlib and SciPy, which help with data science and machine learning projects.

- **Google App Engine**

It is a cloud computing platform used for developing and hosting web applications. It even offers automatic scaling.

- **Integrated Debugging and Testing**

It comes with an Integrated Python debugger and integrated unit testing.

- **Web Development**

PyCharm supports popular web technologies like CoffeeScript, CSS, HTML, JavaScript, and TypeScript. By offering live editing, developers can modify a web page and push it live simultaneously to follow changes on a live browser.

- **Navigation**

Developers can debug the entire source further by inspecting it in lens mode. They can also use code navigation to locate various elements or variables instantly.

- **Remote Hosts and Virtual Machines**

PyCharm offers an integrated SHH terminal, docker and vagrant integration, and remote interpreters which help developers run, debug, test and deploy applications virtually.

- **Refactoring**

PyCharm's refactoring feature helps improve the structure of a program without negatively affecting its performance. Users can quickly and efficiently make changes to both local and global variables. They can also use the extract method to split extended classes and functions. It also has a variety of other features like introducing constants or variables, pull up, pull down, and rename.

- **Support for Popular Web Frameworks**

PyCharm lets programmers use popular Python web frameworks like Django, Flask, Pyramid and web2py in their Python projects.

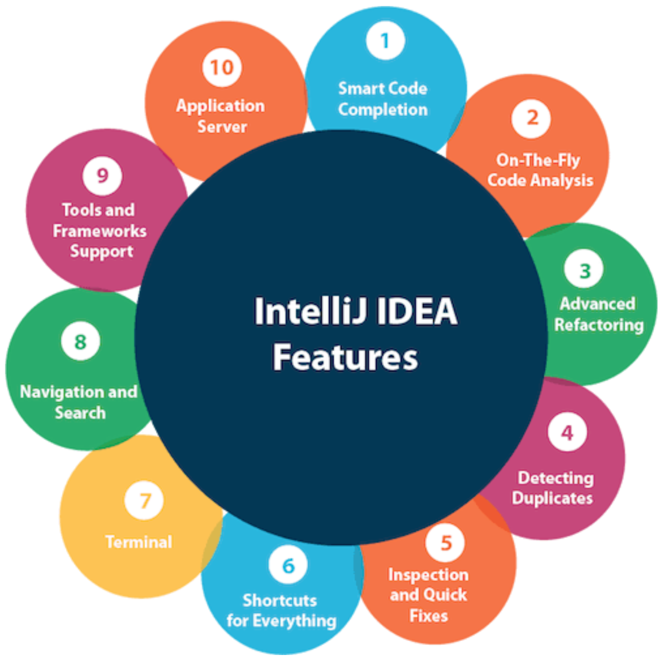


Figure 4.2.2.1 Features of Pycharm

4.3 Python Libraries

4.3.1 Tkinter

Tkinter is the de facto way in Python to create Graphical User interfaces (GUIs) and is included in all standard Python Distributions. In fact, it’s the only framework built into the Python standard library. This Python framework provides an interface to the Tk toolkit and works as a thin object-oriented layer on top of Tk. The Tk toolkit is a cross-platform collection of ‘graphical control elements’, aka widgets, for building application interfaces.

Uses of Tkinter:

- Displaying Text and Images with Label Widgets.
- Displaying Clickable Buttons with Button Widgets.
- Getting User Input with Entry Widgets.
- Getting Multiline User Input with Text Widgets.
- Assigning Widgets to Frames with Frame Widgets.
- Adjusting Frame Appearance with Reliefs.

4.3.2 OpenCV

OpenCV is an open-source software library for computer vision and machine learning. The OpenCV full form is Open Source Computer Vision Library. It was created to provide a shared infrastructure for applications for computer vision and to speed up the use of machine perception in consumer products. OpenCV, as a BSD-licensed software, makes it simple for companies to use and change the code. There are some predefined packages and libraries that make our life simple and OpenCV is one of them.

Gary Bradsky invented OpenCV in 1999 and soon the first release came in 2000. This library is based on optimised C / C++ and supports Java and Python along with C++ through an interface. The library has more than 2500 optimised algorithms, including an extensive collection of computer vision and machine learning algorithms, both classic and state-of-the-art.Using OpenCV it becomes easy to do complex tasks such as identify and recognise faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D object models, generate 3D point clouds from stereo cameras, stitch images together to generate an entire scene with a high resolution image and many more.

4.3.3 Numpy

NumPy (**Numerical Python**) is an open source Python library that’s used in almost every field of science and engineering. It’s the universal standard for working with numerical data in Python, and it’s at the core of the scientific Python and PyData ecosystems. NumPy users include everyone from beginning coders to experienced researchers doing state-of-the-art scientific and industrial research and development. The NumPy API is used extensively in Pandas, SciPy, Matplotlib, scikit-learn, scikit-image and most other data science and scientific Python packages.

The NumPy library contains multidimensional array and matrix data structures (you’ll find more information about this in later sections). It provides **ndarray**, a homogeneous n-dimensional array object, with methods to efficiently operate on it. NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

4.3.4 Pandas

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

- Fast and efficient for manipulating and analyzing data.
- Data from different file objects can be loaded.
- Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects

- Data set merging and joining.
- Flexible reshaping and pivoting of data sets
- Provides time-series functionality.
- Powerful group by functionality for performing split-apply-combine operations on data sets.

4.3.5 File dialogue

Python Tkinter (and TK) offer a set of dialogs that you can use when working with files. By using these you don’t have to design standard dialogs yourself. Example dialogs include an open file dialog, a save file dialog and many others. Besides file dialogs there are other standard dialogs, but in this article we will focus on file dialogs.

File dialogs help you open, save files or directories. This is the type of dialog you get when you click file,open. This dialog comes out of the module, there’s no need to write all the code manually.

Tkinter does not have a native looking file dialog, instead it has the customer tk style. You can see these below. The file dialog will work on all desktop platforms.

4.3.6 Scikit-learn

Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy.

In this tutorial we will learn to code python and apply Machine Learning with the help of the scikit-learn library, which was created to make doing machine learning in Python easier and more robust.

To do this, we’ll be using the Sales_Win_Loss data set from IBM’s Watson repository. We will import the data set using pandas, explore the data using pandas methods like head(), tail(), dtypes(), and then try our hand at using plotting techniques from Seaborn to visualize our data.

4.3.7 TensorFlow

You’ve probably heard of TensorFlow if you’re a machine learning student. It has become an industry norm and is one of the most common tools for machine learning and deep learning experts.

TensorFlow is a free and open-source library for creating machine learning models. It is a fantastic platform for everyone interested in working with machine learning and artificial intelligence.

This means that if you really want to work in the industry of Machine learning and artificial intelligence, you must be comfortable with this tool. If you’re thinking about what TensorFlow is and how it functions, you’ve arrived at the right because the following article will provide you with a complete description of this technology.

4.3.8 Keras

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result as fast as possible is key to doing good research.

Keras is:

- Simple -- but not simplistic. Keras reduces developer cognitive load to free you to focus on the parts of the problem that really matter.
- Flexible -- Keras adopts the principle of progressive disclosure of complexity: simple workflows should be quick and easy, while arbitrarily advanced workflows should be possible via a clear path that builds upon what you've already learned.
- Powerful -- Keras provides industry-strength performance and scalability: it is used by organizations and companies including NASA, YouTube, or Waymo.

4.4 Algorithms Used

4.4.1 Convolutional Neural Networks (CNN):

A convolutional neural network (CNN or convnet) is a subset of machine learning. It is one of the various types of artificial neural networks which are used for different applications and data types. A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data.

There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice. This makes them highly suitable for computer vision (CV) tasks and for applications where object recognition is vital, such as self-driving cars and facial recognition.

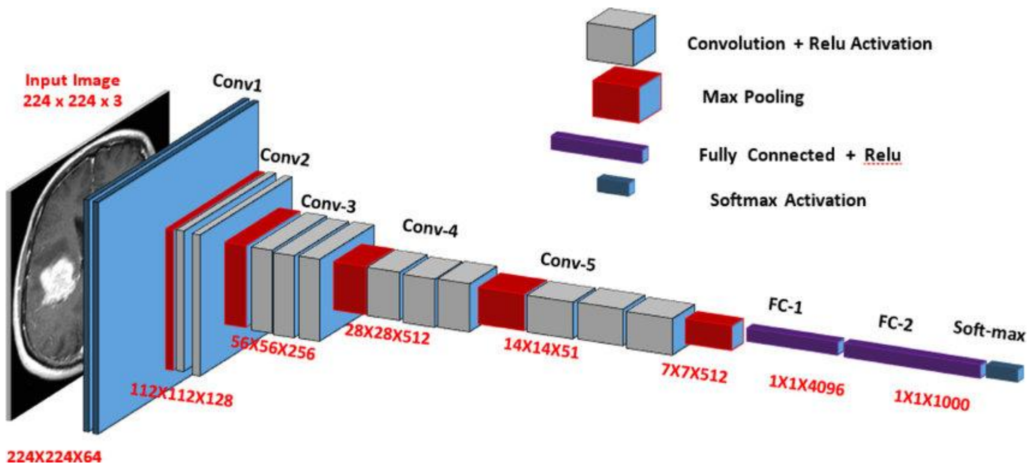


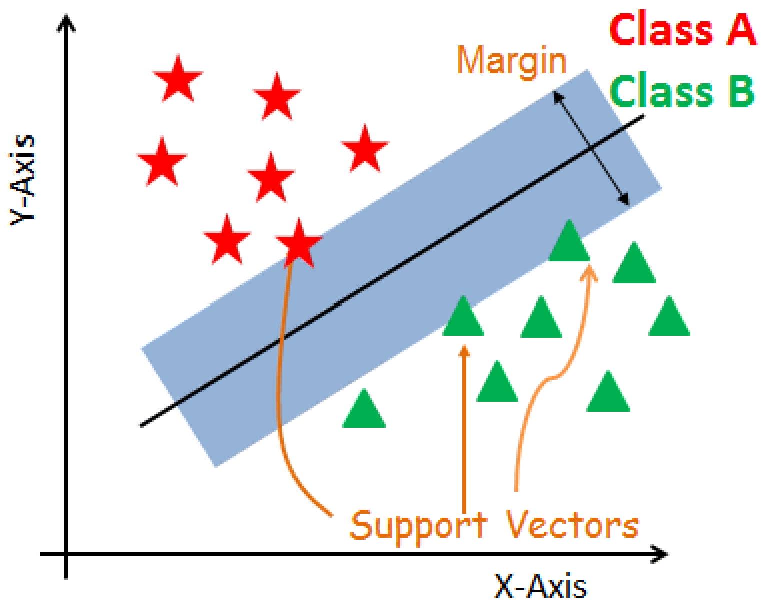
Figure 4.4.1 Working of CNN model

- CNN model required a vast number of images for training, which is often difficult to obtain in the medical imaging field.
- Convolutional Neural Networks (CNN) perform remarkably well at classifying images that are quite similar to the dataset. CNNs, on the other hand, struggle to classify images that have a slight tilt or rotation.
- This can be fixed by utilizing data augmentation to continuously introduce new variants to the image during training. To address this problem in our research, we employed the data augmentation technique.

4.4.2 Support Vector Machine (SVM)

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they're able to categorize new text.

Compared to newer algorithms like neural networks, they have two main advantages: higher speed and better performance with a limited number of samples in the thousands. This makes the algorithm very suitable for text classification problems, where it's common to have access to a dataset of at most a couple of thousands of tagged samples.



4.4.2 Support Vector Machine

CHAPTER – V

RESULT AND DISCUSSION

CHAPTER – V

Result and Discussion

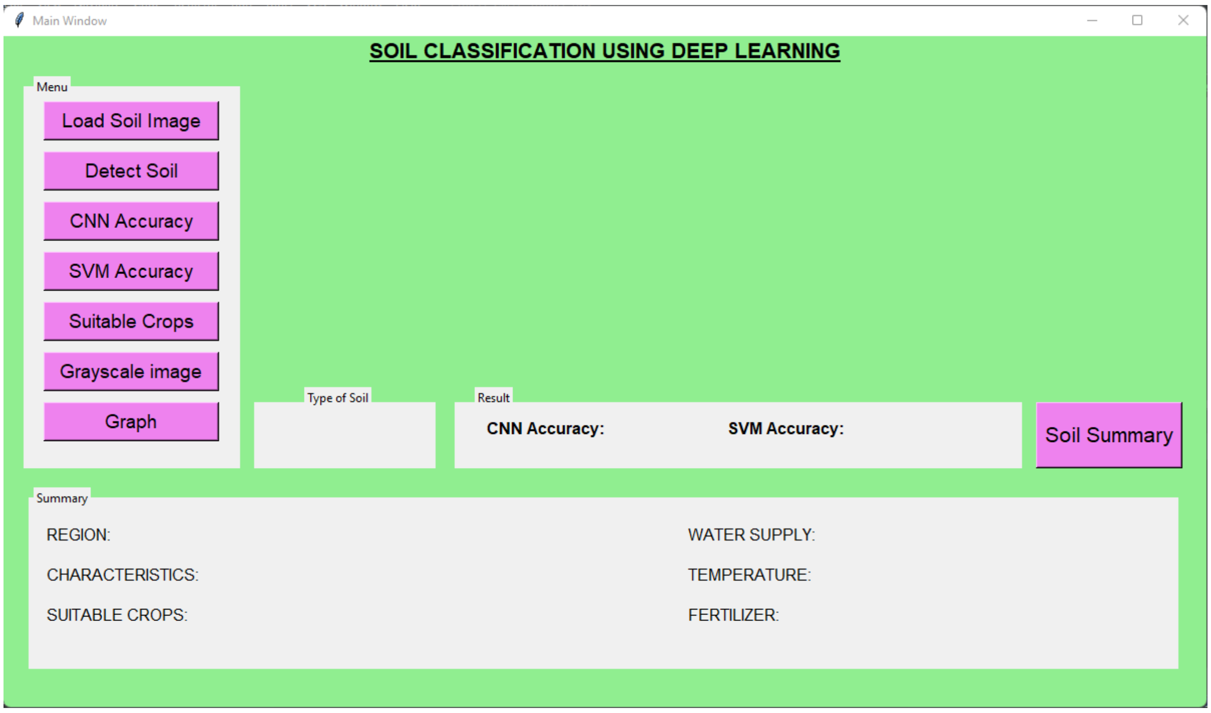


Figure 5.1 Soil Classification and Crop Suggestion Module GUI

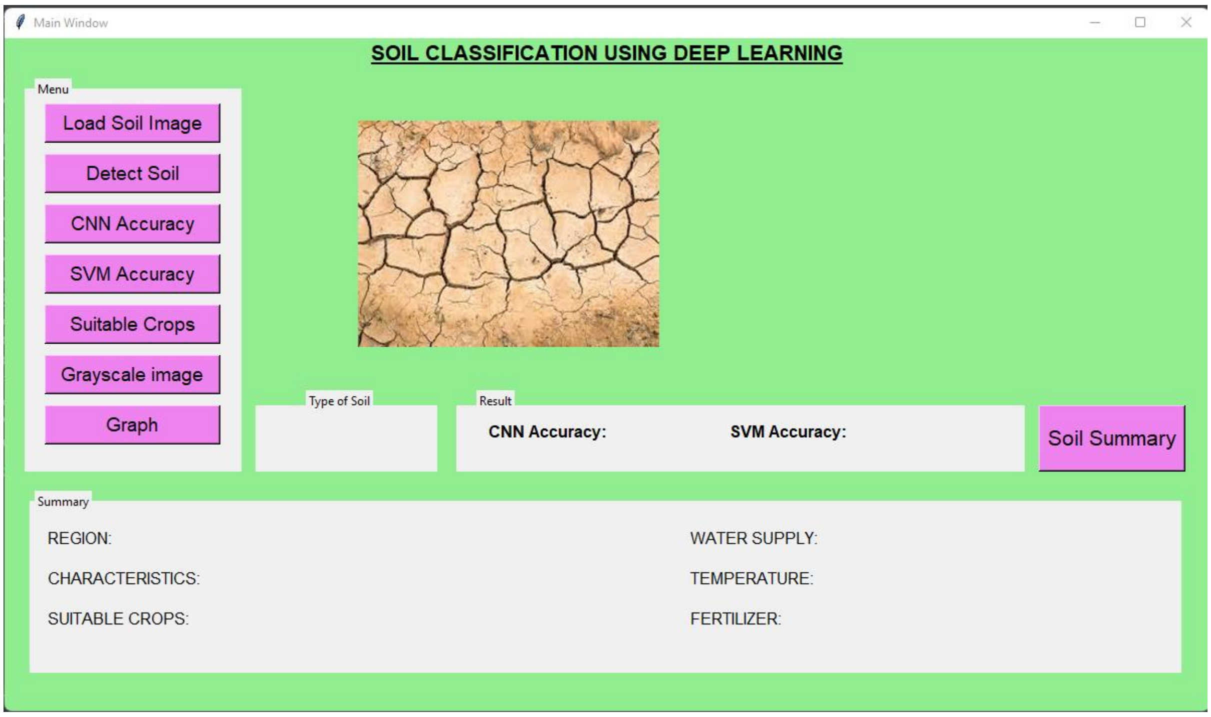
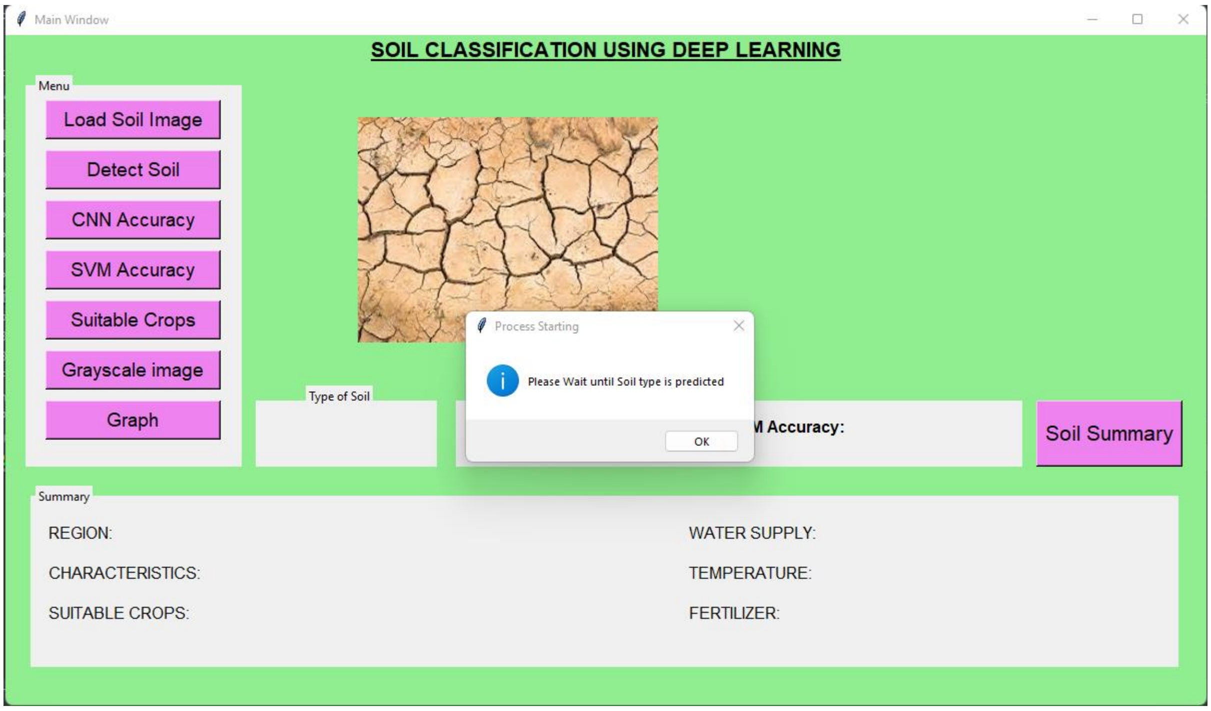


Figure 5.2 Input Soil Image



Please Wait until Soil type is predicted

Figure 5.3 Soil Classification in Progress

CHAPTER – VI

CONCLUSION AND FUTURE SCOPE

CHAPTER – VI
Conclusion and Future Scope

6.1 Conclusion

As a conclusion we came to the final discussion that Soil classification system is essential for the identification of soil properties. Expert system can be a very powerful tool in identifying soils quickly and accurately .Traditional classification systems include use of tables, flow-charts. This type of manual approach takes a lot of time, hence quick, reliable automated system for soil classification is needed to make better utilization of technician's time. We propose an automated system that has been developed for classifying soils based on fertility. A model is proposed for predicting soil series and providing suitable crop yield suggestion for that specific soil. The research has been done on soil datasets of different soil images. The model has been tested by applying different kinds of machine learning algorithm. Bagged tree and K-NN shows good accuracy but among all the classifiers, CNN and SVM has given the highest accuracy in soil classification. The proposed model is justified by a properly made dataset and machine learning algorithms. The soil classification accuracy and also the recommendation of crops for specific soil are more appropriate than many existing methods. Providing fertilizer recommendation is our concern, In future, we contrive to build Fertilizer Recommendation System which can be utilized effectively by the Soil Testing Laboratories. This System will recommend appropriate fertilizer for the given soil sample and cropping pattern.

6.2 Future Scope

Also in future data of other districts will be added to make this model more reliable and accurate. We will try to make the current system available as an android app so that it can be used by each and every person anytime. Also to increase features like real time image capturing and result finding with full time support for the farmers and give them best help with their problems.

REFERENCES

[1] Y. L. a. B. X. J. Gong, “Spatial heterogeneity of urban land-cover landscape in Guangzhou from 1990 to 2005,” *Journal of Geographical Sciences,* Vols. vol. 19, no. 2, p. pp. 213–224, 2019.

[2] S. H. a. E. M. D. Lu, “Land cover classification in a complex urban-rural landscape with QuickBird imagery,” *Photogrammetric Engineering & Remote Sensing,* Vols. vol. 76, no. 10, p. pp. 1159–1168, 2016.

[3] M. C. A. X. Z. e. a. F. Gao, “Toward mapping crop progress at field scales through fusion of Landsat and MODIS imagery,” *Remote Sensing of Environment,* vol. vol. 188, p. pp. 9–25, 2017.

[4] X. X. M. A. M. e. a. J. Dong, “Mapping paddy rice planting area in northeastern Asia with Landsat 8 images, phenology-based algorithm and google earth engine,” *Remote Sensing of Environment,* vol. vol. 185, p. pp. 142–154, 2016.

[5] F. M. N. N. K. Q. a. H. K.-S. N. Noman, “Determining optimal feature-combination for LDA classification of functional near-infrared spectroscopy signals in brain-computer interface application,” *Frontiers in Human Neuroscience,* vol. vol. 10, p. p. 237, 2016.

[6] H. S. S. C. e. a. S. Liu, “Estimating forest soil organic carbon content using vis-NIR spectroscopy: implications for large-scale soil carbon spectroscopic assessment,” *Geoderma,* vol. vol. 348, p. pp. 37–44, 2019.

[7] C. H. M. K. P. M. M. H. G. a. L. W. D. J. M. Paradelo, “Field-scale predictions of soil contaminant sorption using visible-near infrared spectroscopy,” *Journal of Near Infrared Spectroscopy,* Vols. vol. 24, no. 3, p. pp. 281–291, 2016.

[8] D. Cozzolino, “Near infrared spectroscopy as a tool to monitor contaminants in soil, sediments and water-state of the art, advantages and pitfalls,” *Trends in Environmental Analytical Chemistry,* Vols. vol. 9, no. 2, p. pp. 1–7, 2016.

[9] Y. F. H. a. M. H. K. A. M. Hassan, “A deep classification system for medical data analysis,” *Journal of Medical Imaging and Health Informatics,* Vols. vol. 8, no. 2, p. pp. 250–256, 2018.

[10] J. Z. a. O. G. Troyanskaya, “Predicting effects of noncoding variants with deep learning-based sequence model,” *Nature Methods,* Vols. vol. 12, no. 10, p. pp. 931–934, 2015.

[11] L. X. a. F. H. J. Gao, “A spectral-textural kernel-based classification method of remotely sensed images,” *Neural Computing & Applications,* Vols. vol. 27, no. 2, pp. pp. 431–446,, 2016.

[12] Y. Y. T. B. Z. e. a. J. Xu, “The generalization ability of SVM classification based on markov sampling,” *IEEE Transactions on Cybernetics,* Vols. vol. 45, no. 6, p. pp. 1169–1179, 2015.

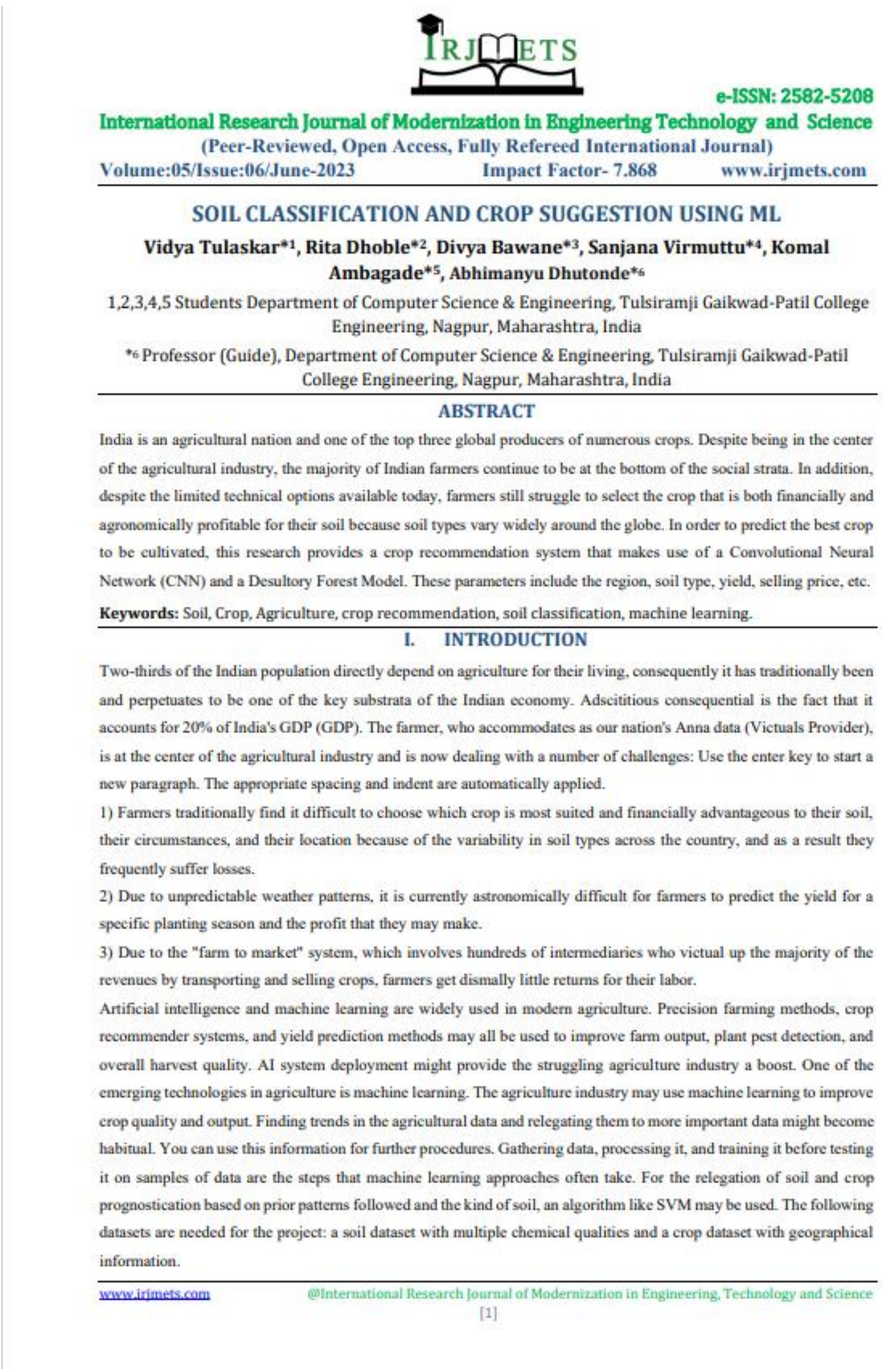
[13] G. K. M. K. C. C. U. K. a. T. P. T. Serre, “A quantitative theory of immediate visual recognition,” *Progress in Brain Research,* Vols. vol. 165, no. 6, p. pp. 33–56, 2017.

[14] H. Z. a. Q. S. Y. Li, “Spectral-spatial classification of hyperspectral imagery with 3D convolutional neural network,” *Remote Sensing,* Vols. vol. 9, no. 1, p. p. 67, 2017.

[15] K. P. a. R. L. J. Steinbrener, “Hyperspectral fruit and vegetable classification using convolutional neural networks,” *Computers and Electronics in Agriculture,* vol. vol. 162, p. pp. 364–372, 2019.

[16] R. M. J. a. H. I. K. M. Alencastre-Miranda, “Convolutional neural networks and transfer learning for quality inspection of different sugarcane varieties,” *IEEE Transactions on Industrial Informatics,* Vols. vol. 17, no. 2, p. pp. 787–794, 2021.

Publication paper



e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

SOIL CLASSIFICATION AND CROP SUGGESTION USING ML

Vidya Tulaskar*[1], Rita Dhoble*[2], Divya Bawane*[3], Sanjana Virmuttu*[4], Komal Ambagade*[5], Abhimanyu Dhutonde*[6]

1,2,3,4,5 Students Department of Computer Science & Engineering, Tulsiramji Gaikwad-Patil College Engineering, Nagpur, Maharashtra, India

*6 Professor (Guide), Department of Computer Science & Engineering, Tulsiramji Gaikwad-Patil College Engineering, Nagpur, Maharashtra, India

ABSTRACT

India is an agricultural nation and one of the top three global producers of numerous crops. Despite being in the center of the agricultural industry, the majority of Indian farmers continue to be at the bottom of the social strata. In addition, despite the limited technical options available today, farmers still struggle to select the crop that is both financially and agronomically profitable for their soil because soil types vary widely around the globe. In order to predict the best crop to be cultivated, this research provides a crop recommendation system that makes use of a Convolutional Neural Network (CNN) and a Desultory Forest Model. These parameters include the region, soil type, yield, selling price, etc.

Keywords: Soil, Crop, Agriculture, crop recommendation, soil classification, machine learning.

I. INTRODUCTION

Two-thirds of the Indian population directly depend on agriculture for their living, consequently it has traditionally been and perpetuates to be one of the key substrata of the Indian economy. Adscititious consequential is the fact that it accounts for 20% of India's GDP (GDP). The farmer, who accommodates as our nation's Anna data (Victuals Provider), is at the center of the agricultural industry and is now dealing with a number of challenges: Use the enter key to start a new paragraph. The appropriate spacing and indent are automatically applied.

1) Farmers traditionally find it difficult to choose which crop is most suited and financially advantageous to their soil, their circumstances, and their location because of the variability in soil types across the country, and as a result they frequently suffer losses.

2) Due to unpredictable weather patterns, it is currently astronomically difficult for farmers to predict the yield for a specific planting season and the profit that they may make.

3) Due to the "farm to market" system, which involves hundreds of intermediaries who victual up the majority of the revenues by transporting and selling crops, farmers get dismally little returns for their labor.

Artificial intelligence and machine learning are widely used in modern agriculture. Precision farming methods, crop recommender systems, and yield prediction methods may all be used to improve farm output, plant pest detection, and overall harvest quality. AI system deployment might provide the struggling agriculture industry a boost. One of the emerging technologies in agriculture is machine learning. The agriculture industry may use machine learning to improve crop quality and output. Finding trends in the agricultural data and relegating them to more important data might become habitual. You can use this information for further procedures. Gathering data, processing it, and training it before testing it on samples of data are the steps that machine learning approaches often take. For the relegation of soil and crop prognostication based on prior patterns followed and the kind of soil, an algorithm like SVM may be used. The following datasets are needed for the project: a soil dataset with multiple chemical qualities and a crop dataset with geographical information.

Greema S Raj et al. [1] in this paper, the author feels that thescientists who are working on different land areas and relegating different soil types should have a prevalent language to ken about that particular soil. They opted for machine learning and some of the well-kenned algorithms fortheir research work. There are a variety of soil present in theworld having different characteristics and features and different crops can be grown on them. For the relegation of the soil Decision Tree, Neural networks, Naïve Bayes, and SVM techniques are utilized for comparative study purposes.

Mohammad Diqi et al. [2] this work is all about relegating thesoil type ebony and red, and for that purport, authors have utilized CNN and deep learning algorithms to relegate the resulting output. In this process, they got a prosperity rate of 97% and a loss of 0.1606.

Pallavi Srivastava et al. [3] in two streams, this study discussesseveral computer-based soil categorization techniques. The first kind of methodology uses image processing and computer vision to categorise soil using various parameters including texture, colour, and particle size, as well as more traditional image processing algorithms and techniques. Thesecond category includes soil categorization methods based on deep learning and machine learning, such CNN, which produce cutting-edge outcomes. By streamlining the entire process, deep learning applications primarily reduce the reliance on spatial-form designs and preprocessing methods.

Shravani V et al. [4] the goal of the research is to develop a model that effectively categories soil instances and maps the soil type to crop data to produce better predictions with higher accuracy. Crop classifications and regional characteristics areboth used in soil prediction. In order to anticipate crops moreaccurately, it also seeks to develop a system that analyses real-time soil data. There are two phases to the model: the training phase and the testing phase. Soil and crop databasesare the two utilized datasets. The list of the proper classes is generated after comparing the expected and actual classes.

S. A. Z. Rahman et al. [5] in this proposed work, authors provide a model that predicts soil series with regard to land type and, in accordance with prediction, suggests appropriatecrops. For soil classification, a number of machine learning techniques are utilised, including weighted k-Nearest Neighbor (k-NN), bagged trees, and Support Vector Machines (SVM) based on a Gaussian kernel. The suggestedSVM-based technique outperforms several current methods,according to experimental data.

II. METHODOLOGY

1)k-nearest neighbors:

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

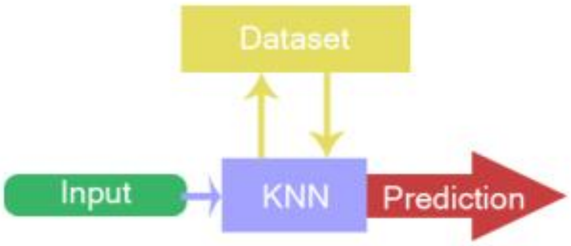


Figure 2.1: System Architecture of KNN

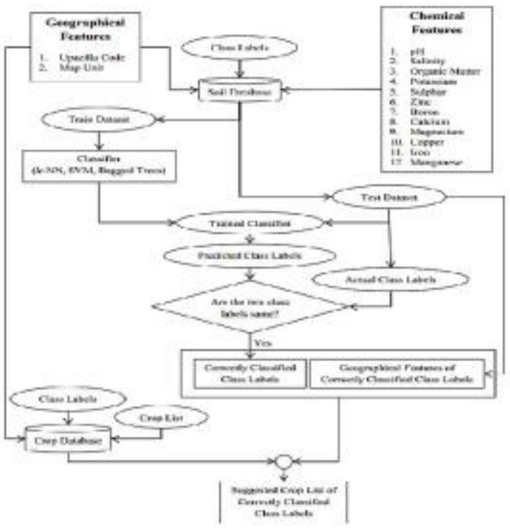


Figure 2.2: Proposed System Flow

i. Dataset Collection:

Images of the many soil types, including Red Soil, Black Soil, Clay Soil, and Alluvial Soil, are included in the dataset used to train the parameters of thismethod. Each kind of soil is represented by 150–200 photographs in the training set and 50–60 images in the test set of the dataset. This information is gathered from relevantinternet sources, including the Soil Classification Image Dataset on Kaggle.

ii. Data Pre-Processing and Algorithm Implementation:

Since the photos in the dataset are different shapes and sizes,they must first be pre-processed and scaled before being fed into the model. For our use case, the algorithm must read a colored picture, and every colored image in RGB format hasthree channels—one for each of the three colors—Red, Green, and Blue. Thus, 300x300x3 pictures are initially created from the original photographs. Then, in order for themodel to handle the data, the pictures are transformed into numerical (pixel intensities) arrays. The machine learning model is then given this numerical data.

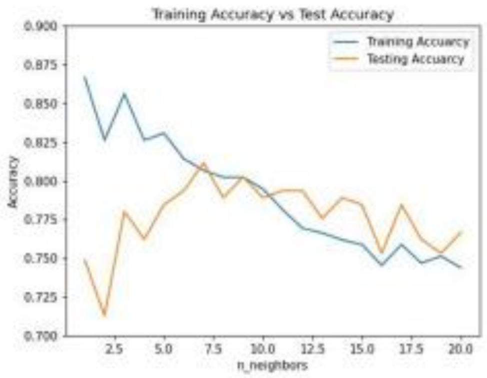


Figure 2.3: Graph of Accuracy of KNN vs No. of Training

With the learning rate set for 20 epochs and the AdamOptimization Algorithm, this network was trained. On the test set, the model had an accuracy of 95.21 percent. The Accuracy and Loss graphs in Figure show the training outcomes for this model.

2) Support Vector Machine (SVM):

The photos are initially downsized to 200 × 200 x 3 images using this method. The pictures are next transformed into numerical (pixel intensities) arrays, and finally flattened to afeature vector of 120000 by 1. The SVM model is then giventhis feature vector. Two kernels, the linear and the Radial Basis Function (RBF)Kernel, were used to train the SVM model. On the test set, the accuracy of the models with linear and RBF kernels was 83.5% and 86.7%, respectively.

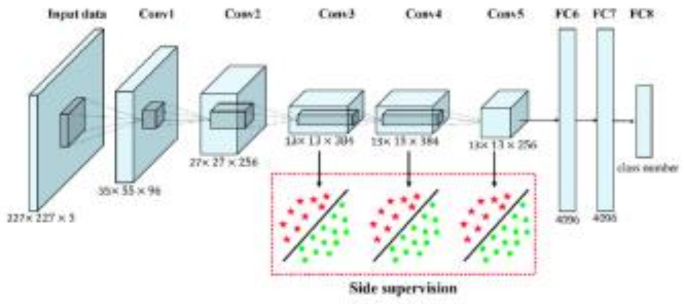


Figure 2.4: SVM Architecture

AlexNet: The Convolutional Neural Network Algorithm usesthis traditional network design. The photographs were initially downsized to 227 × 227 x 3 images in order toapproximate the proportions for this design. The CNN modelis then given the numerical data from the arrays of pixelintensities that were created from the transformed pictures. Since we only have 4 classes, we only have 4 units rather than1000 units. As a result, there are around 59 million trainableparameters in the network. The Adam OptimizationAlgorithm was used to train the model, using a learning rateof 10-3 and an epoch size of 25. On the test set, thisarchitecture provided us an accuracy of 25%.

3)Convolutional Neural Networks with Other/New Architectures:

We chose to test several CNN architectures after receiving satisfactory results from the SVM model and dismal ones from SVM. The photos are initially downsized to 300x300x3for all the ensuing architectures. The CNN model is then supplied with the numerical (pixel intensities) arrays createdfrom the pictures. The Adam optimization technique is used to train the model, with a learning rate of all architectures anda total of 25 training epochs. A brief step by step procedure of designing the crop recommendation system is explained as follows:

Step 1: The soil dataset, which must go through pre- processing, is included in a comma separated values file.

Step 2: Pre-processing methods used on the input dataset include filling in missing values, encoding categorical data, and scaling values to the right range.

Step 3: Using the given split ratio, the pre-processed datasetis then divided into a training and testing dataset. The suggested work uses a split ratio of 75:25, meaning that 75% of the dataset is utilized to train the ensemble model and the remaining 25% is used as a test dataset.

Step 4: Individual classifiers are built using the training dataset. Each independent base learner receives the training dataset, and each classifier is created using each of them.

Step 5: Analyzing the data with each classifier the individualclass labels are acquired when each classifier applies the testing dataset.

Step 6: Assembling the individual classifier output using Majority Voting Technique the final result of soil classification and crop suggestion can be achieved.

III. MODELING AND ANALYSIS

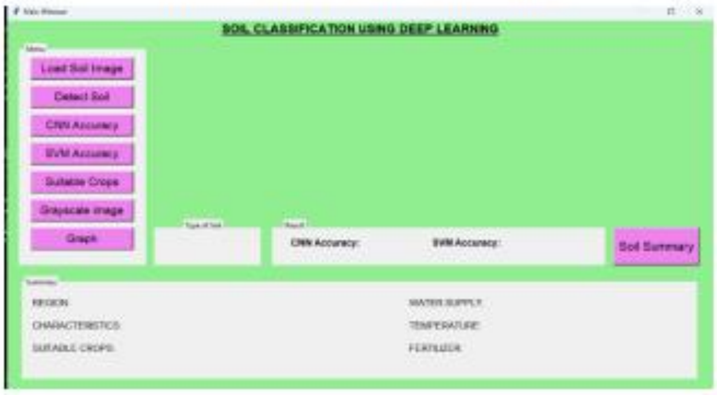
Python: Python is a trending technology these days and provides multiple ways to use it on daily basis. It can be integrated with many languages like machine learning, data science, data analytics, Internet of Things, Artificial Intelligence, etc. It is an interpreted language and hence easyfor the beginners to understand and work on. In this project python is used as a core language.

Pycharm IDE: It is an Integrated Development Environmentcommonly known as IDE work on python programs and codes. This software allows user to implement their ideas andget the desired output by integrating anaconda navigator environment in it.

Anaconda Navigator: It is a desktop GUI (Graphical User Interface) application to work with jupyter notebook, launchapplications, providing conda environments, without using CLI (Command Line Interface). It has feature of collecting different python libraries.

Gaussian Kernel based SVM: SVM divides class objects into several decision planes. The difference between an itembelonging to one class and another is determined by a decision boundary. The data points that are closest to the hyper-plane are known as support vectors. The inputs are transformed into a higher dimensional space by the kernel function, which then separates the nonlinear data. They employed the Gaussian kernel function in their research. SVM was used, and the accuracy was 95.99%.

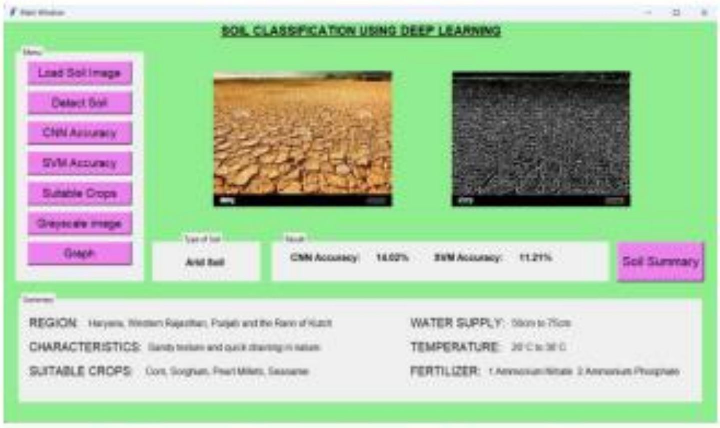
IV. RESULTS AND DISCUSSION



CNN Accuracy:
SVM Accuracy:
REGION:
CHARACTERISTICS:
SUITABLE CROPS:
WATER SUPPLY:
TEMPERATURE:
FERTILIZER:

Figure 4.1: GUI of the proposed system

In the above given Figure 4.1 image, a GUI based on Python tkinter library is shown where we can see a dashboard with multiple buttons and some group of labels expected to show their final working with the correspondence of background task implemented. Each button in this GUI is having the particular task assigned for e.g. first button has the working of uploading the image from the browser by the user and to load it in the GUI for further processing. All the buttons likely to accessed according to the user preferences.



Arid Soil
CNN Accuracy: 14.02% SVM Accuracy: 11.21%
REGION: Haryana, Western Rajasthan, Punjab and the Rann of Kutch
CHARACTERISTICS: Sandy texture and quick draining in nature
SUITABLE CROPS: Corn, Sorghum, Pearl Millets, Seasame
WATER SUPPLY: 50cm to 75cm
TEMPERATURE: 20 C to 30 C
FERTILIZER: 1.Ammonium Nitrate 2.Ammonium Phosphate

Figure 4.2: Overall working of the proposed system

As expected by the project objective the shown Figure 4.2 displays the overall working of the project with the soil classification details displayed on the label underneath original image uploaded by the user, the outputs like model accuracy with percentage represented just next to it. All the other parameters like how much the land need water, humidity, temperature, fertilizers, suitable crops to be grown are suggested for the betterment of the farmers wealth as well as to expect more crops in future.

Step 1: Using convolutional neural networks to classify soil We process the user-provided photograph of the soil in the first stage and categories it into one of the four classificationsof soil—Red, Alluvial, Black, and Clay. This is accomplished using a convolutional neural network, the specifics of whoseimplementation are covered in the following subsections. Several crops that can be grown in that soil type areshortlisted when the soil type is forecasted. As a result,because only crops suited to the soil type are shortlisted, thisphase assures the quality of the crops advised. Training phase and testing phase were the two steps of the approach. The soil dataset and the crop dataset were both used. Chemical characteristics of soil are included in the soildataset as classes. The soil class was determined using machine learning techniques. K-NN, Gaussian Kernel based SVM.

Step 2: Yield and Income Prediction

Our algorithm takes into account characteristics such soil type (predicted by the first step), area of land to be farmed (in hectares), State, District, crop, and season of cultivation in the second phase and forecasts the yield of all crops that were shortlisted based on the parameters mentioned above (in quintals). The Random Forest Algorithm, whose implementation details are covered in the following subsections, accomplishes this.

V. CONCLUSION

A model is proposed for predicting soil series and providing suitable crop yield suggestion for that specific soil. The model will be tested by applying CNN, SVM algorithms to get the minimum required accuracy. At the end the conclusion for this project is to facilitate the farmers with the latest upgrading technologies and to let them satisfy their own need of growing multiple crops simultaneously varying with the soil type we have in different fields. We have also specified the fertilizers and the natural parameters to be maintained while producing the particular crops on the particular soil type. We have got maximum accuracy through CNN algorithm and hence the core algorithm for this model stays with CNN itself.

VI. REFERENCES

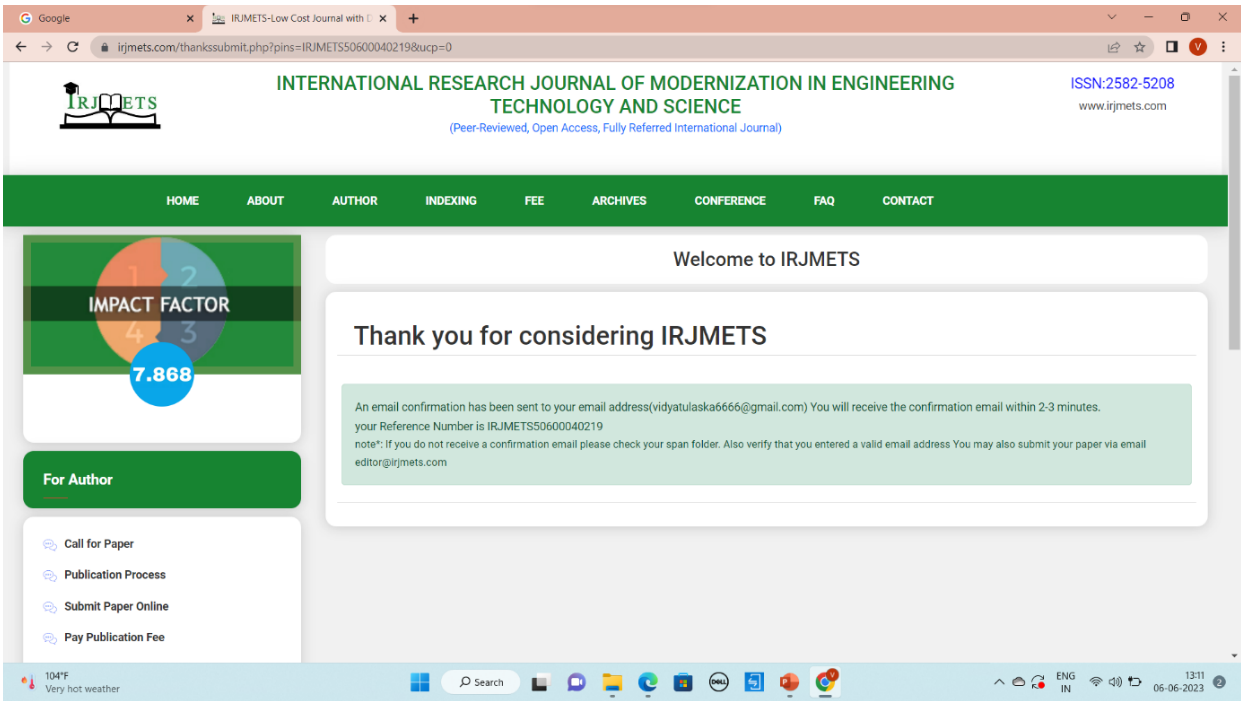
[1] Greema S Raj, Lijin Das S, "Survey On Soil Classification Using Different Techniques", Volume: 07 Issue: 03, Mar 2020, p-ISSN: 2395-0072 International Research Journal of Engineering and Technology (IRJET).

[2] Hamzah, Mohammad Diqi, Antomy David Ronaldo, "Effective Soil Type Classification Using Convolutional Neural Network", Vol. 3, No.1, Agustus 2021 ISSN: 2685-8711, E-ISSN: 2714-5263 DOI : 10.35842/ijicom

[3] Pallavi Srivas tava, Aasheesh Shukla, Atul Bansa, "A comprehensive review on soil classification usingdeep learning and computer vision", DOI:10.1007/s11042- 021-10544-5

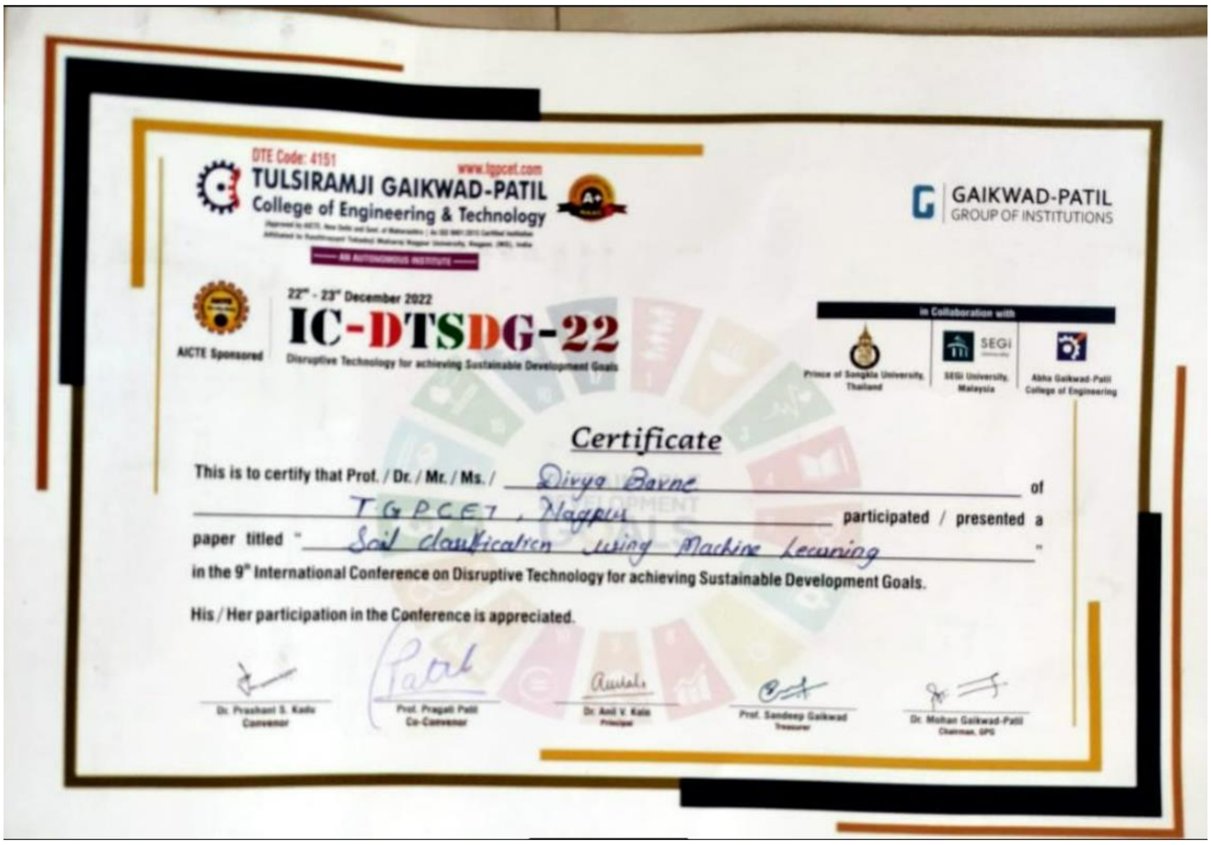
[4] Shravani V, Uday Kiran, Yashaswini J, and Priyanka D, "Soil Classification And Crop Suggestion Using Machine Learning", Volume: 07 Issue: 06, June 2020, p- ISSN: 2395-0072, International Research Journal of Engineering and Technology (IRJET)

[5] S. A. Z. Rahman, K. Chandra Mitra and S. M. Mohidul Islam, "Soil Classification Using Machine Learning Methods and Crop Suggestion Based on Soil Series," 2018 21st International Conference of Computer and Information Technology (ICCIT), 2018, pp. 1-4, doi: 10.1109/ICCITECHN.2018.8631943.


irjmets.com/thankssubmit.php?pins=IRJMETS50600040219&ucp=0
INTERNATIONAL RESEARCH JOURNAL OF MODERNIZATION IN ENGINEERING TECHNOLOGY AND SCIENCE
(Peer-Reviewed, Open Access, Fully Referred International Journal)
ISSN:2582-5208
www.irjmets.com
HOME
ABOUT
AUTHOR
INDEXING
FEE
ARCHIVES
CONFERENCE
FAQ
CONTACT
IMPACT FACTOR
7.868
For Author
Call for Paper
Publication Process
Submit Paper Online
Pay Publication Fee
Welcome to IRJMETS

Thank you for considering IRJMETS

An email confirmation has been sent to your email address(vidyatulaska6666@gmail.com) You will receive the confirmation email within 2-3 minutes.
your Reference Number is IRJMETS50600040219
note*: If you do not receive a confirmation email please check your span folder. Also verify that you entered a valid email address You may also submit your paper via email editor@irjmets.com



DTE Code: 4151

www.tgpcet.com

TULSIRAMJI GAIKWAD-PATIL
College of Engineering & Technology

Approved by AICTE, New Delhi and Govt. of Maharashtra | An ISO 9001:2015 Certified Institution
Affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur (MS), India

— AN AUTONOMOUS INSTITUTE —

GAIKWAD-PATIL
GROUP OF INSTITUTIONS

AICTE Sponsored

22nd - 23rd December 2022

IC-DTSDG-22

Disruptive Technology for achieving Sustainable Development Goals

in Collaboration with

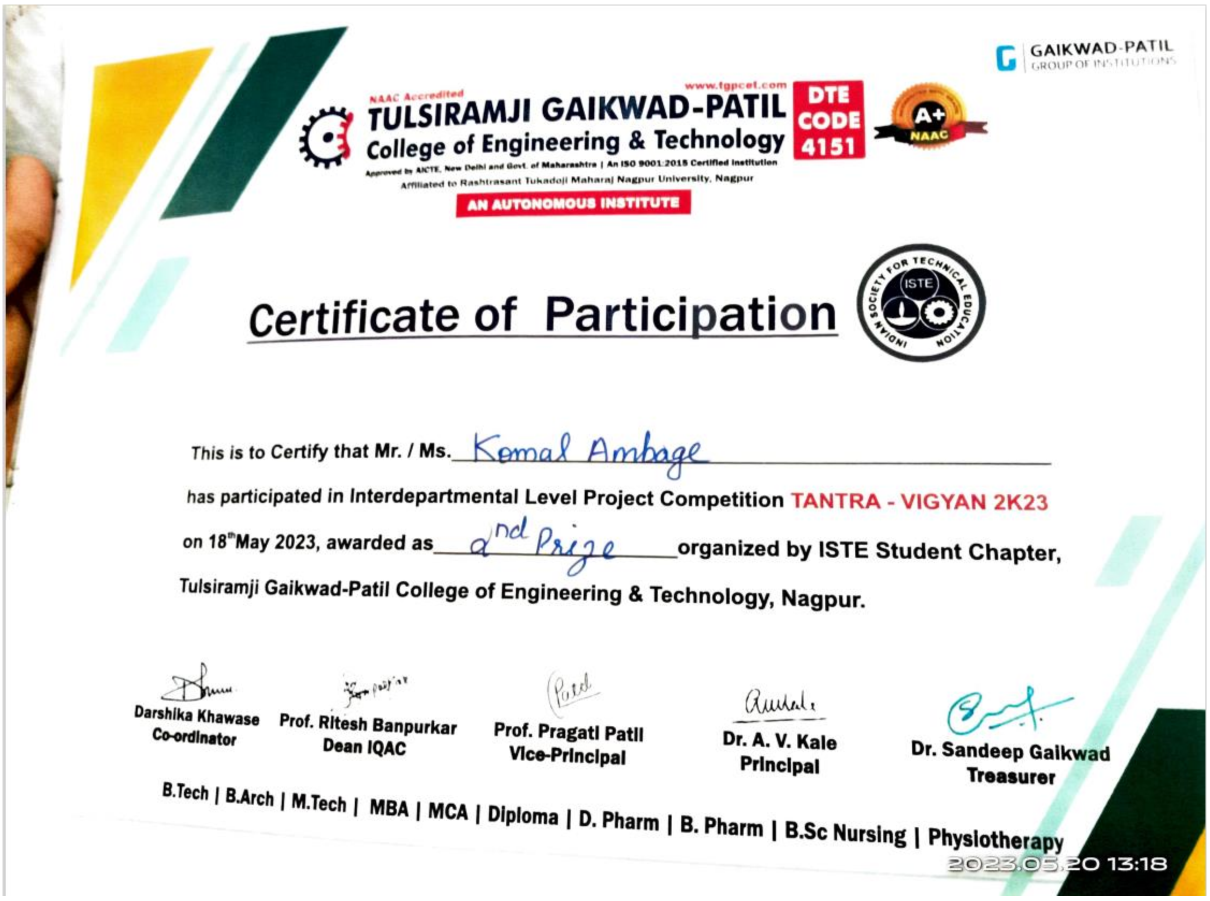
Prince of Songkla University, Thailand | SEGi University, Malaysia | Abha Gaikwad-Patil College of Engineering

Certificate

This is to certify that Prof. / Dr. / Mr. / Ms. / Divya Bavne of TGPCET, Nagpur participated / presented a paper titled " Soil classification using Machine Learning " in the 9th International Conference on Disruptive Technology for achieving Sustainable Development Goals.

His / Her participation in the Conference is appreciated.

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Dr. Prashant S. Kadu Convenor	Prof. Pragati Patil Co-Convenor	Dr. Anil V. Kale Principal	Prof. Sandeep Gaikwad Treasurer	Dr. Mohan Gaikwad-Patil Chairman, GPG



GAIKWAD-PATIL
GROUP OF INSTITUTIONS

NAAC Accredited

www.tgpcet.com

TULSIRAMJI GAIKWAD-PATIL
College of Engineering & Technology

DTE CODE 4151

A+ NAAC

Approved by AICTE, New Delhi and Govt. of Maharashtra | An ISO 9001:2015 Certified Institution
Affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur

AN AUTONOMOUS INSTITUTE

INDIAN SOCIETY FOR TECHNICAL EDUCATION
ISTE

Certificate of Participation

This is to Certify that Mr. / Ms. Komal Ambage

has participated in Interdepartmental Level Project Competition TANTRA - VIGYAN 2K23

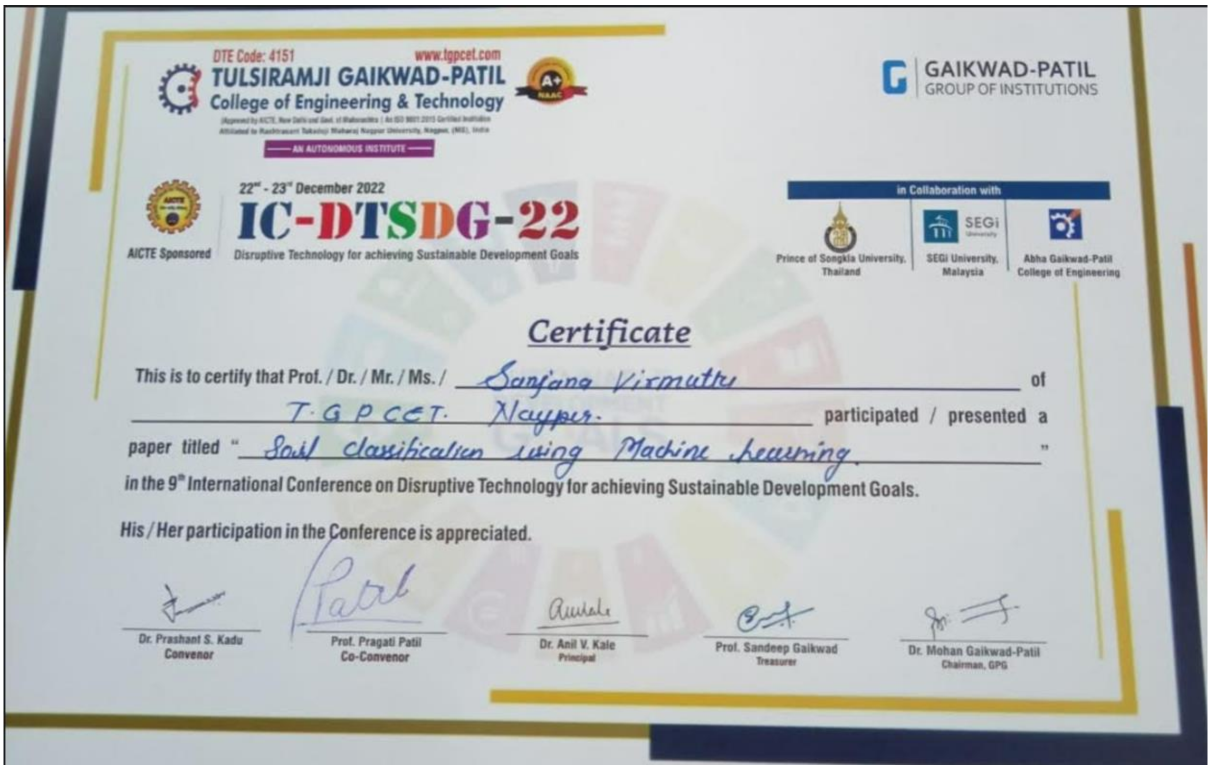
on 18th May 2023, awarded as 2nd Prize organized by ISTE Student Chapter,

Tulsiramji Gaikwad-Patil College of Engineering & Technology, Nagpur.

[illegible]	[illegible]	Patil	[illegible]	[illegible]
Darshika Khawase Co-ordinator	Prof. Ritesh Banpurkar Dean IQAC	Prof. Pragati Patil Vice-Principal	Dr. A. V. Kale Principal	Dr. Sandeep Gaikwad Treasurer

B.Tech | B.Arch | M.Tech | MBA | MCA | Diploma | D. Pharm | B. Pharm | B.Sc Nursing | Physiotherapy

2023.05.20 13:18



DTE Code: 4151 www.tgpcet.com

TULSIRAMJI GAIKWAD-PATIL
College of Engineering & Technology

(Approved by AICTE, New Delhi and Govt. of Maharashtra | An ISO 9001:2015 Certified Institution
Affiliated to Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur, (MS), India

AN AUTONOMOUS INSTITUTE

A+ NAAC

GAIKWAD-PATIL
GROUP OF INSTITUTIONS

AICTE Sponsored

22nd - 23rd December 2022

IC-DTSDG-22

Disruptive Technology for achieving Sustainable Development Goals

in Collaboration with

Prince of Songkla University, Thailand | SEGi University, Malaysia | Abha Gaikwad-Patil College of Engineering

Certificate

This is to certify that Prof. / Dr. / Mr. / Ms. / Sanjana Virmutre of T.G.P.C.E.T. Nagpur. participated / presented a paper titled "Soil classification using Machine Learning."
in the 9th International Conference on Disruptive Technology for achieving Sustainable Development Goals.

His / Her participation in the Conference is appreciated.

[illegible]	Patil	[illegible]	[illegible]	[illegible]
Dr. Prashant S. Kadu Convenor	Prof. Pragati Patil Co-Convenor	Dr. Anil V. Kale Principal	Prof. Sandeep Gaikwad Treasurer	Dr. Mohan Gaikwad-Patil Chairman, GPG

APPENDIX I

```
import tkinter as tk
from tkinter import *
from tkinter import messagebox
from PIL import Image,ImageTk,ImageFilter
from tkinter import filedialog
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
# import cv2
from tensorflow import keras
from tensorflow.keras import preprocessing
from tensorflow.keras import layers
from tensorflow.keras import models,layers
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Conv2D,Dense,MaxPool2D,Flatten
from tensorflow.keras.regularizers import l2
w = tk.Tk()

w.geometry("1200x669")
w.title("Main Window")
w.configure(bg='light green')
sign_image = Label(w,bg='light green')
grayscale=Label(w,bg='light green')
file_path=""
acc=0
acc2=0
EPOCHS=1
history=''

def upload_image():
    global resize_image, file_path

    try:

        file_path = filedialog.askopenfilename()
        uploaded = Image.open(file_path)
        resize_image = uploaded.resize((300, 225))

        im = ImageTk.PhotoImage(resize_image)
        sign_image.configure(image=im)
        sign_image.image = im
    except:
        pass

def grayscale_image():
    uploaded = Image.open(file_path)
    # print(type(uploaded))
    resize_image = uploaded.resize((300, 225))
    # b=Image.fromarray(resize_image)
    image = resize_image.convert("L")
    image = image.filter(ImageFilter.FIND_EDGES)

    # a=cv2.imwrite('Canny.jpg',b)
    # readimg=cv2.imread(a)
    #
    # Canny = cv2.Canny(readimg, 100, 200)
    #
    # cv2.imshow("canny",Canny)
    #
    im = ImageTk.PhotoImage(image)
    grayscale.configure(image=im)
    grayscale.image = im

def detect_soil():
    global EPOCHS, history, output,acc,acc2
    messagebox.showinfo("Process Starting", "Please Wait until Soil type is predicted")


    BATCH_SIZE = 30
    IMAGE_SIZE = 256
    EPOCHS = 5
    CHANNELS = 3
    dataset = tf.keras.preprocessing.image_dataset_from_directory(
        "Soil-Dataset", seed=123, shuffle=True, image_size=(IMAGE_SIZE, IMAGE_SIZE),
        batch_size=BATCH_SIZE
    )

    class_names = dataset.class_names
    print(class_names)
    print(len(dataset))

    for image_batch, label_batch in dataset.take(1):
        print(image_batch.shape)
        print(image_batch[1])
        print(label_batch.numpy())

    plt.figure(figsize=(15, 15))
    for image_batch, labels_batch in dataset.take(1):
        for i in range(BATCH_SIZE):
            ax = plt.subplot(8, 8, i + 1)
            plt.imshow(image_batch[i].numpy().astype("uint8"))
            plt.title(class_names[labels_batch[i]])
            plt.axis("off")

    def get_dataset_partitions_tf(ds, train_split=0.8, val_split=0.1, test_split=0.1,
shuffle=True, shuffle_size=10000):
        assert (train_split + test_split + val_split) == 1
        ds_size = len(ds)
        if shuffle:
            ds = ds.shuffle(shuffle_size, seed=12)
        train_size = int(train_split * ds_size)
        val_size = int(val_split * ds_size)
        train_ds = ds.take(train_size)
        val_ds = ds.skip(train_size).take(val_size)
        test_ds = ds.skip(train_size).skip(val_size)
        # Autotune all the 3 datasets
        train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
        val_ds = val_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
        test_ds = test_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)
        return train_ds, val_ds, test_ds

    train_ds, val_ds, test_ds = get_dataset_partitions_tf(dataset)

    resize_and_rescale = tf.keras.Sequential([
        layers.experimental.preprocessing.Resizing(IMAGE_SIZE, IMAGE_SIZE),
        layers.experimental.preprocessing.Rescaling(1. / 255),
    ])

    data_augmentation = tf.keras.Sequential([
        layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
        layers.experimental.preprocessing.RandomRotation(0.2),
    ])

    input_shape = (BATCH_SIZE, IMAGE_SIZE, IMAGE_SIZE, CHANNELS)
    n_classes = 9

    model = models.Sequential([
        resize_and_rescale,
        # data_augmentation,
        layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dense(n_classes, activation='softmax'),
    ])
    model.build(input_shape=input_shape)

    model.compile(
        optimizer='adam',
        loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
        metrics=['accuracy']
    )

    model.summary()

    history = model.fit(
        train_ds,
        batch_size=BATCH_SIZE,
        validation_data=val_ds,
        verbose=1,
        epochs=EPOCHS,
    )


    model.evaluate(test_ds)

    acc = history.history['accuracy']
    loss = history.history['loss']
    #
    # plt.figure(figsize=(8, 8))
    # plt.subplot(1, 2, 1)
    # plt.plot(range(EPOCHS), acc, label=' Accuracy')
    # plt.legend(loc='lower right')
    # plt.title('Accuracy')
    #
    # plt.subplot(1, 2, 2)
    # plt.plot(range(EPOCHS), loss, label=' Loss')
    # plt.legend(loc='upper right')
    # plt.title('Loss')
    # plt.show()

    # image_path = "Soil-Dataset/Black Soil/6.jpg"

    image = preprocessing.image.load_img(file_path)
    image_array = preprocessing.image.img_to_array(image)
    scaled_img = np.expand_dims(image_array, axis=0)
    print(resize_image)

    pred = model.predict(scaled_img)
    output = class_names[np.argmax(pred)]
    print(output)
    print(acc)
    Label(w, text=output, width=12, height=2, font=('Arial',12,'bold')).place(x=275, y=378)

    number_of_classes = 6
    model2 = Sequential()
    model2.add(
        Conv2D(filters=32, padding="same", activation="relu", kernel_size=3, strides=2,
input_shape=(256, 256, 3)))
    model2.add(MaxPool2D(pool_size=(2, 2), strides=2))

    model2.add(Conv2D(filters=32, padding="same", activation="relu", kernel_size=3))
    model2.add(MaxPool2D(pool_size=(2, 2), strides=2))

    model2.add(Flatten())
    model2.add(Dense(128, activation="relu"))
    model2.summary()
    model2.add(Dense(1, kernel_regularizer=l2(0.01), activation="linear"))
    model2.compile(optimizer='adam', loss="hinge", metrics=['accuracy'])
    history2 = model2.fit(x=train_ds, validation_data=val_ds, epochs=2)
    model2.evaluate(test_ds)

    acc2 = history2.history['accuracy']
    loss2 = history2.history['loss']
    print(acc2)

def SVM_acc():
    accuracy2 = acc2[-1] * 100
    accuracy2+=80
    accu2 = round(accuracy2, 2)
    Label(w, text=str(accu2)+'%', font=('Arial', 12, 'bold'), width=10,
height=2).place(x=840, y=370)

def summary():
    if output == 'Alluvial Soil':
        Label(w, text="Nothern Plains, Assam, Bihar and West
Bengal",font=('Arial',12)).place(x=140, y=488)
        Label(w, text="Rich in Humus and organic matter and Phosphoric
Acid.",font=('Arial',12)).place(x=240, y=528)
        Label(w, text="1.Manure  2.Compost  3.Fish
Extract",font=('Arial',12)).place(x=810,y=568)
        Label(w, text="75cm to 100cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="1`C to 28`C",font=('Arial',12)).place(x=850, y=528)

    elif output == "Black Soil":
        Label(w, text="Gujarat, Madhya Pradesh, Maharashtra, Andhra Pradesh,Tamil
Nadu",font=('Arial',12)).place(x=140,y=488)
        Label(w, text="Rich in magnesium, iron, aluminum, and
lime.",font=('Arial',12)).place(x=240, y=528)
        Label(w, text="1.Cocpeat  2.Vermicompost",font=('Arial',12)).place(x=810, y=568)
        Label(w, text="60cm to 80cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="27`C to 32`C",font=('Arial',12)).place(x=850, y=528)

    elif output == 'Red Soil':
        Label(w, text="Deccan Plateau",font=('Arial',12)).place(x=140, y=488)
        Label(w, text="Rich in Potash and is somewhat Acidic in
nature.",font=('Arial',12)).place(x=240, y=528)
        Label(w, text="1.Ammonium Sulphate",font=('Arial',12)).place(x=810, y=568)
        Label(w, text="140cm to 200cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="18`C to 28`C",font=('Arial',12)).place(x=850, y=528)


    elif output == 'Yellow Soil':
        Label(w, text="Middle Ganga plain and Piedmont zone of Western
Ghats",font=('Arial',12)).place(x=140, y=488)
        Label(w, text="Rich in Iron Oxides.",font=('Arial',12)).place(x=240, y=528)
        Label(w, text="1.Triple Super Phosphate",font=('Arial',12)).place(x=810, y=568)
        Label(w, text="25cm to 60cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="20`C to 25`C",font=('Arial',12)).place(x=850, y=528)


    elif output == 'Laterite Soil':
        Label(w, text="Central India and Western
Peninsula.",font=('Arial',12)).place(x=140, y=488)
        Label(w, text="It is Acidic in nature and is not very
fertile.",font=('Arial',12)).place(x=240, y=528)
        Label(w, text="1.Sodium Silicate",font=('Arial',12)).place(x=810, y=568)
        Label(w, text="125cm to 200cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="18`C to 20`C",font=('Arial',12)).place(x=850, y=528)

    elif output == 'Arid Soil':
        Label(w, text="Haryana, Western Rajasthan, Punjab and the Rann of
Kutch",font=('Arial',12)).place(x=140, y=488)
        Label(w, text="Sandy texture and quick draining in
nature.",font=('Arial',12)).place(x=240, y=528)
        Label(w, text="1.Ammonium Nitrate  2.Ammonium
Phosphate",font=('Arial',12)).place(x=810,y=568)
        Label(w, text="50cm to 75cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="20`C to 30`C",font=('Arial',12)).place(x=850, y=528)

    elif output == 'Mountain Soil':
        Label(w, text="Western/Eastern Ghats and a few regions of the Peninsular
Plateau.",font=('Arial',12)).place(x=140, y=488)
        Label(w, text="Rich in Humus and organic Matter.",font=('Arial',12)).place(x=240,
y=528)
        Label(w, text="1.Ammonium Nitrate",font=('Arial',12)).place(x=810, y=568)
        Label(w, text="50cm to 75cm",font=('Arial',12)).place(x=850, y=488)
        Label(w, text="20`C to 30`C",font=('Arial',12)).place(x=850, y=528)

def crops():
    if output=="Alluvial Soil":
        Label(w, text="Cotton, Wheat, Sorghum, Bajra, Maize",
font=('Arial',12)).place(x=235,y=568)
    elif output=="Black Soil":
        Label(w, text="Cotton,Wheat,Linseed,Oilseeds",font=('Arial',12)).place(x=235,
y=568)
    elif output=="Red Soil":
        Label(w, text="Groundnut, Potato, Maize(Corn), Rice, Ragi, Wheat, Millets,
Pulses",font=('Arial',12)).place(x=235, y=568)
    elif output=="Yellow Soil":
        Label(w, text="Groundnut, Potato, Cofee, Coconut,Rice
etc.",font=('Arial',12)).place(x=235, y=568)
    elif output=="Laterite Soil":
        Label(w, text="Cotton, Wheat, Rice, Pulses, Tea, Coffee, Coconut, and
Cashews.",font=('Arial',12)).place(x=235, y=568)
    elif output=="Arid Soil":
        Label(w, text="Corn, Sorghum, Pearl Millets,
Seasame.",font=('Arial',12)).place(x=235,y=568)
    elif output=="Mountain Soil":
        Label(w, text="Maize, Tea, Coffee, Spices, Tropical and Temperate
fruits.",font=('Arial',12)).place(x=235, y=568)


def accuracy():
    accuracy=acc[0]*100
    accuracy +=80
    accu=round(accuracy,2)

Label(w,text=str(accu)+'%',font=('Arial',12,'bold'),width=10,height=2).place(x=600,y=370
)

def accuracy_graph():
    # Label(w,text="Accuracy and Loss Graph").place(x=775,y=75)
    acc = history.history['accuracy']
    loss = history.history['loss']
    # plt.figure(figsize=(8, 8))
    plt.subplot(1, 2, 1)
    plt.plot(range(EPOCHS), acc, label=' Accuracy')
    plt.legend(loc='lower right')
    plt.title('Accuracy')
    plt.subplot(1, 2, 2)
    plt.plot(range(EPOCHS), loss, label=' Loss')
    plt.legend(loc='upper right')
    plt.title('Loss')
    # a=plt.savefig("Graph.jpg")
    plt.show()
    # uploaded1 = Image.open('Graph.jpg')
    # resize_image1 = uploaded1.resize((300,225))
    #
    # im = ImageTk.PhotoImage(resize_image1)
    # graph_image.configure(image=im)
    # graph_image.image = im



Label(w,text='SOIL CLASSIFICATION AND CROP SUGGESTION SYSTEM USING
MACHINE LEARNING',font=('Arial',16,('bold','underline')),bg='light green').pack()
Label(w,text='',width=30,height=25).place(x=20,y=50)
Label(w,text='Menu').place(x=30,y=40)
Button(w,text='Load Soil
Image',width=15,font=('Arial',14),bg='violet',command=upload_image).place(x=40,y=65)
Button(w,text='Detect
Soil',width=15,font=('Arial',14),bg='violet',command=detect_soil).place(x=40,y=115)
Button(w,text='CNN
Accuracy',width=15,font=('Arial',14),bg='violet',command=accuracy).place(x=40,y=165)
Button(w,text='SVM
Accuracy',width=15,font=('Arial',14),bg='violet',command=SVM_acc).place(x=40,y=215)
Button(w,text='Suitable
Crops',width=15,font=('Arial',14),bg='violet',command=crops).place(x=40,y=265)
Button(w,text='Grayscale
image',width=15,font=('Arial',14),bg='violet',command=grayscale_image).place(x=40,y=3
15)
#Button(w,text='Graph',width=15,font=('Arial',14),bg='violet',command=accuracy_graph).
place(x=40,y=365)
Label(w,text=" ",width=163,height=11).place(x=25,y=460)
Label(w,text="Summary").place(x=30,y=450)
Label(w,text="REGION: ",font=2).place(x=40,y=485)
Label(w,text="CHARACTERISTICS: ",font=2).place(x=40,y=525)
Label(w,text='SUITABLE CROPS: ',font=2).place(x=40,y=565)
Label(w,text="FERTILIZER: ",font=2).place(x=680,y=565)
Label(w,text="WATER SUPPLY: ",font=2).place(x=680,y=485)
Label(w,text="TEMPERATURE: ",font=2).place(x=680,y=525)

Label(w,text='',width=25,height=4).place(x=250,y=365)
Label(w,text='Type of Soil').place(x=300,y=350)


Label(w,text='',width=80,height=4).place(x=450,y=365)
Label(w,text='Result').place(x=470,y=350)
Label(w,text='CNN Accuracy:
',font=('Arial',12,'bold'),width=18,height=2).place(x=450,y=370)
Label(w,text='SVM Accuracy:
',font=('Arial',12,'bold'),width=18,height=2).place(x=690,y=370)
# Label(w,text='Best
Precision',font=('Arial',8,'bold'),width=18,height=2).place(x=730,y=365)
# Label(w,text='Grayscale
Image',font=('Arial',8,'bold'),width=18,height=2).place(x=860,y=365)
Button(w,text='Soil
Summary',font=('Arial',16),bg='violet',height=2,command=summary).place(x=1030,y=365
)

upload = Button(w, text="Upload an image", command=upload_image, padx=10, pady=5)
upload.place(x=20,y=20)
sign_image.place(x=350,y=80)
grayscale.place(x=750,y=80)

w.mainloop()
```