

INTELLIGENT CROP RECOMMENDATION SYSTEM USING MACHINE LEARNING

Submitted in partial fulfillment of the requirements for the
award of Bachelor of Science in Computer Science

By

ERNEST PRINCE SAMVEL.P

(Reg. No. 39290024)

KARTHICK.G (Reg. No.39290046)



- **DEPARTMENT OF COMPUTER SCIENCE**

SCHOOL OF COMPUTING

**SATHYABAMA
INSTITUTE OF SCIENCE AND
TECHNOLOGY**

**JEPPIAAR NAGAR, RAJIV GANDHI
SALAI, CHENNAI – 600119,
TAMILNADU**

MARCH 2022



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **ERNEST PRINCE SAMVEL.P (Reg. No. 39290024) KARTHICK.G (Reg. No. 39290046)** who carried out the project entitled **“INTELLIGENT CROP RECOMMENDATION SYSTEM USING MACHINE”** under my supervision from November 2020 to April 2021.

Internal Guide

Dr. M. SELVI, M.E., Ph.D.,

Head of the Department

Dr. LAKSHMANAN, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **KARTHICK.G (Reg. No. 39290046)** hereby declare that the Project Report entitled "**INTELLIGENT CROP RECOMMENDATION SYSTEM USING MACHINE LEARNING**" done by me under the guidance of **Dr. M. SELVI** is submitted in partial fulfillment of the requirements for the award of Bachelor of Science degree in Computer Science.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. L. LAKSHMANAN, M.E., Ph.D.**, Head of the Department, Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. M. SELVI , M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways For the Completion of the project.

ABSTRACT

Music plays a very important role in people's lives. Music brings like-minded people together and is the glue that holds communities together. Communities can be recognized by the type of songs that they compose, or even listen to. The purpose of our project and research is to find a better machine learning algorithm than the pre-existing models that predicts the genre of songs.

Genres can be defined as categorical labels created by humans to identify or characterize the style of music. The concept of automatic music genre classification has become very popular in recent years as a result of the rapid growth of the digital entertainment industry.

This work presents a comprehensive machine learning approach to the problem of automatic musical genre classification using the audio signal. The system is developed using a KNN and SVM to recognize the genre.

TABLE OF CONTENTS

CHAPTER No	TITLE	PAGE
	LIST OF FIGURES	Vi
	LIST OF ABBREVIATIONS	Vii
1	INTRODUCTION	
1.1	About the Project	
2	SYSTEM ANALYSIS	
2.1	Existing system	
2.2	Proposed system	
3	REQUIREMENTS SPECIFICATION	
3.1	Introduction	
3.2	Hardware and Software specification	
3.3	Technologies Used	
3.3.1	Python	
3.3.3.1	Introduction to Python	
4.1	Design and Implementation Constraints	
4.2	Other Nonfunctional Requirements	
5	Architecture Diagram	
6.	Module	
6.2	Module explanation	
7	CODING AND TESTING	
7.1	Coding	

7.2	Coding standards	
7.3	Test procedure	
7.4	Test data and output	
	SOURCE CODE	
	SNAP SHOTS	
	REFERENCES	

LIST OF ABBREVIATIONS

CHAPTER 1

INTRODUCTION

Aim:

Predict the crop yield and deliver the end user with proper recommendations about required fertilizer ratio based on soil parameters. We also recommend the crop price using machine learning.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Now a day's, dilettante farmers are hard to understand the cultivation process, crop type, climate change, etc. Farming is that the spine for every nation's economy. Future agriculture depends on dilettante formers. But new farmers not so strong at farming, So Machine learning help to solve their problems The existing system predicts the crop yield by using the soil parameters and recommend Fertilizer using machine learning. It uses the crop yield information to make the end users decide on the crop to be sown. Hence the system is not simple enough for dilettante farmers to understand.

2.2PROPOSED SYSTEM

The system prepared predict major crops yield in a particular district in Tamil Nadu. The client on their first login has to register themselves on the Web application created by flask. The login details are stored in SQLite database. Once the user login into the system they gets all the access for predicting crop yield and using the input such as location, nitrogen, phosphorous, potassium and pH values depends on their forming land environment.. We can also find the primary nutrients of soil by given the input as crop name. It passes the various inputs to the controller which uses the Random Forest for classification. We recommend to the former how much fertilizer required in ratio based on soil parameters and the crop price using machine learning techniques.

CHAPTER 3

REQUIREMENT SPECIFICATIONS

3.1 INTRODUCTION

INDIA is a highly populated country and randomly change in the climatic conditions need to secure the world food resources. Framers face serious problems in drought conditions. Type of soil plays a major role in the crop yield. Suggesting the use of fertilizers may help the farmers to

make the best decision for their cropping situation [1]. The number of studies Information and Communication Technology (ICT) can be applied for prediction of crop yield [2]. By the use of Data Mining, we can also predict the crop yield. By fully analyze the previous data we can suggest the farmer for a better crop for the better yield [3]. For the better yield we need to consider soil type and soil fertility things and also one of the major factors rainfall and groundwater availability if it is dry land it is better to go for cash crops and if is wetland it is better to go for wheat and sugarcane[4]. There are 15 agro-climatic regions in India these regions are divided on the bases of a type of the land. Each agro-climatic regions can grow some specific crops. Based on that we need to suggest the farmer that which crop is best among those crops which belong to those climatic regions. Achieving the maximum crop at minimum yield is the ultimate Aim of the project. Early detection of problems and management of that problems can help the farmers for better crop yield[5]. Crop yield prediction is the important research which helps to secure food[6,7]. For the better understanding of the crop yield, we need to study of the huge data with the help of machine learning algorithm so it will give the accurate yield for that crop and suggest the farmer for a better crop. Improving the quantity of the crop is the key goal of precision agriculture means obtaining a better understanding of the crop using the information technology methods. The main goal of precision agriculture is profitability and sustainability [8]. From ancient times agriculture has become the backbone of our country. Nowadays climatic conditions vary very often. So, it is hard to grow crops by understanding weather conditions[8]. We need to use some technology to find or understand the crop details and guide the farmers to grow crops accordingly and moreover fertilizer also one of the major factors to grow crops accordingly[9]. If fertilizer is used more or less in the field the soil may lose it fertility and crop may not give the expected yield. so, fertilizer also becomes the major factor in it [10]. mostly

understanding the temperature conditions is much necessary for India because we can improve the Indian economy with the help of the crop prediction because it plays a major role in the Indian economy. Generally, machine learning algorithms will predict the most efficient output of the yield [11]. Previously yield is predicted on the bases of the farmers prior experience but now weather conditions may change drastically so they cannot guess the yield [12]. so, technology can help them to predict the yield of the crop weather to go for that crop or no. machine learning model will understand the pattern of the crop and yield based on the several conditions and predicts the yield of the area in which he is going to crop [13].

3.2 HARDWARE AND SOFTWARE SPECIFICATION

3.2.1 HARDWARE REQUIREMENTS

- Hard Disk : 500GB and Above
- RAM : 4GB and Above
- Processor : I3 and Above

3.2.2 SOFTWARE REQUIREMENTS

- Operating System : Windows 7 , 8, 10 (64 bit)
- Software : Python and Anaconda
- Framework : Flask, HTML
- Database : SQLite

3.3 TECHNOLOGIES USED

- Python

3.3.1 Introduction to Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- System scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some

other programming languages.

- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- Python 2.0 was released in 2000, and the 2.x versions were the prevalent releases until December 2008. At that time, the development team made the decision to release version 3.0, which contained a few relatively small but significant changes that were not backward compatible with the 2.x versions. Python 2 and 3 are very similar, and some features of Python 3 have been backported to Python 2. But in general, they remain not quite compatible.
- Both Python 2 and 3 have continued to be maintained and developed, with periodic release updates for both. As of this writing, the most recent versions available are 2.7.15 and 3.6.5. However, an official [End Of Life date of January 1, 2020](#) has been established for Python 2, after which time it will no longer be maintained.
- Python is still maintained by a core development team at the Institute, and Guido is still in charge, having been given the title of BDFL (Benevolent Dictator For Life) by the Python community. The name Python, by the way, derives not from the snake, but from the British comedy troupe [Monty Python's Flying Circus](#), of which Guido was, and presumably still is, a fan. It is common to find references to Monty Python sketches and movies scattered throughout the Python documentation.
- It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language

with influence from mathematics.

- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python is Interpreted

- Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer's processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.
- This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.
- One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.
- In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth it for most applications.
- For all its syntactical simplicity, Python supports most constructs that would be expected in a very high-level language, including complex dynamic data types, structured and functional programming, and [object-oriented programming](#).
- Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language, such as database manipulation or GUI programming.
- Python accomplishes what many programming languages don't: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

Introduction Machine learning

Introduction:

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through learning. In its application across business problems, machine learning is also referred to as predictive analytics.

Machine learning tasks:

Machine learning tasks are classified into several broad categories. In supervised learning, the algorithm builds a mathematical model from a set of data that contains both the inputs and the desired outputs. For example, if the task were determining whether an image contained a certain object, the training data for a supervised learning algorithm would include images with and without that object (the input), and each image would have a label (the output) designating whether it

contained the object. In special cases, the input may be only partially available, or restricted to special feedback. Semi algorithms develop mathematical models from incomplete training data, where a portion of the sample input doesn't have labels.

Classification algorithms and regression algorithms are types of supervised learning. Classification algorithms are used when the outputs are restricted to a limited set of values. For a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email. For an algorithm that identifies spam emails, the output would be the prediction of either "spam" or "not spam", represented by the Boolean values true and false. Regression algorithms are named for their continuous outputs, meaning they may have any value within a range. Examples of a continuous value are the temperature, length, or price of an object.

In unsupervised learning, the algorithm builds a mathematical model from a set of data that contains only inputs and no desired output labels. Unsupervised learning algorithms are used to find structure in the data, like grouping or clustering of data points. Unsupervised learning can discover patterns in the data, and can group the inputs into categories, as in feature learning. Dimensionality reduction is the process of reducing the number of "features", or inputs, in a set of data.

Active learning algorithms access the desired outputs (training labels) for a limited set of inputs based on a budget and optimize the choice of inputs for which it will acquire training labels. When used interactively, these can be presented to a human user for labeling. Reinforcement learning algorithms are given feedback in the form of positive or negative reinforcement in a dynamic environment and are used in autonomous vehicles or in learning to play a game against a human opponent. Other specialized algorithms in machine learning include topic modeling, where the computer program is given a set of natural language documents and finds other

documents that cover similar topics. Machine learning algorithms can be used to find the unobservable probability density function in density estimation problems. Meta learning algorithms learn their own inductive bias based on previous experience. In developmental robotics, robot learning algorithms generate their own sequences of learning experiences, also known as a curriculum, to cumulatively acquire new skills through self-guided exploration and social interaction with humans. These robots use guidance mechanisms such as active learning, maturation, motor synergies, and imitation.

Types of learning algorithms:

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

Supervised learning:

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Supervised learning algorithms include classification and regression. Classification algorithms are used when the outputs are restricted to a limited set of values, and regression algorithms are used when the outputs may have any numerical value within a range. Similarity learning is an area of supervised machine learning closely related to regression and classification, but the goal is to learn from examples using a similarity function that measures how similar or related two objects are. It has applications in ranking, recommendation systems, visual identity tracking, face verification, and speaker verification.

In the case of semi-supervised learning algorithms, some of the training examples are missing training labels, but they can nevertheless be used to improve the quality of a model. In weakly supervised learning, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

Unsupervised learning:

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called *clusters*) so that observations within the same cluster are similar according to one or more pre designated criteria, while observations drawn from different

clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some *similarity metric* and evaluated, for example, by *internal compactness*, or the similarity between members of the same cluster, and *separation*, the difference between clusters. Other methods are based on *estimated density* and *graph connectivity*.

Semi-supervised learning:

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

Support Vector Machine (SVM)

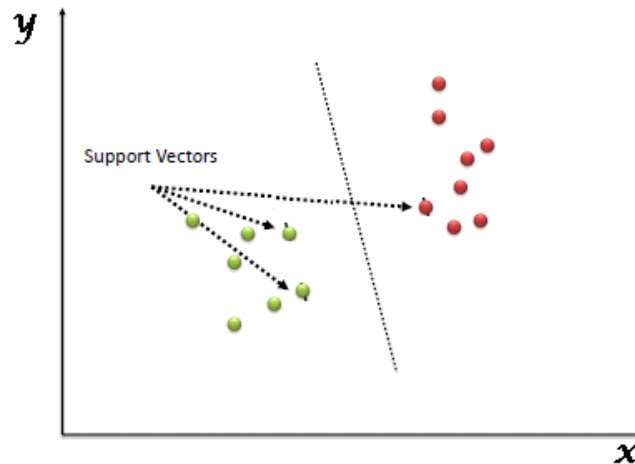
Introduction:

Mastering machine learning algorithms isn't a myth at all. Most of the beginners start by learning regression. It is simple to learn and use, but does that solve our purpose? Of course not! Because, you can do so much more than just Regression!

Think of machine learning algorithms as an armory packed with axes, sword, blades, bow, dagger etc. You have various tools, but you ought to learn to use them at the right time. As an analogy, think of 'Regression' as a sword capable of slicing and dicing data efficiently, but incapable of dealing with highly complex data. On the contrary, 'Support Vector Machines' is like a sharp knife – it works on smaller datasets, but on them, it can be much stronger and powerful in building models.

Support Vector Machine

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for either classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).



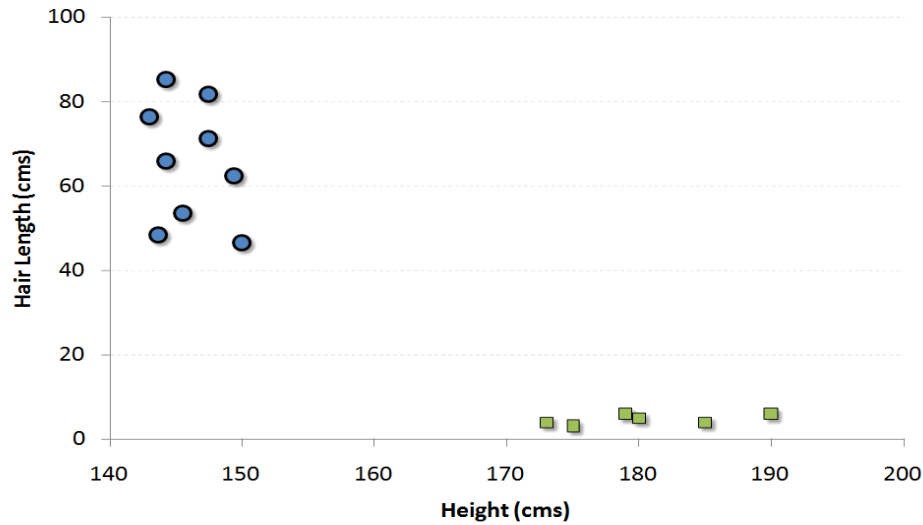
Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line). You can look at definition of support vectors and a few examples of its working here.

Support Vector Machine – Simplified

Let's talk about how support vector machine works. **This is suitable for readers who do not know much about this algorithm and have a curiosity to learn a new technique.** In following explore the technique in detail and analyze cases where such techniques are stronger than other techniques.

Classification analysis

Let's consider an example to understand these concepts. We have a population composed of 50%-50% Males and Females. Using a sample of this population, you want to create some set of rules which will guide us the gender class for rest of the population. Using this algorithm, we intend to build a robot which can identify whether a person is a Male or a Female. This is a sample problem of classification analysis. Using some set of rules, we will try to classify the population into two possible segments. For simplicity, let's assume that the two differentiating factors identified are: Height of the individual and Hair Length. Following is a scatter plot of the sample.



The blue circles in the plot represent females and green squares represents male. A few expected insights from the graph are:

1. Males in our population have a higher average height.
2. Females in our population have longer scalp hairs.

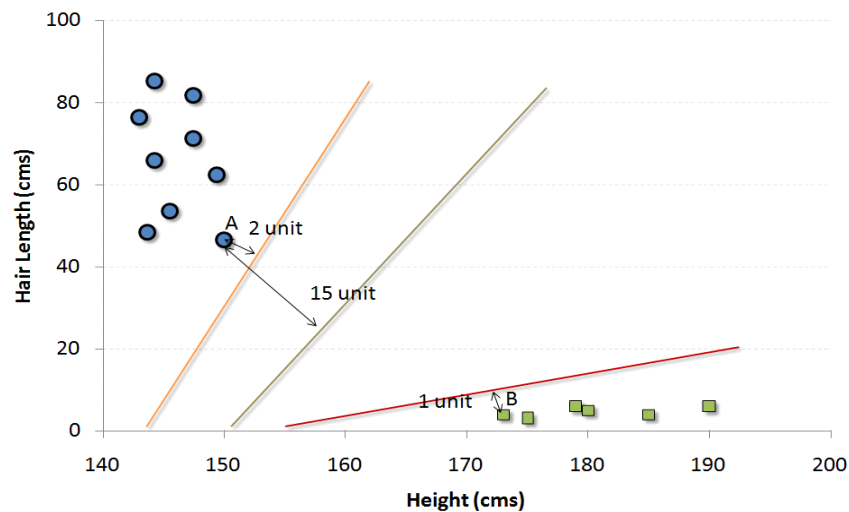
If we were to see an individual with height 180 cm's and hair length 4 cm's, our best guess will be to classify this individual as a male. This is how we do a classification analysis.

A Support Vector and SVM

Support Vectors are simply the co-ordinates of individual observation. For instance, (45,150) is a support vector which corresponds to a female. Support Vector Machine is a frontier which best segregates the Male from the Females. In this case, the two classes are well separated from each other; hence it is easier to find a SVM.

How to find the Support Vector Machine for case in hand?

There are many possible frontiers which can classify the problem in hand. Following are the three possible frontiers.

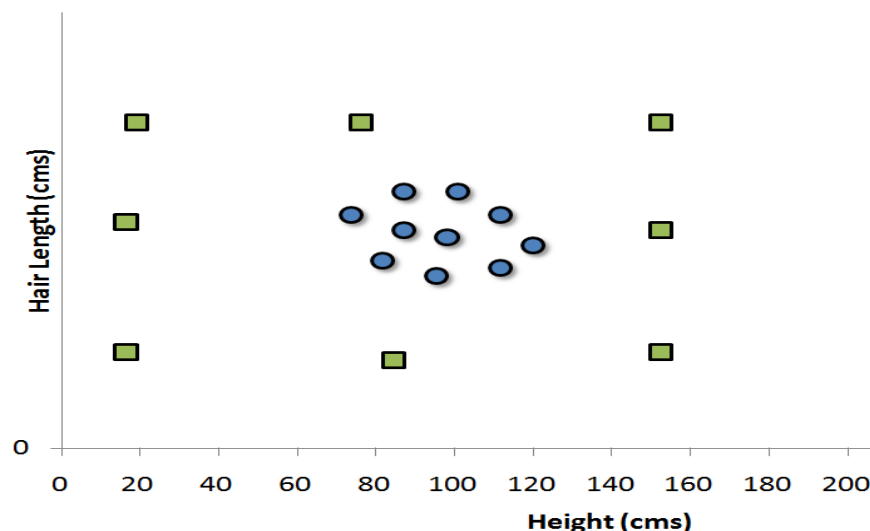


How do we decide which is the best frontier for this particular problem statement?

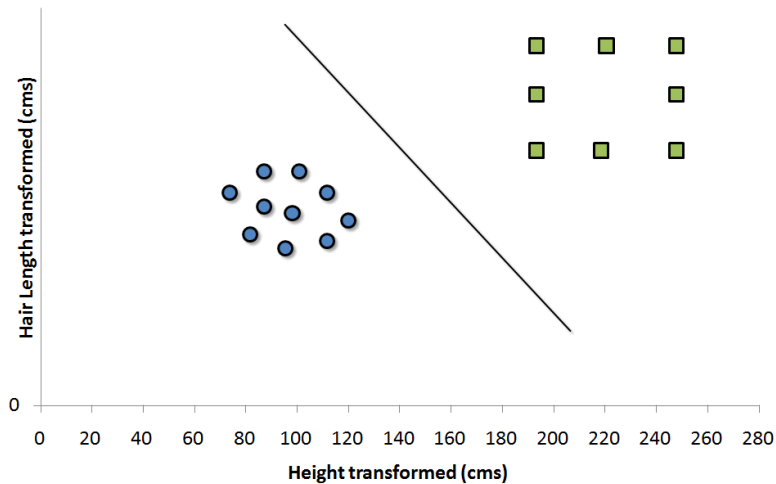
The easiest way to interpret the objective function in a SVM is to find the minimum distance of the frontier from closest support vector (this can belong to any class). For instance, orange frontier is closest to blue circles. And the closest blue circle is 2 units away from the frontier. Once we have these distances for all the frontiers, we simply choose the frontier with the maximum distance (from the closest support vector). Out of the three shown frontiers, we see the black frontier is farthest from nearest support vector (i.e. 15 units).

What if we do not find a clean frontier which segregates the classes?

Our job was relatively easier finding the SVM in this business case. What if the distribution looked something like as follows?



In such cases, we do not see a straight line frontier directly in current plane which can serve as the SVM. In such cases, we need to map these vectors to a higher dimension plane so that they get segregated from each other. Such cases will be covered once we start with the formulation of SVM. For now, you can visualize that such transformation will result into following type of SVM.



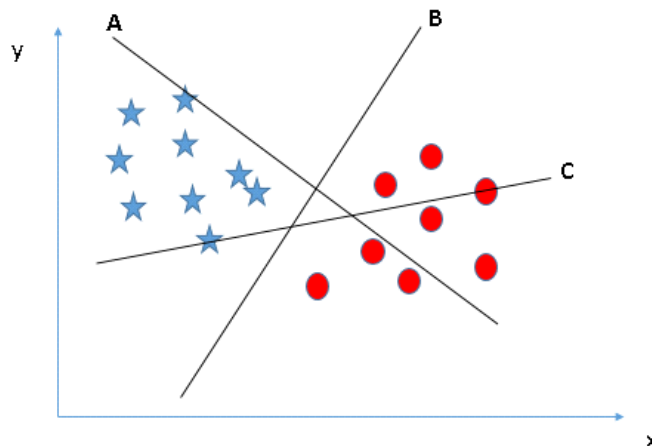
Each of the green squares in original distribution is mapped on a transformed scale. And transformed scale has clearly segregated classes. Many algorithms have been proposed to make these transformations.

How does it work?

Above, we got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is “How can we identify the right hyper-plane?” Don’t worry; it’s not as hard as you think!

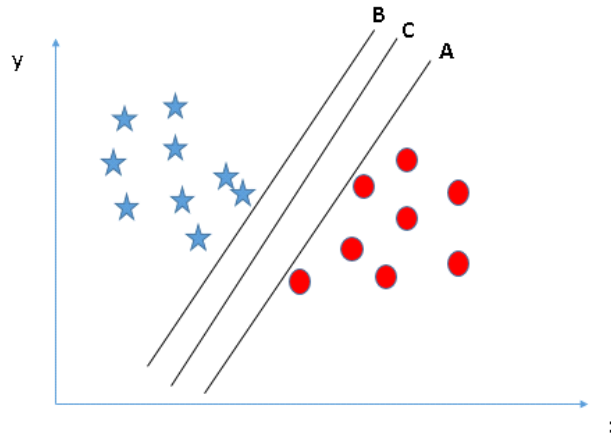
Let’s understand:

- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.

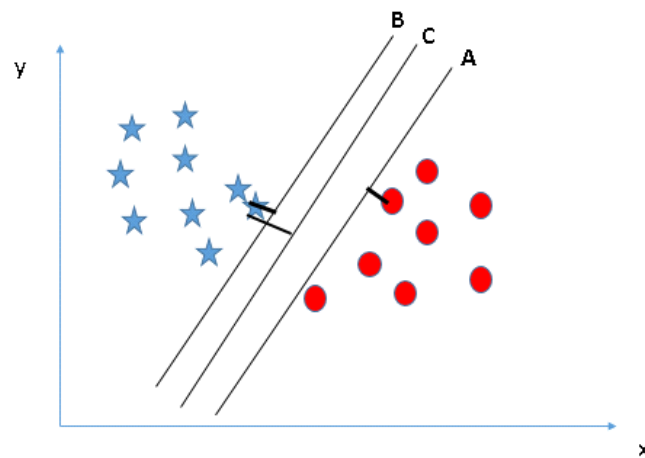


You need to remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.

- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, how can we identify the right hyper-plane?

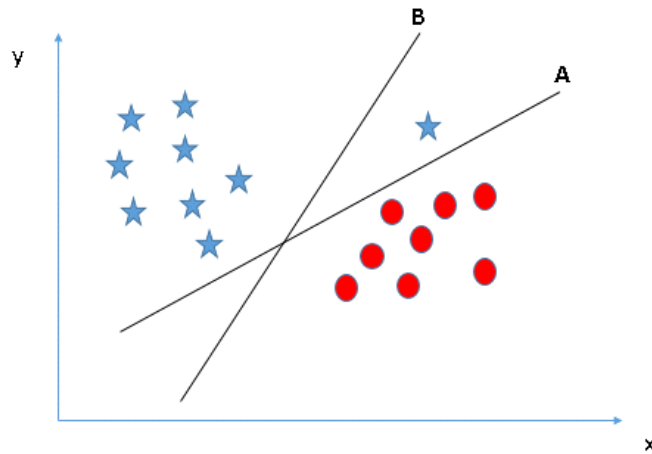


Here, maximizing the distances between nearest data points (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let's look at the below snapshot:



Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

- **Identify the right hyper-plane (Scenario-3):** Hint: Use the rules as discussed in previous section to identify the right hyper-plane

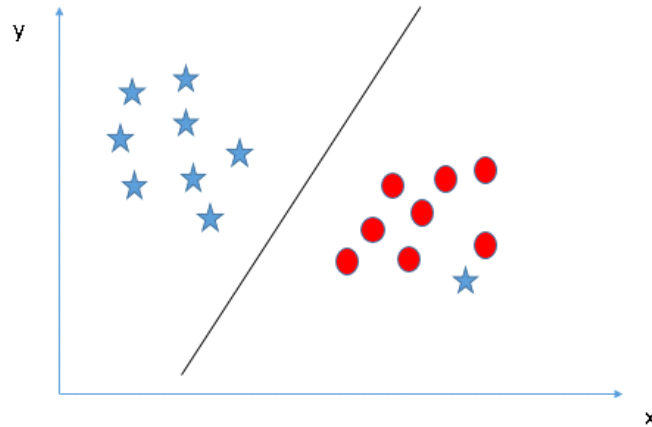


Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A**. But, here is the catch; SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A**.

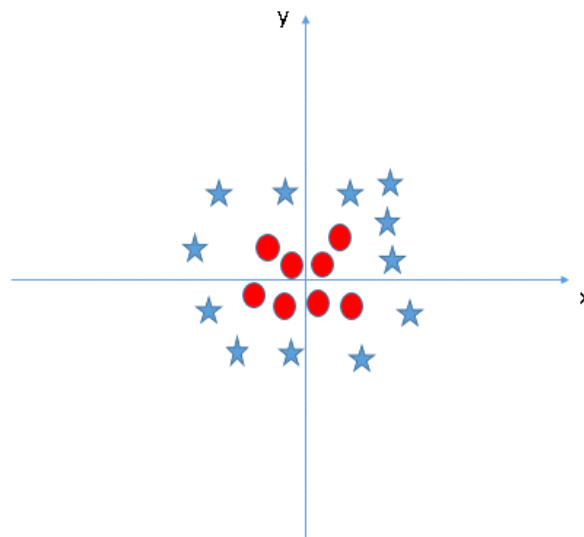
- **Can we classify two classes (Scenario-4)?:** Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other(circle) class as an outlier.



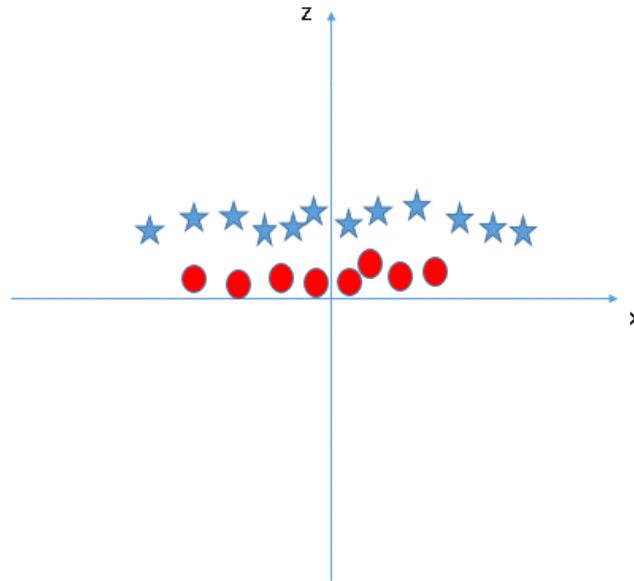
As I have already mentioned, one star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.



- **Find the hyper-plane to segregate to classes (Scenario-5):** In the scenario below, we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.



SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature $z = x^2 + y^2$. Now, let's plot the data points on axis x and z:

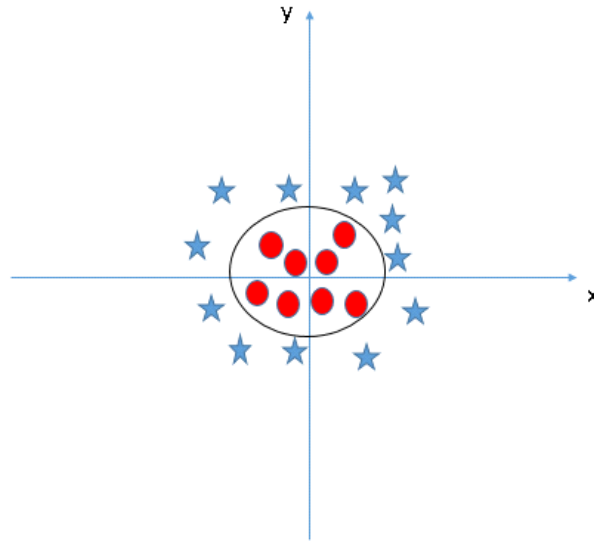


In above plot, points to consider are:

- All values for z would be positive always because z is the squared sum of both x and y
- In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z .

In SVM, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the **kernel trick**. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs you've defined.

When we look at the hyper-plane in original input space it looks like a circle:



Pros and Cons associated with SVM

- **Pros:**
 - It works really well with clear margin of separation
 - It is effective in high dimensional spaces.
 - It is effective in cases where number of dimensions is greater than the number of samples.
 - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- **Cons:**
 - It doesn't perform well, when we have large data set because the required training time is higher
 - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
 - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is related SVC method of Python scikit-learn library

Random Forest

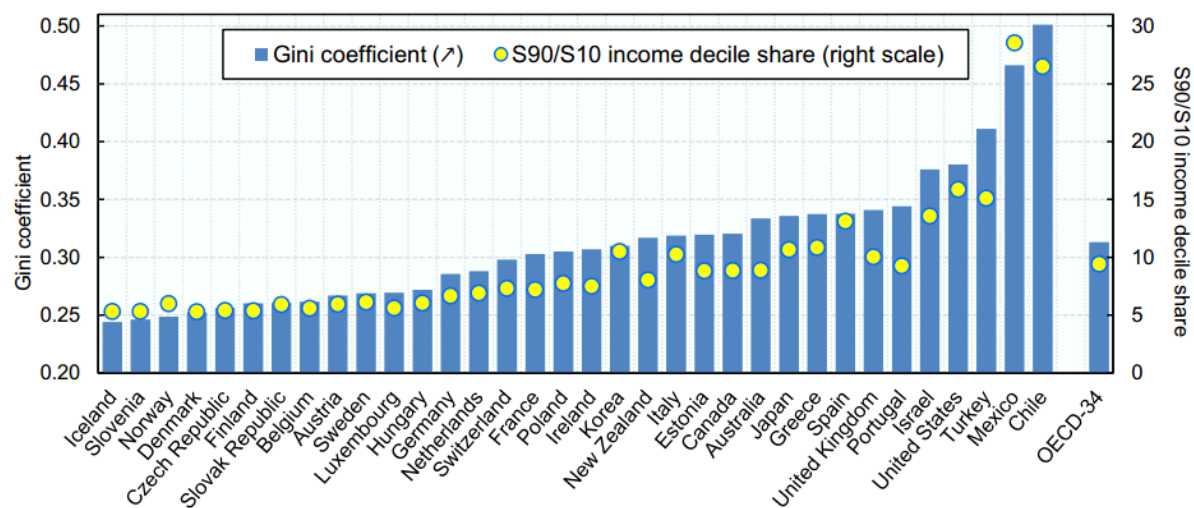
Introduction

With increase in computational power, we can now choose algorithms which perform very intensive calculations. One such algorithm is “Random Forest”, which we will discuss in this article. While the algorithm is very popular in various competitions (e.g. like the ones running on Kaggle), the end output of the model is like a black box and hence should be used judiciously.

Before going any further, here is an example on the importance of choosing the best algorithm.

Random Forest Case Study

Following is a distribution of Annual income Gini Coefficients across different countries:



Mexico has the second highest Gini coefficient and hence has a very high segregation in annual income of rich and poor. Our task is to come up with an accurate predictive algorithm to estimate annual income bracket of each individual in Mexico. The brackets of income are as follows:

1. below \$40,000
2. \$40,000 – 150,000
3. More than \$150,000

Following are the information available for each individual:

1. Age, 2. Gender, 3. Highest educational qualification, 4. Working in Industry, 5. Residence in Metro/Non-metro

We need to come up with an algorithm to give an accurate prediction for an individual who has following traits:

1. Age: 35 years, 2. Gender: Male, 3. Highest Educational Qualification: Diploma holder, 4. Industry: Manufacturing, 5. Residence: Metro

We will only talk about random forest to make this prediction in this article.

The algorithm of Random Forest

Random forest is like bootstrapping algorithm with Decision tree (CART) model. Say, we have 1000 observation in the complete population with 10 variables. Random forest tries to build multiple CART model with different sample and different initial variables. For instance, it will take a random sample of 100 observation and 5 randomly chosen initial variables to build a CART model. It will repeat the process (say) 10 times and then make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction.

Back to Case study

Disclaimer: The numbers in this article are illustrative

Mexico has a population of 118 MM. Say, the algorithm Random forest picks up 10k observation with only one variable (for simplicity) to build each CART model. In total, we are looking at 5 CART models being built with different variables. In a real life problem, you will have more number of population sample and different combinations of input variables.

Salary bands:

Band 1: Below \$40,000

Band 2: \$40,000 – 150,000

Band 3: More than \$150,000

Following are the outputs of the 5 different CART model.

CART 1: Variable Age

	Salary Band	1	2	3
Age	Below 18	90%	10%	0%
	19-27	85%	14%	1%
	28-40	70%	23%	7%
	40-55	60%	35%	5%
	More than 55	70%	25%	5%

CART 2: Variable Gender

	Salary Band	1	2	3
Gender	Male	70%	27%	3%
	Female	75%	24%	1%

CART 3: Variable Education

	Salary Band	1	2	3
Education	<=High School	85%	10%	5%
	Diploma	80%	14%	6%
	Bachelors	77%	23%	0%
	Master	62%	35%	3%

CART 4: Variable Residence

	Salary Band	1	2	3
Residence	Metro	70%	20%	10%
	Non-Metro	65%	20%	15%

CART 5: Variable Industry

	Salary Band	1	2	3
Industry	Finance	65%	30%	5%
	Manufacturing	60%	35%	5%
	Others	75%	20%	5%

Using these 5 CART models, we need to come up with single set of probability to belong to each of the salary classes. For simplicity, we will just take a mean of probabilities in this case study. Other than simple mean, we also consider vote method to come up with the final prediction. To come up with the final prediction let's locate the following profile in each CART model:

1. Age: 35 years, 2, Gender: Male, 3. Highest Educational Qualification: Diploma holder, 4. Industry: Manufacturing, 5. Residence: Metro

For each of these CART model, following is the distribution across salary bands:

CART	Band	1	2	3
Age	28-40	70%	23%	7%
Gender	Male	70%	27%	3%
Education	Diploma	80%	14%	6%
Industry	Manufacturing	60%	35%	5%
Residence	Metro	70%	20%	10%
Final probability		70%	24%	6%

The final probability is simply the average of the probability in the same salary bands in different CART models. As you can see from this analysis, that there is 70% chance of this individual falling in class 1 (less than \$40,000) and around 24% chance of the individual falling in class 2.

Conclusion

Random forest gives much more accurate predictions when compared to simple CART/CHAID or regression models in many scenarios. These cases generally have high number of predictive variables and huge sample size. This is because it captures the variance of several input variables at the same time and enables high number of observations to participate in the prediction.

K-Nearest Neighbors

Introduction

In four years of the analytics built more than 80% of classification models and just 15-20% regression models. These ratios can be more or less generalized throughout the industry. The reason of a bias towards classification models is that most analytical problem involves making a decision. For instance will a customer attrite or not, should we target customer X for digital campaigns, whether customer has a high potential or not etc. This analysis is more insightful and directly links to an implementation roadmap. In this article, we will talk about another widely used classification technique called K-nearest neighbors (KNN). Our focus will be primarily on how does the algorithm work and how does the input parameter effect the output/prediction.

KNN algorithm

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

1. Ease to interpret output
2. Calculation time
3. Predictive Power

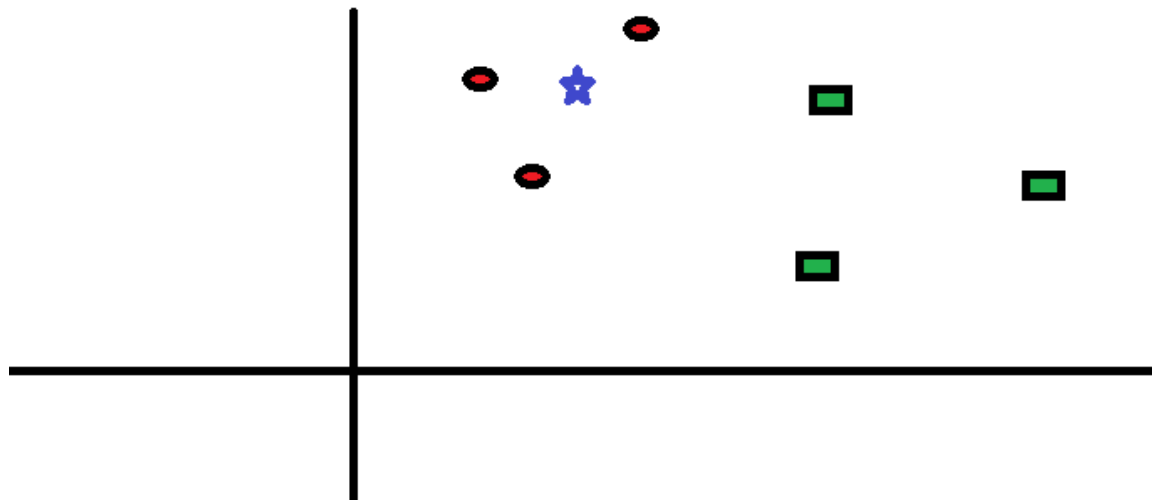
Let us take a few examples to place KNN in the scale:

	Logistic Regression	CART	Random Forest	KNN
1. Ease to interpret output	2	3	1	3
2. Calculation time	3	2	1	3
3. Predictive Power	2	2	3	2

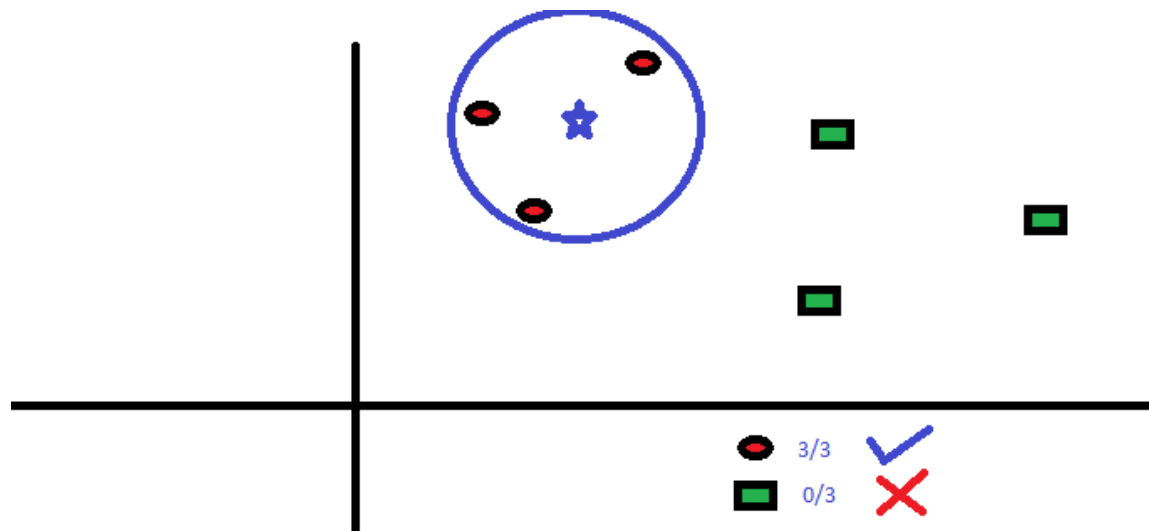
KNN algorithm fares across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time.

The KNN algorithm work

Let's take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS):



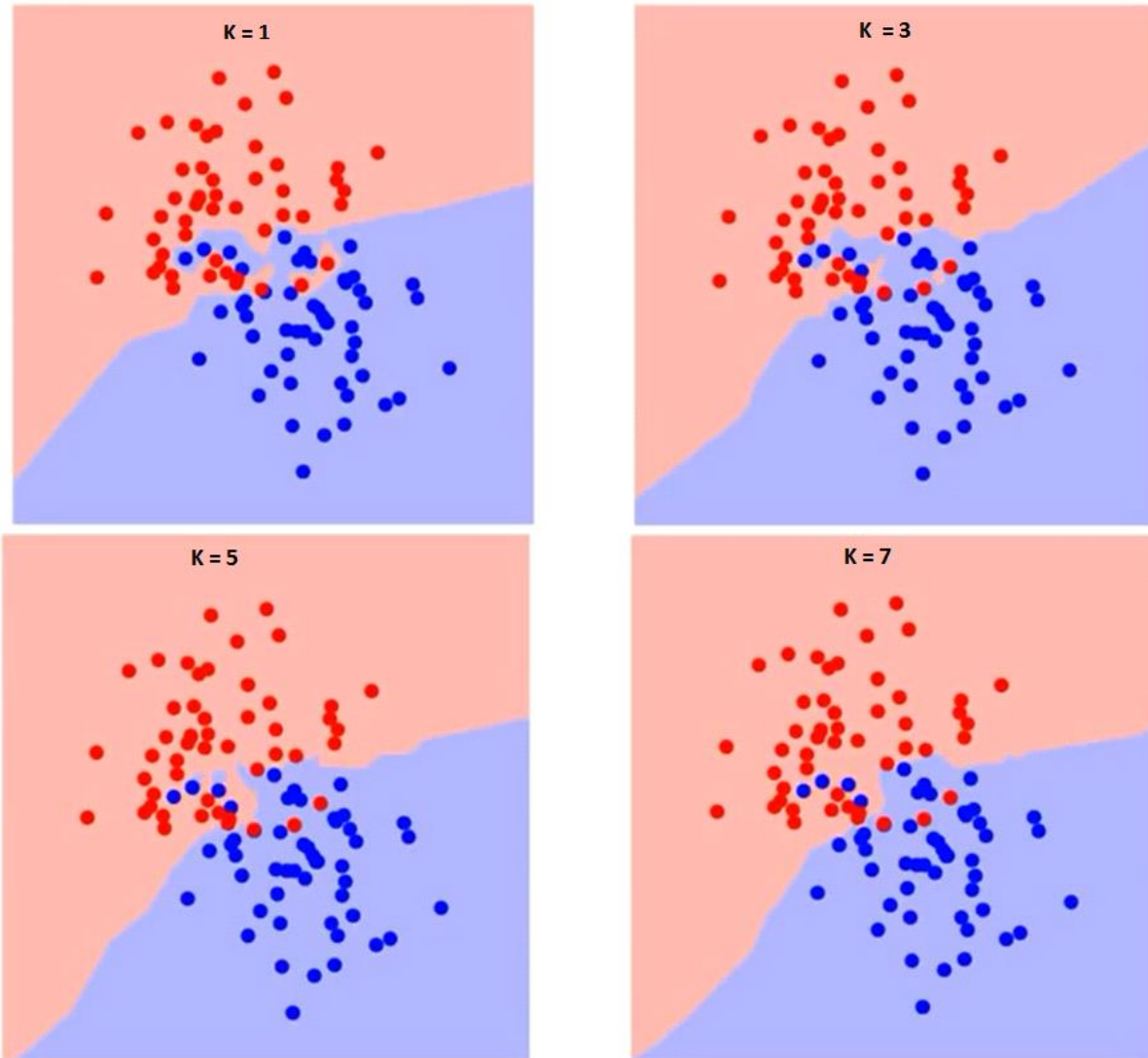
You intend to find out the class of the blue star (BS). BS can either be RC or GS and nothing else. The “K” in KNN algorithm is the nearest neighbors we wish to take vote from. Let's say $K = 3$. Hence, we will now make a circle with BS as center just as big as to enclose only three data points on the plane. Refer to following diagram for more details:



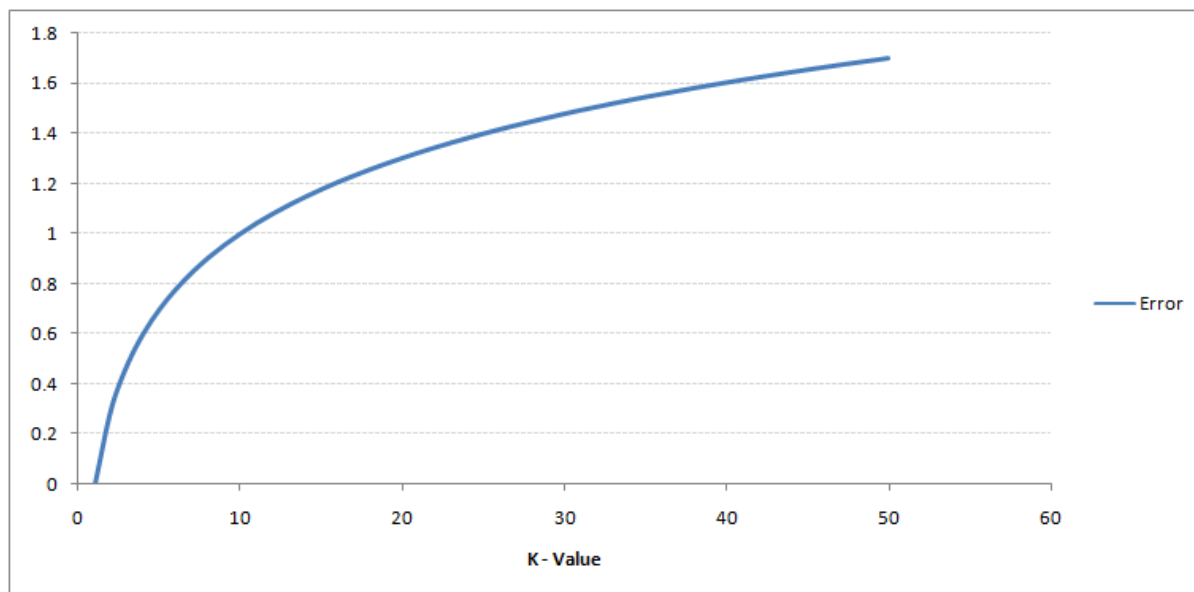
The three closest points to BS is all RC. Hence, with good confidence level we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm.

How do we choose the factor K?

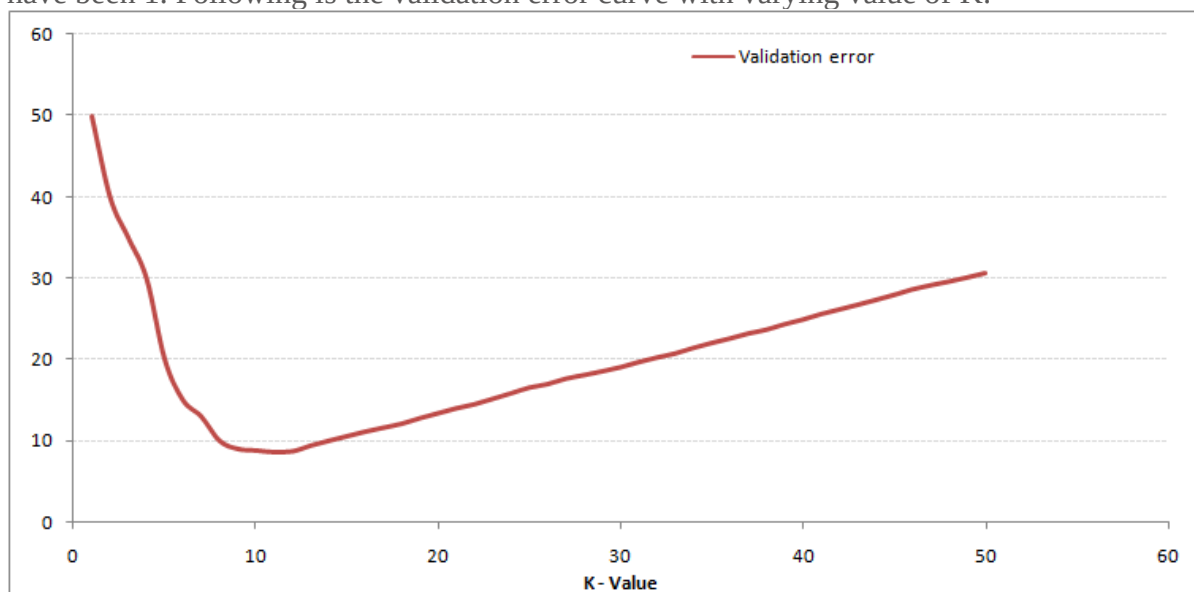
First let us try to understand what exactly does K influence in the algorithm. If we see the last example, given that all the 6 training observation remain constant, with a given K value we can make boundaries of each class. These boundaries will segregate RC from GS. The same way, let's try to see the effect of value "K" on the class boundaries. Following are the different boundaries separating the two classes with different values of K.



If you watch carefully, you can see that the boundary becomes smoother with increasing value of K . With K increasing to infinity it finally becomes all blue or all red depending on the total majority. The training error rate and the validation error rate are two parameters we need to access on different K -value. Following is the curve for the training error rate with varying value of K :



As you can see, the error rate at $K=1$ is always zero for the training sample. This is because the closest point to any training data point is itself. Hence the prediction is always accurate with $K=1$. If validation error curve would have been similar, our choice of K would have been 1. Following is the validation error curve with varying value of K :



This makes the story more clear. At $K=1$, we were over fitting the boundaries. Hence, error rate initially decreases and reaches a minimal. After the minima point, it then increases with increasing K . To get the optimal value of K , you can segregate the training and validation from the initial dataset. Now plot the validation error curve to get the optimal value of K . This value of K should be used for all predictions.

Breaking it down – Pseudo Code of KNN

We can implement a KNN model by following the below steps:

- Load the data
- Initialize the value of k
- For getting the predicted class, iterate from 1 to total number of training data points
 - Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
 - Sort the calculated distances in ascending order based on distance values
 - Get top k rows from the sorted array
 - Get the most frequent class of these rows
 - Return the predicted class

Conclusion

KNN algorithm is one of the simplest classification algorithms. Even with such simplicity, it can give highly competitive results. KNN algorithm can also be used for regression problems. The only difference from the discussed methodology will be using averages of nearest neighbors rather than voting from nearest neighbors.

CHAPTER 4

4.1 Design and Implementation Constraints

4.1.1 Constraints in Analysis

- Constraints as Informal Text
- Constraints as Operational Restrictions
- Constraints Integrated in Existing Model Concepts

- Constraints as a Separate Concept
- Constraints Implied by the Model Structure

4.1.2 Constraints in Design

- Determination of the Involved Classes
- Determination of the Involved Objects
- Determination of the Involved Actions
- Determination of the Require Clauses
- Global actions and Constraint Realization

4.1.3 Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object oriented methods.

• Other Nonfunctional Requirements

4.2.1 Performance Requirements

The application at this side controls and communicates with the following three main general components.

- embedded browser in charge of the navigation and accessing to the web service;
- Server Tier: The server side contains the main parts of the functionality of the proposed architecture. The components at this tier are the following.

Web Server, Security Module, Server-Side Capturing Engine, Preprocessing Engine, Database System, Verification Engine, Output Module.

4.2.2 Safety Requirements

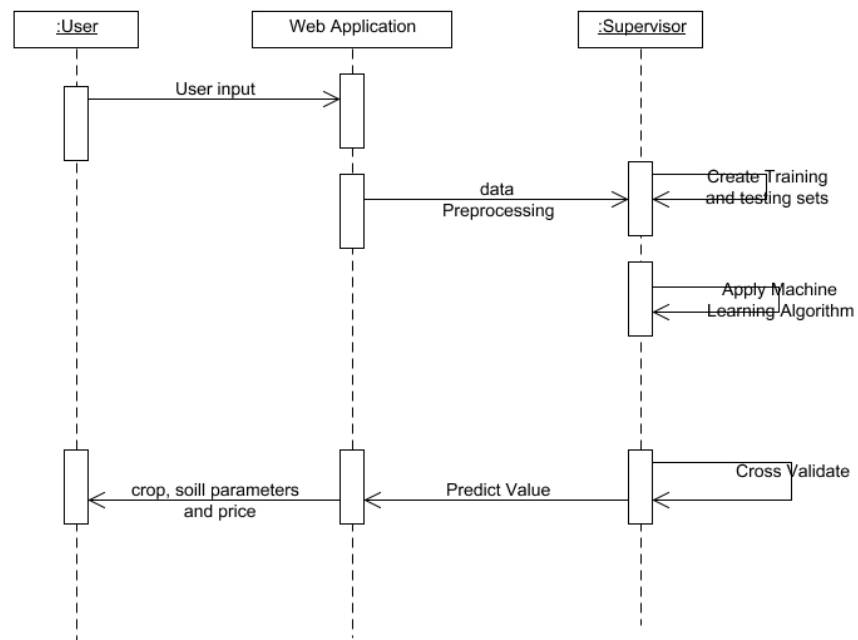
- The software may be safety-critical. If so, there are issues associated with its integrity level
- The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions.
- If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.
- There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.
- If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.
- Systems with different requirements for safety levels must be separated.
- Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

CHAPTER 5

5.1 Architecture Diagram:

5.2 Sequence Diagram:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.



5.3 Use Case Diagram:

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

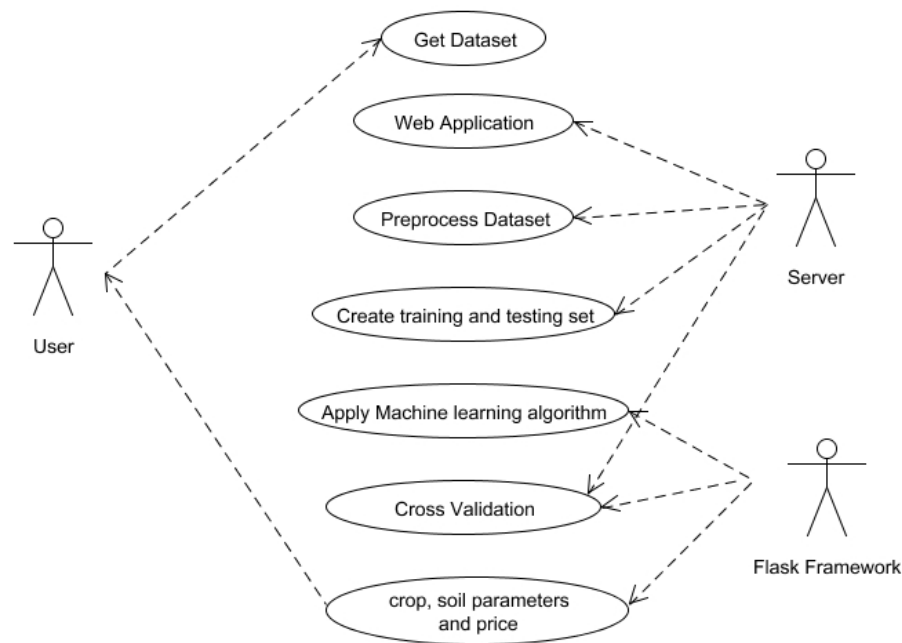
5.3.1. USECASE DIAGRAM

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

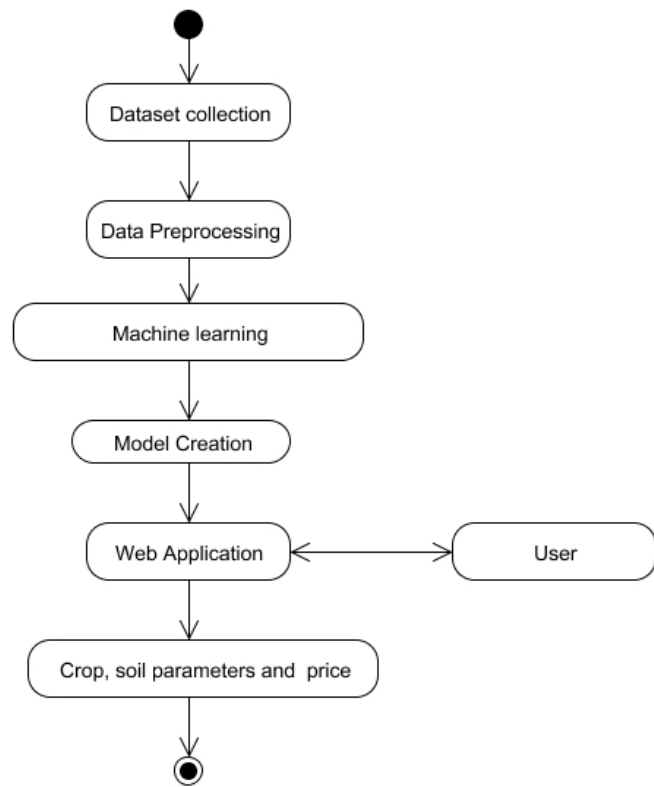


5.4 Activity Diagram:

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

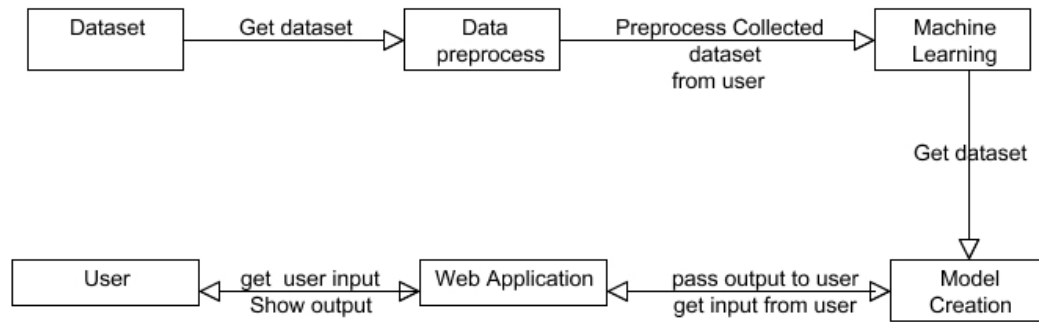
The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.



5.5 Collaboration Diagram:

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.



CHAPTER 6

6.1 MODULES

Module Description:

Dataset collection

Implementation

Prediction

6.2 MODULE EXPLANATION:

6.2.1 Dataset Collection:

Our crop prediction project dataset are collected from kaggle.com. Data is pre-processed after collection of various records. The dataset contains a more number of records, where some records are with some missing values. Those missing records have been removed from the dataset and the remaining records are used in pre-processing.

Implementation:

The Classification Algorithms to produce the best results. We are using Random Forest Algorithm to predict the crop using ML. On an analysis conducted within various algorithms, the Random Forest was found to provide highest efficiency and precision compared to Decision tree. Because RF contain number of decision tree algorithm, that take the average to improve the predictive accuracy of dataset. Hence the RF algorithm is used in the proposed system to find the suitable crop.

Prediction:

Preprocessed data are trained and input given by the user goes to the trained dataset. After prediction the predict value given as an output on web application (Flask Framework).

CHAPTER 7

CODING AND TESTING

7.1 CODING

Once the design aspect of the system is finalized the system enters into the coding and testing phase. The coding phase brings the actual system into action by converting the design of the system into the code in a given programming language. Therefore, a good coding style has to be taken whenever changes are required it easily screwed into the system.

7.2 CODING STANDARDS

Coding standards are guidelines to programming that focuses on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

- Program should be simple, clear and easy to understand.

- Naming conventions

- Value conventions

- Script and comment procedure

- Message box format

- Exception and error handling

7.2.1 NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The conventions are adopted for **easy understanding** of the intended message by the user. So it is customary to follow the conventions. These conventions are as follows:

Class names

Class names are problem domain equivalence and begin with capital letter and have mixed cases.

Member Function and Data Member name

Member function and data member name begins with a lowercase letter with each subsequent letters of the new words in uppercase and the rest of letters in lowercase.

7.2.2 VALUE CONVENTIONS

Value conventions ensure values for variable at any point of time. This involves the following:

- Proper default values for the variables.
- Proper validation of values in the field.
- Proper documentation of flag values.

7.2.3 SCRIPT WRITING AND COMMENTING STANDARD

Script writing is an art in which indentation is utmost important. Conditional and looping statements are to be properly aligned to facilitate easy understanding. Comments are included to minimize the number of surprises that could occur when going through the code.

7.2.4 MESSAGE BOX FORMAT

When something has to be prompted to the user, he must be able to understand it properly. To achieve this, a specific format has been adopted in displaying messages to the user. They are as follows:

- X – User has performed illegal operation.
- ! – Information to the user.

7.3 TEST PROCEDURE

SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

7.4 TEST DATA AND OUTPUT

7.4.1 UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

7.4.2 FUNCTIONAL TESTS

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

7.4.3 PERFORMANCE TEST

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

7.4.4 STRESS TEST

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

7.4.5 STRUCTURED TEST

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.

- Exercise internal data structures to assure their validity.
- Checking attributes for their correctness.
- Handling end of file condition, I/O errors, buffer problems and textual errors in output information

7.4.6 INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

7.5 TESTING TECHNIQUES / TESTING STRATERGIES

7.5.1 TESTING

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet –undiscovered error. A successful

test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding an yet undiscovered error. A successful test is one that uncovers an yet undiscovered error. Any engineering product can be tested in one of the two ways:

7.5.1.1 WHITE BOX TESTING

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been design to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

- Flow graph notation
- Cyclometric complexity
- Deriving test cases

- Graph matrices Control

7.5.1.2 BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

- Graph based testing methods
- Equivalence partitioning
- Boundary value analysis
- Comparison testing

7.5.2 SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

- Testing begins at the module level and works “outward” toward the integration of the entire computer based system.
- Different testing techniques are appropriate at different points in time.
- The developer of the software and an independent test group conducts testing.
- Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

7.5.2.1 INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when we put them together. The problem of course, is “putting them together”- interfacing. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

7.5.2.2 PROGRAM TESTING:

The logical and syntax errors have been pointed out by program testing. A syntax error is an error in a program statement that in violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax error. These errors are shown through error messages generated by the computer. A logic error on the other hand deals with the incorrect data fields, out-off-range items and invalid combinations. Since the compiler s will not deduct logical error, the programmer must examine the output. Condition testing exercises the logical conditions contained in a module. The possible types of elements in a condition include a Boolean operator, Boolean variable, a pair of Boolean parentheses A relational operator or on arithmetic expression. Condition testing method focuses on testing each condition in the program the purpose of condition test is to deduct not only errors in the condition of a program but also other a errors in the program.

7.5.2.3 SECURITY TESTING:

Security testing attempts to verify the protection mechanisms built in to a system well, in fact, protect it from improper penetration. The system security must be tested for invulnerability from frontal attack must also be tested for invulnerability from rear attack. During security, the tester places the role of individual who desires to penetrate system.

7.5.2.4 VALIDATION TESTING

At the culmination of integration testing, software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test-validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in manner that is reasonably expected by the customer. Software validation is achieved through a series of black box tests that demonstrate conformity with requirement. After validation test has been conducted, one of two conditions exists.

- * The function or performance characteristics confirm to specifications and are accepted.
- * A validation from specification is uncovered and a deficiency created.

Deviation or errors discovered at this step in this project is corrected prior to completion of the project with the help of the user by negotiating to establish a method for resolving deficiencies. Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily. Though there were deficiencies in the system they were not catastrophic

7.5.2.5 USER ACCEPTANCE TESTING

User acceptance of the system is key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system and user at the time of developing and making changes whenever required. This is done in regarding to the following points.

- Input screen design.
- Output screen design.

Source Code

```
from flask import Flask ,render_template,request,jsonify,session
```

```
import sqlite3 as sql
```

```
import base64
```

```
import pandas as pd
```

```
from sklearn.preprocessing import LabelEncoder
```

```
#from flask_bootstrap import Bootstrap
```

```
import numpy as np
```

```
from sklearn.utils import shuffle
```

```
import joblib
```

```
app = Flask(__name__)
```

```
app.secret_key = 'any random string'
```

```
@app.route('/', methods=['GET', 'POST'])
```

```
def home():
```

```
    return render_template('index.html')
```

```
def validate(username,password):
```

```
    con = sql.connect('static/chat.db')
```

```
    completion = False
```

```
with con:
```

```
    cur = con.cursor()
```

```
    cur.execute('SELECT * FROM persons')
```

```
    rows = cur.fetchall()
```

```
    for row in rows:
```

```
        dbuser = row[1]
```

```
        dbpass = row[2]
```

```
        if dbuser == username:
```

```
            completion = (dbpass == password)
```

```
    return completion
```

```
@app.route('/login', methods=['GET', 'POST'])
```

```
def login():
```

```
    error = None
```

```
    if request.method == 'POST':
```

```
        username = request.form['username']
```

```
        password = request.form['password']
```

```
        completion = validate(username,password)
```

```
        if completion == False:
```

```
            error = 'invalid Credentials. please try again.'
```

```
        else:
```

```
            session['username'] = request.form['username']
```

```
        return render_template('index111.html')

    return render_template('index.html', error=error)


@app.route('/view', methods=['GET', 'POST'])
def view():

    return render_template('index111.html')


@app.route('/cropa', methods=['GET', 'POST'])
def cropa():

    return render_template('cropdetails.html')


@app.route('/about', methods=['GET', 'POST'])
def about():

    return render_template('About.html')
```

```
@app.route('/register', methods = ['GET','POST'])
def register():
    if request.method == 'POST':
        try:
            name = request.form['name']
            username = request.form['username']
            password = request.form['password']
            with sql.connect("static/chat.db") as con:
                cur = con.cursor()
                cur.execute("INSERT INTO persons(name,username,password) VALUES
(?,?,?)",(name,username,password))
                con.commit()
                msg = "Record successfully added"
        except:
            con.rollback()
            msg = "error in insert operation"
        finally:
            return render_template("index.html",msg = msg)
            con.close()
    return render_template('register.html')
```

```
@app.route('/list')

def list():

    con = sql.connect("static/chat.db")

    con.row_factory = sql.Row

    cur = con.cursor()

    cur.execute("select * from persons")

    rows = cur.fetchall();

    return render_template("list.html",rows = rows)

@app.route('/crop_predict',methods=['GET','POST'])

def cr():

    lt= []

    lst = []

    data = pd.read_csv(r'dataset\Crop.csv')

    city = pd.read_csv(r'dataset\Ploted_600.csv')

    # data = shuffle(data)

    y = data.loc[:,'Crop']

    labelEncoder_y = LabelEncoder()

    y = labelEncoder_y.fit_transform(y)
```

```
data['crop_num'] = y

X = data.loc[:,['N','P','K','pH']].astype(float)

y = data.loc[:, 'crop_num']


# Training Model

from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier(n_jobs=10, n_neighbors=10, weights='distance')

clf.fit(X,y)


if request.method == 'POST':

    city_name = request.form['city']

    N = request.form['Nitrogen']

    P = request.form['Phosphorous']

    K = request.form['Potassium']

    pH = request.form['pH']


if len(city_name) != 0:

    print(city_name)


    npk = city[city['Location'] == city_name]
```

```

val = []

for index, row in npk.iterrows():

    val = [row['N'],row['P'],row['K'],row['pH']]

    print(val)

    columns = ['N','P','K','pH']

    values = np.array([val[0],val[1],val[2],val[3]])

    pred = pd.DataFrame(values.reshape(-1, len(values)),columns=columns)


    prediction = clf.predict(pred)

    # distances, indices = clf.kneighbors(pred, n_neighbors=10)

    # prediction

    real_pred = clf.predict_proba(pred)

    print(real_pred)


    lst = []

    for i in range(101):

        if real_pred[0][i] != 0.0:

            lst.append(i)


    lt= []

    for i in range(10):

```

```

        load_data = data[data.index == lst[i]]

        for index, row in load_data.iterrows():

            lt.append(row['Crop'])

else:

    print(N,P,K,pH)

    columns = ['N','P','K','pH']

    values = np.array([N,P,K,pH])

    pred = pd.DataFrame(values.reshape(-1, len(values)),columns=columns)

    prediction = clf.predict(pred)

    # distances, indices = clf.kneighbors(pred, n_neighbors=10)

    # prediction

    real_pred = clf.predict_proba(pred)


for i in range(101):

    if real_pred[0][i] != 0.0:

        lst.append(i)


for i in range(10):

    load_data = data[data.index == lst[i]]

```

```

        for index, row in load_data.iterrows():

            lt.append(row['Crop'])

            # print(city,N,P,K,pH)

        return render_template('crop.html',crops=lt,crop_num = len(lt),display=True)


@app.route('/crop_pre',methods=['GET','POST'])
def crop():

    val_final=[]

    data = pd.read_csv(r'dataset\Crop.csv')

    #print(data)

    city = pd.read_csv(r'dataset\Ploted_600.csv')

    #print(city)

    X = data.loc[:,['N','P','K','pH']]

    y = data.loc[:,['Crop']]

    X = data.loc[:,['N','P','K','pH']].astype(float)

    from sklearn.neighbors import KNeighborsClassifier

    clf = KNeighborsClassifier(n_jobs=10, n_neighbors=10,weights='distance')

```

```
clf.fit(X,y)
```

```
lst = []
```

```
lt = []
```

```
if request.method == 'POST':
```

```
    crop = request.form['crop']
```

```
    N1 = request.form['Nitrogen']
```

```
    P1 = request.form['Phosphorous']
```

```
    K1 = request.form['Potassium']
```

```
    pH1 = request.form['pH']
```

```
#crop = str(input('Crop_name'))
```

```
if len(crop) != 0:
```

```
    print(crop)
```

```
    npk = data[data['Crop'] == crop]
```

```
val = []

for index, row in npk.iterrows():

    val = [row['N'],row['P'],row['K'],row['pH']]

    print(val)

    a=float(N1)

    b=float(P1)

    c=float(K1)

    d=float(pH1)


    val1=tuple(val)

    #val2=np.array(val0)

    val0=(a,b,c,d)

    val2=tuple(val0)

    val_final=(val1[0]-val2[0],val1[1]-val2[1],val1[2]-val2[2],val1[3]-val2[3])


df=pd.DataFrame(val_final,columns=['as'])


for i in val_final:
```

```
lst.append(i)
```

```
for i in range(4):
```

```
    load_data = df[df.index == lst[i]]
```

```
    for index, row in load_data.iterrows():
```

```
        lt.append(row['as'])
```

```
return render_template('soil_parameters.html',crops=lst,val_final=len(lst),display=True)
```

```
@app.route('/crop_price',methods=['GET','POST'])
```

```
def price():
```

```
    lst=[]
```

```
    if request.method == 'POST':
```

```
        Location = request.form['Location']
```

```
        date = request.form['date']
```

```
        temp = request.form['temp']
```

```
Humidity = request.form['Humidity']

pH = request.form['pH']

rainfall = request.form['rainfall']

test_data=[temp,Humidity,pH,rainfall]


# Load the model from the file

ranfor_from_joblib = joblib.load(r'C:/Users/ernest/Intelligent Crop Recommendation
System/M3/Models/ranfor_model1.pkl')


# Use the loaded model to make predictions

prediction_crop=ranfor_from_joblib.predict([test_data])


# Load the model from the file

ranfor_from_joblib1 = joblib.load(r'C:/Users/ernest/Intelligent Crop Recommendation
System/M3/Models/ranfor_model2.pkl')


# Use the loaded model to make predictions

prediction_price=ranfor_from_joblib1.predict([test_data])


for dat in [prediction_crop,prediction_price]:

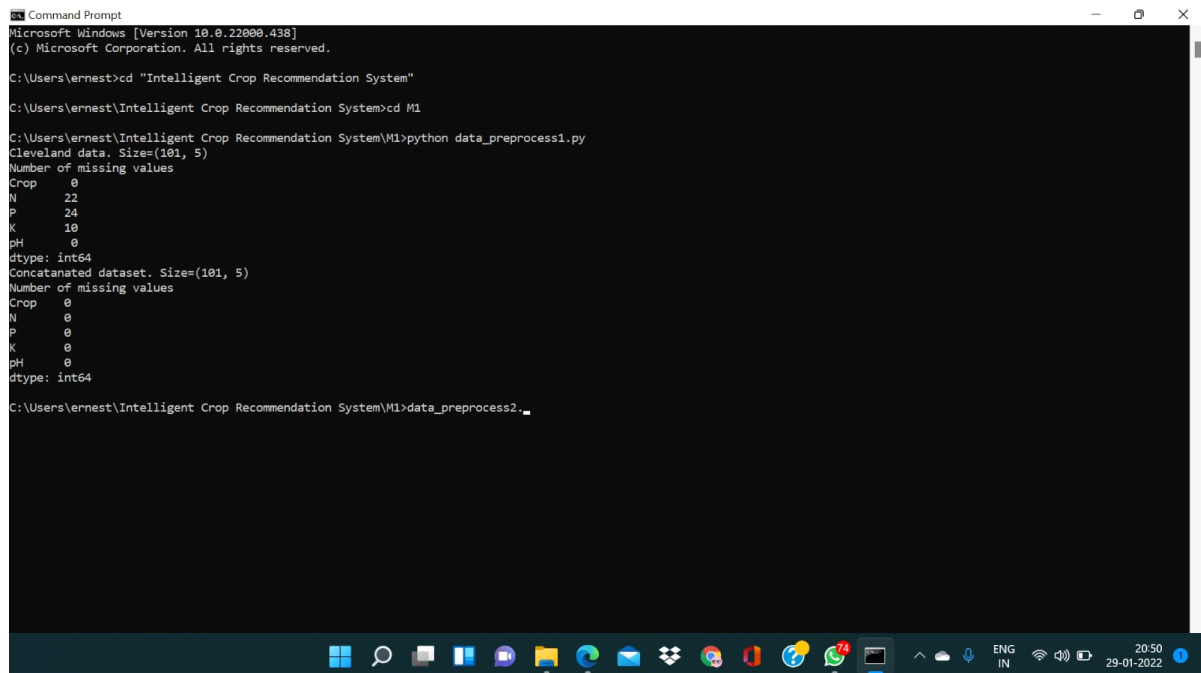
    lst.append(dat)


return render_template('price.html',crops=lst,display=True)
```

```
if __name__ == '__main__':  
  
    app.run(debug = True )
```

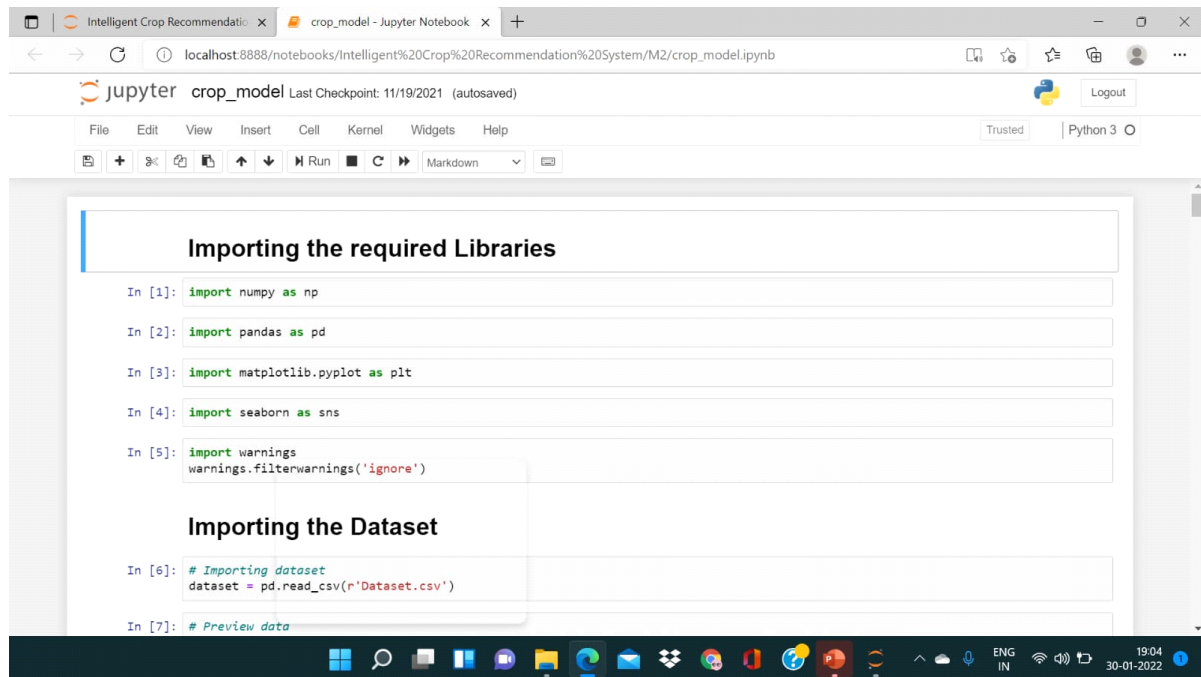
Screenshots:

DATASET COLLECTION



```
Command Prompt  
Microsoft Windows [Version 10.0.22000.438]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\ernest>cd "Intelligent Crop Recommendation System"  
  
C:\Users\ernest\Intelligent Crop Recommendation System>cd M1  
  
C:\Users\ernest\Intelligent Crop Recommendation System\M1>python data_preprocess1.py  
Cleveland data. Size=(101, 5)  
Number of missing values  
Crop      0  
N         22  
P         24  
K         10  
pH         0  
dtype: int64  
Concatanated dataset. Size=(101, 5)  
Number of missing values  
Crop      0  
N         0  
P         0  
K         0  
pH         0  
dtype: int64  
  
C:\Users\ernest\Intelligent Crop Recommendation System\M1>data_preprocess2._
```

IMPLEMENTATION



The screenshot displays a Jupyter Notebook titled 'crop_model' running on a local host. The interface includes a top navigation bar with the Jupyter logo, the notebook name, and a 'Logout' button. Below this is a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A toolbar with various icons for cell manipulation is also present. The notebook content is divided into two sections: 'Importing the required Libraries' and 'Importing the Dataset'. The first section contains five code cells for importing numpy, pandas, matplotlib, seaborn, and warnings. The second section contains two code cells for loading a CSV file and previewing the data. The Windows taskbar is visible at the bottom of the screen.

```
In [1]: import numpy as np

In [2]: import pandas as pd

In [3]: import matplotlib.pyplot as plt

In [4]: import seaborn as sns

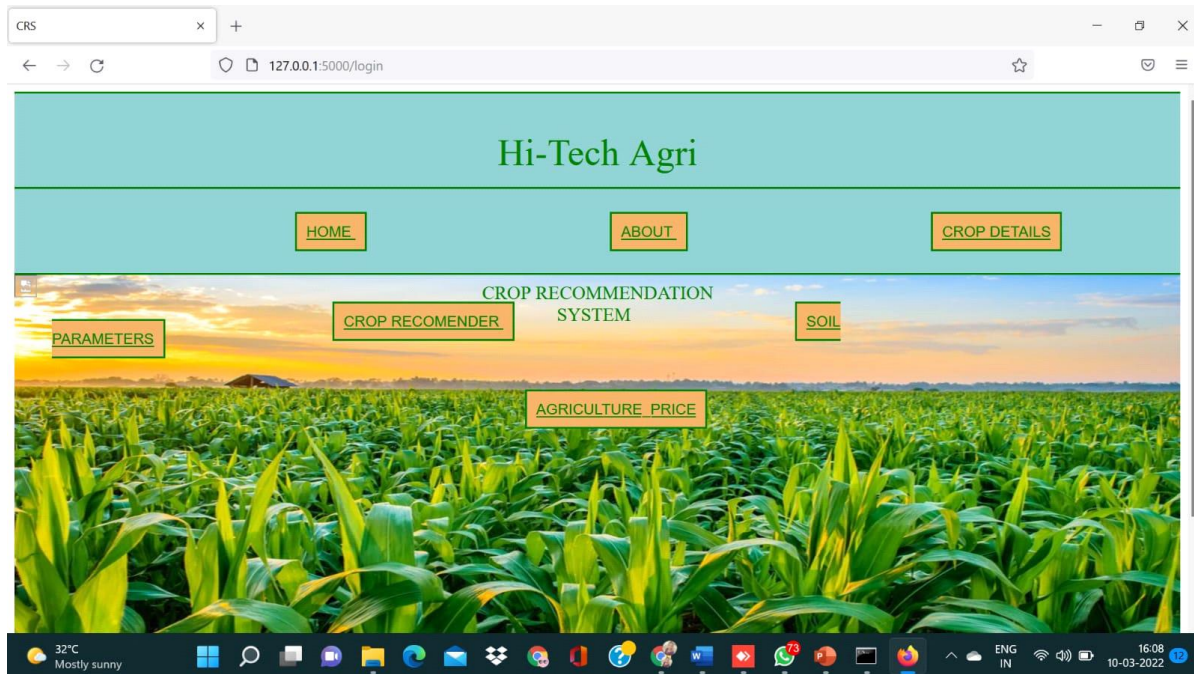
In [5]: import warnings
warnings.filterwarnings('ignore')

Importing the Dataset

In [6]: # Importing dataset
dataset = pd.read_csv(r'Dataset.csv')

In [7]: # Preview data
```

PREDICTION



REFERENCES

- [1] Niketa Gandhi et al , " Rice Crop Yield Forecasting of Tropical Wet and Dry Climatic Zone of India Using Data Mining Techniques",IEEE International Conference on Advances in Computer Applications (ICACA) , 2016.
- [2] K.E. Eswari. L.Vinitha. " Crop Yield Prediction in Tamil Nadu Using Bayesian Network ", International Journal of Intellectual Advancements and Research in Engineering Computations, Volume-6,Issue-2,ISSN: 2348-2079.
- [3] Shruti Mishra, Priyanka Paygude, Snehal Chaudhary, Sonali Idate “Use of Data Mining in Crop Yield Prediction” IEEE Xplore Compliant – Part Number:CFP18J06-ART, ISBN:978-1-5386-0807-4; DVD PartNumber: CFP18J06DVD , ISBN:978-1-5386-0806-7.

- [4] Anna Chlingaryana, Salah Sukkarieha, Brett Whelanb — Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review , Computers and Electronics in Agriculture 151 (2018) 61–69,Elisver,2018.
- [5] Dakshayini Patil et al,"Rice Crop Yield Prediction using Data Mining Techniques:An Overview", International Journal of Advanced Research in Computer Science and Software Engineering ,Volume 7, Issue 5, May 2017.
- [6] A.T.M Shakil Ahamed, Navid Tanzeem Mahmood, Nazmul Hossain, Mohammad Tanzir Kabir, Kallal Das, Faridur Rahman, Rashedur M Rahman “Applying Data Mining Techniques to Predict Annual Yield of Major Crops and Recommend Planting Different Crops in Different Districts in Bangladesh” 978-1-4799-8676-7/15/\$31.00 copyright 2015 IEEE SNPD 2015, June 1-3 2015.
- [7] K. Menaka, N.Yuvaraj,Indian Journal of Innovations and Developments Vol 5(12), October, 2016 ISSN (online) : 2277-5390 ISSN (Print): 2277- 5382
- [8] Machine Learning by Tom M. Mitchell, ISBN: 0070428077
- [9] Snehal S.Dahikar, Dr.Sandeep V.Rode , "Agricultural Crop Yield Prediction Using Artificial Neural Network Approach ", International journal of innovative and research in electrical, instrumentation and control engineering, volume 2,Issue 2,2014.

- [10] D Ramesh, B Vishnu Vardhan, "Analysis of crop yield prediction using data mining techniques", International Journal of research in engineering and technology,eISSN:2319-1163.
- [11] Ramesh A. Medar, "A survey on data mining techniques for crop yield prediction", International Journal of advance in computer science and management studies, ISSN:2231-7782, volume 2, Issue 9, 2014.
- [12] Pooja M C et al, —Implementation of Crop Yield Forecasting Using Data Mining , International Research Journal of Engineering and Technology (IRJET),2018,p-ISSN: 2395-0072
- [13] Jig Han Jeong et al , “Random Forests for Global and Regional Crop Yield Predictions ”,PLOS-ONE DOI:10.1371/journal.pone.0156571 June 3, 2016.
- [14] Jharna Majumdar, Sneha Naraseeyappa and Shilpa Ankalaki, Analysis of agriculture data using data mining techniques: application of big data, Springer journal,2017.
- [15] S.kanaga Subba Raju et al, Demand based crop recommender system for farmers, International Conference on Technological Innovations in ICT For Agriculture and Rural Development, 2017.
- [16] Shriya Sahu et al," An Efficient Analysis of Crop Yield Prediction Using Hadoop framework based on random Forest approach, International Conference on Computing, Communication and Automation , 2017.

[17] E Manjula and S. Djodkitachoumy International journal of computation
Intelligence and informatics, Vol 6:No 4 March 2017