# Soil Nutrients Based Crop Recommendation Using Machine Learning

A mini project report submitted in partial fulfillment of requirement for award of VI semester of

**BACHELOR OF TECHNOLOGY**

In

**ELECTRONICS & COMMUNICATION ENGINEERING**

By


| | |
|---|---|
| **NAKKA BHAVITH** | **(19K41A0450)** |
| **PARIKIPANDLA SUJEETH** | **(19K41A0452)** |
| **RAGHAV BANG** | **(19K41A0454)** |
| **THEDLA ANAND** | **(19K41A0459)** |
| **VISRAM APUROOP** | **(18K41A04H2)** |

Under the guidance of

**Dr. Syed Musthak Ahmed**

Professor in ECE & Dean (Students Affairs)

**S R ENGINEERING COLLEGE**

Ananthasagar, Warangal

## CERTIFICATE

This is to certify that this project entitled "**SOIL NUTRIENTS BASED CROP RECOMMENDATION USING MACHINE LEARNING**" is the bonafide work carried out by N. Bhavith, P. Sujeeth, Raghav Bang, V. Apuroop and T. Anand as a minor project in the partial fulfillment of the requirement for the award of degree of VI semester of BACHELOR OF TECHNOLOGY in ELECTRONICS & COMMUNICATION ENGINEERING during the academic year 2021-2022 under our guidance and Supervision.

**Dr. Syed Musthak Ahmed**

Professor in ECE, Dean (SA) & Guide,

S R Engineering College,

Ananthasagar, Warangal.

**Dr. Sandip Bhattachatya**

Associate Professor & HOD

S R Engineering College,

Ananthasagar, Warangal.

# ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and a deep sense of respect to our beloved principal, Dr. V. Mahesh, S R Engineering College, Ananthasagar, Warangal, for making us available all the required assistance and for his support and inspiration to carry out this work in the institute.

We owe an enormous debt of gratitude to Dr. Sandip Bhattachatya, Associate Professor & HOD of ECE Department and as well as our project guide Dr. Syed Musthak Ahmed, Professor & Dean (Student Affairs) for guiding us from the beginning through the end of the project with their intellectual advices and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

I also thank the other staff members and friends assisted us. Finally, I thank my parents who inspired us always to do the best.

# ABSTRACT

India was reportedly having 60.43% of agricultural land and 58% of the population dependent on it. The farmers have enough resources though they are unable to produce up to the expected level of yield. One of the main reasons behind it is growing crops that don’t fulfill the requirements of soil conditions. As we know that soil is one of the most important natural resources and plays a major role in crop production. Soil nutrients like nitrogen, potassium, and phosphorous are very important for any crop growth, the farmers are doing mistakes in choosing the crop even after having soil data that are tested by government labs. To eliminate this problem we proposed a solution i.e web application based on Artificial Intelligence and machine learning, which takes contents of soil as an input from a farmer and recommends an appropriate crop, the machine learning model we used is the Random forest classifier algorithm to get better accuracy. Thus, the application helps to supply information to the dilettante farmers.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1

# INTRODUCTION

## 1.1 EXISTING SOLUTION

In some existing works they forecasts weather and recommend crops based on future weather condition. So this type crops recommendation helps farmers related on crops and weather forecasting. But important in agriculture is, all crops production is based on the soils because soils are fundamental to agriculture development and crops production. If soil is not suitable for particular crop, farmers can't get profit production. So recommend the crops with forecasting of weather and related on soil will help to farmers for efficiently identify suitable crops.

**Issues in existing system:**

- Recommending crops only with on weather prediction.
- Farmers get general information of crops.
- It may lead low production and profit.

## 1.2 PROPOSED SOLUTION

The proposed solution makes use of machine learning algorithm to predict a suitable crop on the basis of soil nutrients and various other parameters. All you need is to take the observations of the soil and enter the observations in their respective fields provided in our website. And this website takes the soil data as input, undergoes through various Machine learning models in the background and thus the resulted prediction is shown as output recommending the suitable crop. This proposal is effective because it provides a clear and specific solution to the problems faced by farmers.

**Advantages over existing system:**

- Provide suitable crop recommendation to farmers.
- It provides more production and profit.

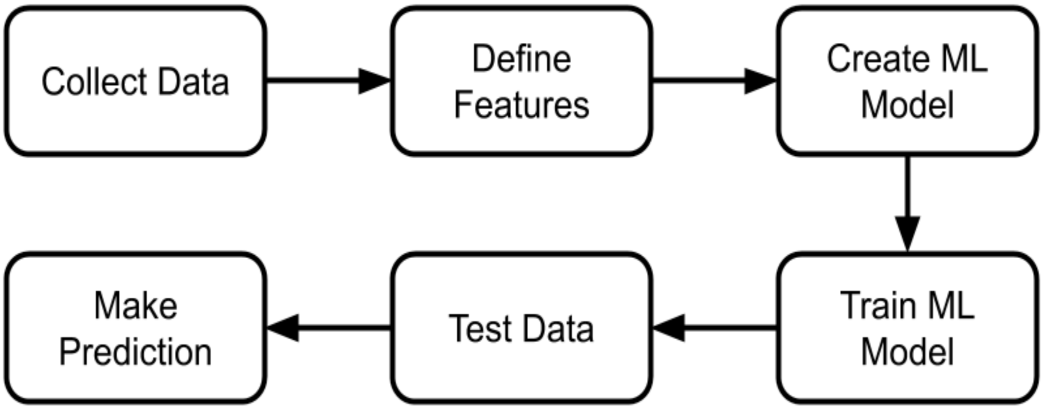


**Figure 1:** Block diagram for proposed solution

### 1.3 LITERATURE REVIEW

Literature review on nurturing technologies employed in agriculture was carried out. Here, glimpse on ML, deep learning, sensing technologies deployed in agriculture were provided.

- S. Vidhya in their paper proposes a Crop Selection Method (CSM) which solves the crop selection problem and improves net yield rate of the crop. It suggests a series of crop to be selected over a season considering factors like weather, soil type, water density, crop type. The predicted value of influential parameters determines the accuracy of CSM. Hence there exists a need to include a prediction method with improved accuracy and performance.
- Marc Sebban and Richard Nock, analysed the filter model with information gain and a statistical test. A hybrid model was implemented using a minimum spanning tree that was replaced by the first nearest neighbour. Lei Yu and Huan Liu, proposed a correlation filter method termed the fast correlation-based filter. Their technique was verified by two different classification algorithms in terms of real-world data, with and without feature selection.
- Maya Gopal and Bhargavi, proposed a wrapper feature selection method featuring Boruta that extracts features from a dataset for crop prediction. The technique improves prediction performance and provides effective predictors. In Boruta, the Z score has the most accurate measure, since it takes into consideration the variability of the mean loss of accuracy among trees in a forest.

# CHAPTER 2

# SOFTWARE TOOLS

## 2.1 ABOUT PYTHON

Python is an Interpreted, high-level and general-purpose programming language. It has a rich arrangement of libraries and tools that makes the assignments simple for Data scientists. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming. Python is often described as a “batteries included” language due to its comprehensive standard library. Data Scientists need to manage a large amount of data known as big data. With simple utilization and a huge arrangement of python libraries, python has become a popular choice to deal with big data.

## 2.2 FLASK APP

A flask application is an instance of the Flask class. Everything about the application, such as configuration and URLs, will be registered with this class. It is a micro web framework written in python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload-handling, various open authentication technologies and several common framework related tools. The main features of Flask are Development server and debugger, Integrated support for unit testing, RESTfull request dispatching, uses jinja templating, support for secure cookies (client side sessions), 100% WSGI (Web Server Gateway Interface) 1.0 client, unicode-based and extensive documentation.

# CHAPTER 3

# PROJECT IMPLEMENTATION

## 3.1 DATASET INSIGHTS

The team have collected the data from various open source sites like towardsdatascience, kaggle, freecodecamp etc.

The dataset consists of 2200 rows and 8 columns of data.

| | NITROGEN | PHOSPHORUS | POTASSIUM | TEMPERATURE | HUMIDITY | PH | RAINFALL | CROP |
|---|---|---|---|---|---|---|---|---|
| 0 | 90 | 42 | 43 | 20.879744 | 82.002744 | 6.502985 | 202.935536 | rice |
| 1 | 85 | 58 | 41 | 21.770462 | 80.319644 | 7.038096 | 226.655537 | rice |
| 2 | 60 | 55 | 44 | 23.004459 | 82.320763 | 7.840207 | 263.964248 | rice |
| 3 | 74 | 35 | 40 | 26.491096 | 80.158363 | 6.980401 | 242.864034 | rice |
| 4 | 78 | 42 | 42 | 20.130175 | 81.604873 | 7.628473 | 262.717340 | rice |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2195 | 107 | 34 | 32 | 26.774637 | 66.413269 | 6.780064 | 177.774507 | coffee |
| 2196 | 99 | 15 | 27 | 27.417112 | 56.636362 | 6.086922 | 127.924610 | coffee |
| 2197 | 118 | 33 | 30 | 24.131797 | 67.225123 | 6.362608 | 173.322839 | coffee |
| 2198 | 117 | 32 | 34 | 26.272418 | 52.127394 | 6.758793 | 127.175293 | coffee |
| 2199 | 104 | 18 | 30 | 23.603016 | 60.396475 | 6.779833 | 140.937041 | coffee |

2200 rows × 8 columns

**Figure 2:** Dataset

**Units for different features:**

Nitrogen: mg/kg

Phosphorus: mg/kg

Potassium: mg/kg

Temperature: °C

Humidity: grams/$m^3$

PH: --

Rainfall: mm

**Using data.info( ):**

```
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   NITROGEN     2200 non-null   int64
 1   PHOSPHORUS   2200 non-null   int64
 2   POTASSIUM    2200 non-null   int64
 3   TEMPERATURE  2200 non-null   float64
 4   HUMIDITY     2200 non-null   float64
 5   PH           2200 non-null   float64
 6   RAINFALL     2200 non-null   float64
 7   CROP         2200 non-null   object
dtypes: float64(4), int64(3), object(1)
```

**Figure 3:** Information of dataset

**Using data.describe( ):**

| | NITROGEN | PHOSPHORUS | POTASSIUM | TEMPERATURE | HUMIDITY | PH | RAINFALL |
|---|---|---|---|---|---|---|---|
| count | 2200.000000 | 2200.000000 | 2200.000000 | 2200.000000 | 2200.000000 | 2200.000000 | 2200.000000 |
| mean | 50.551818 | 53.362727 | 48.149091 | 25.616244 | 71.481779 | 6.469480 | 103.463655 |
| std | 36.917334 | 32.985883 | 50.647931 | 5.063749 | 22.263812 | 0.773938 | 54.958389 |
| min | 0.000000 | 5.000000 | 5.000000 | 8.825675 | 14.258040 | 3.504752 | 20.211267 |
| 25% | 21.000000 | 28.000000 | 20.000000 | 22.769375 | 60.261953 | 5.971693 | 64.551686 |
| 50% | 37.000000 | 51.000000 | 32.000000 | 25.598693 | 80.473146 | 6.425045 | 94.867624 |
| 75% | 84.250000 | 68.000000 | 49.000000 | 28.561654 | 89.948771 | 6.923643 | 124.267508 |
| max | 140.000000 | 145.000000 | 205.000000 | 43.675493 | 99.981876 | 9.935091 | 298.560117 |

**Figure 4:** Analysis of dataset

**Using data.nunique( ):**

```
NITROGEN        137
PHOSPHORUS      117
POTASSIUM        73
TEMPERATURE    2200
HUMIDITY       2200
PH             2200
RAINFALL       2200
CROP             22
dtype: int64
```

**Figure 5:** No. of Unique values in each column

### 3.1.1 DATA VISUALIZATION

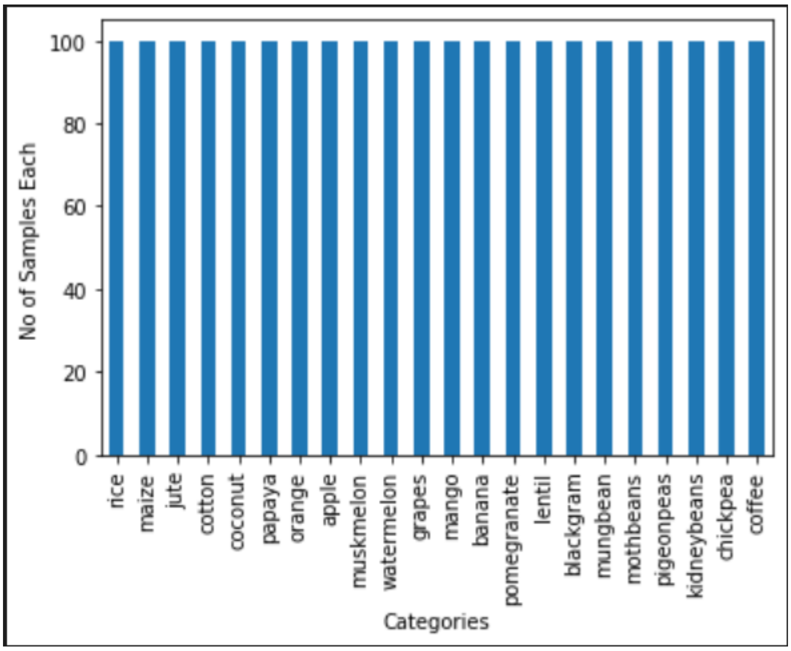


**Figure 6:** No. of Categories vs No. of samples

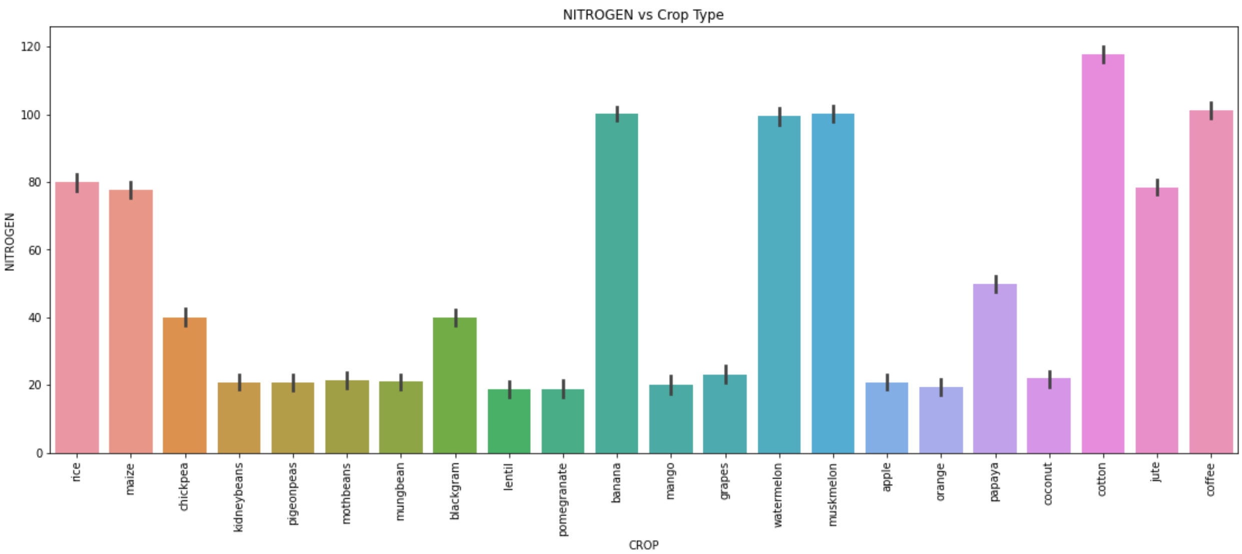


**Figure 7:** Crop Type vs Nitrogen

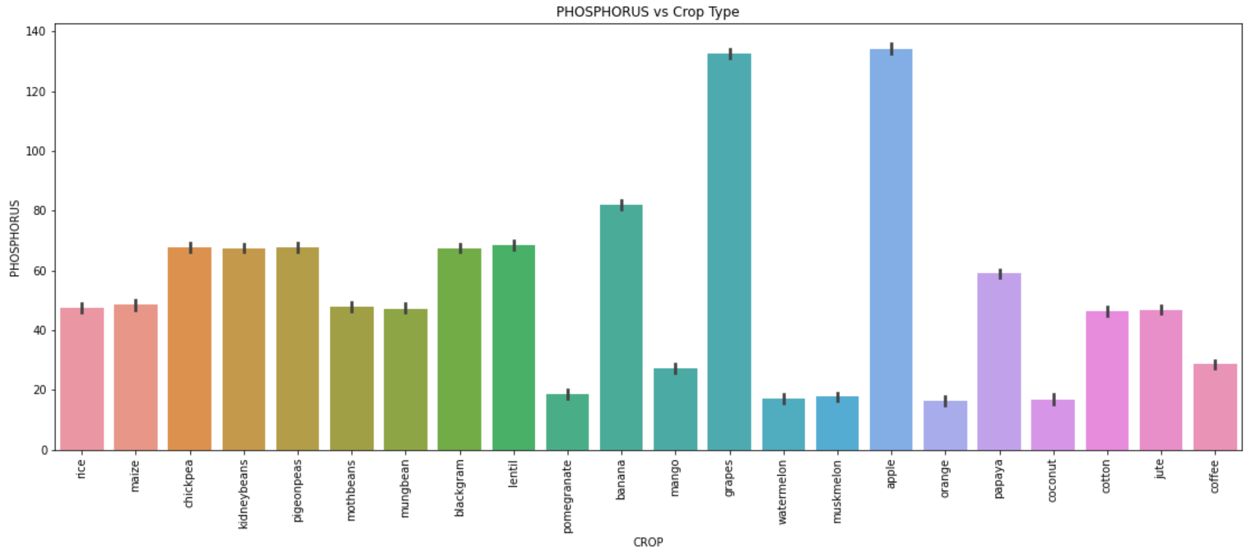


**Figure 8:** Crop Type vs Phosphorus

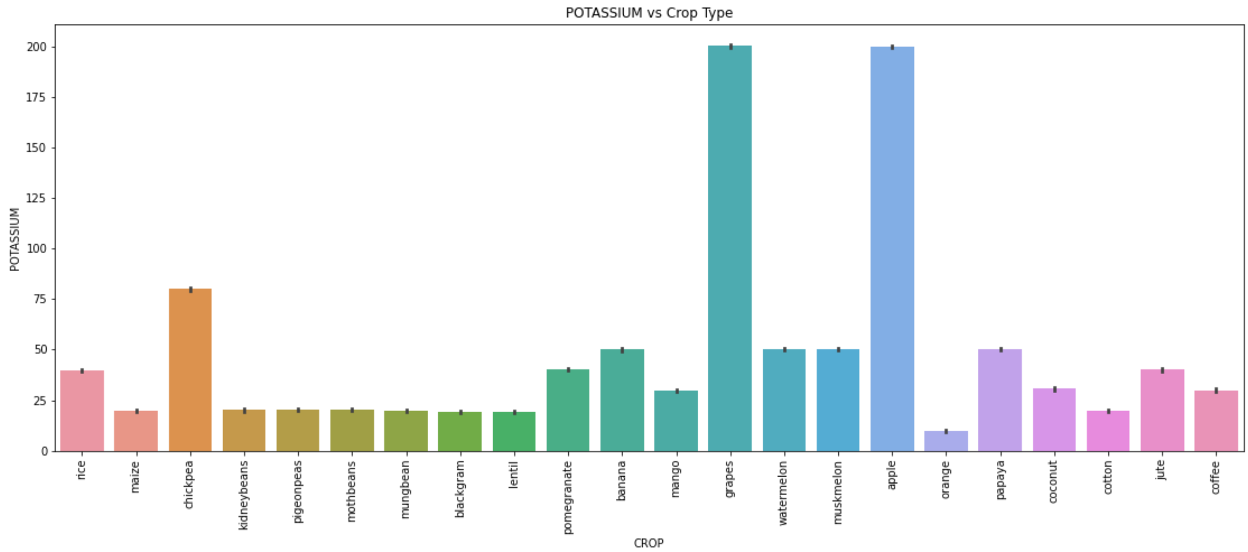


**Figure 9:** Crop Type vs Potassium

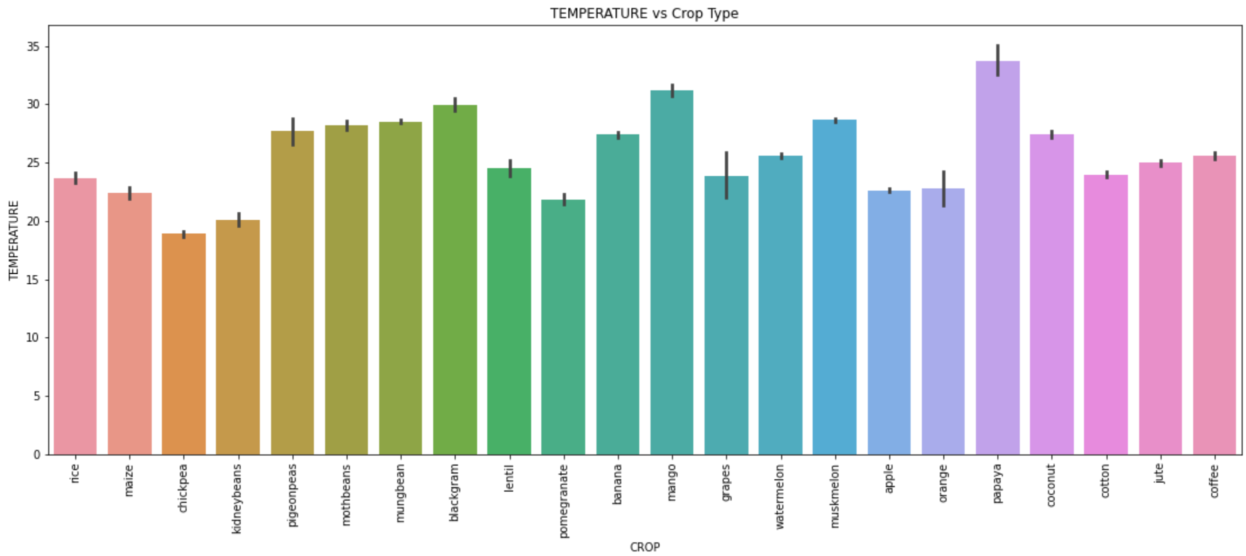


**Figure 10:** Crop Type vs Temperature

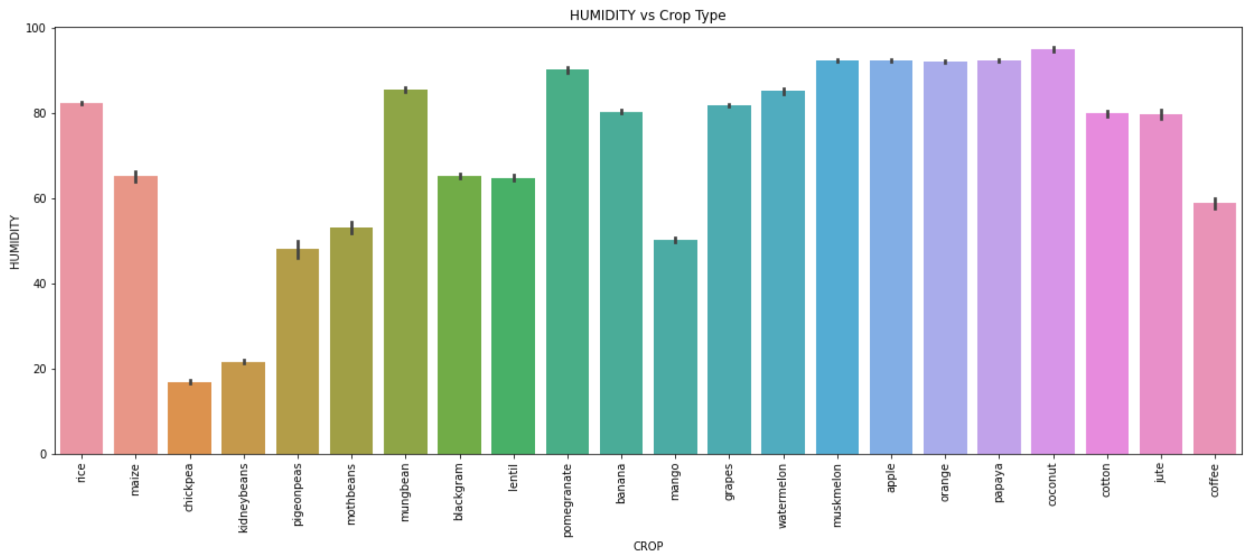


**Figure 11:** Crop Type vs Humidity

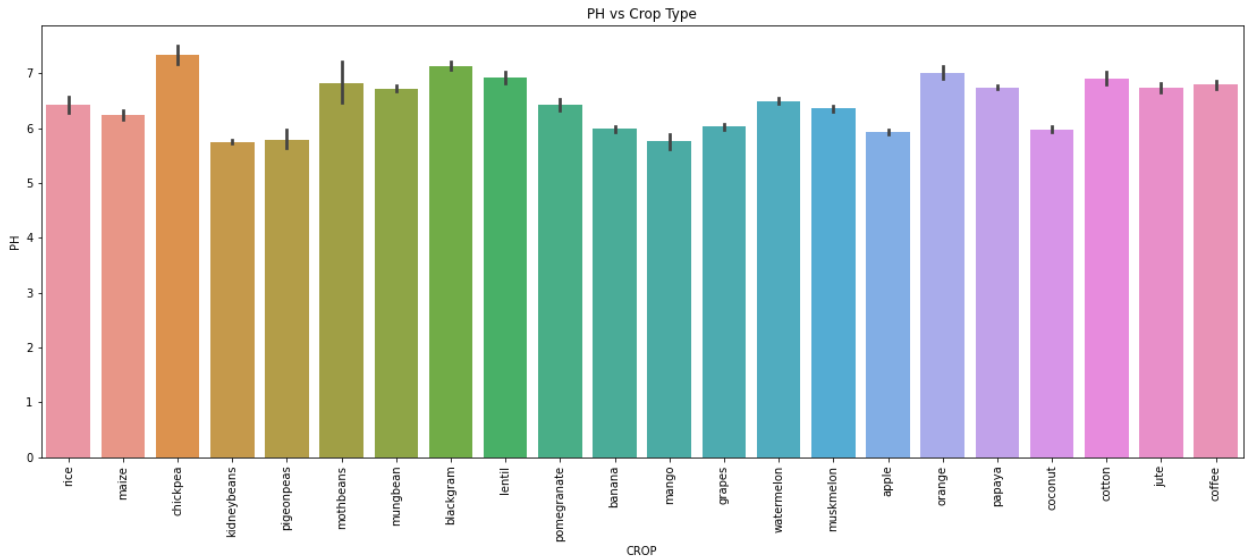


**Figure 12:** Crop Type vs PH

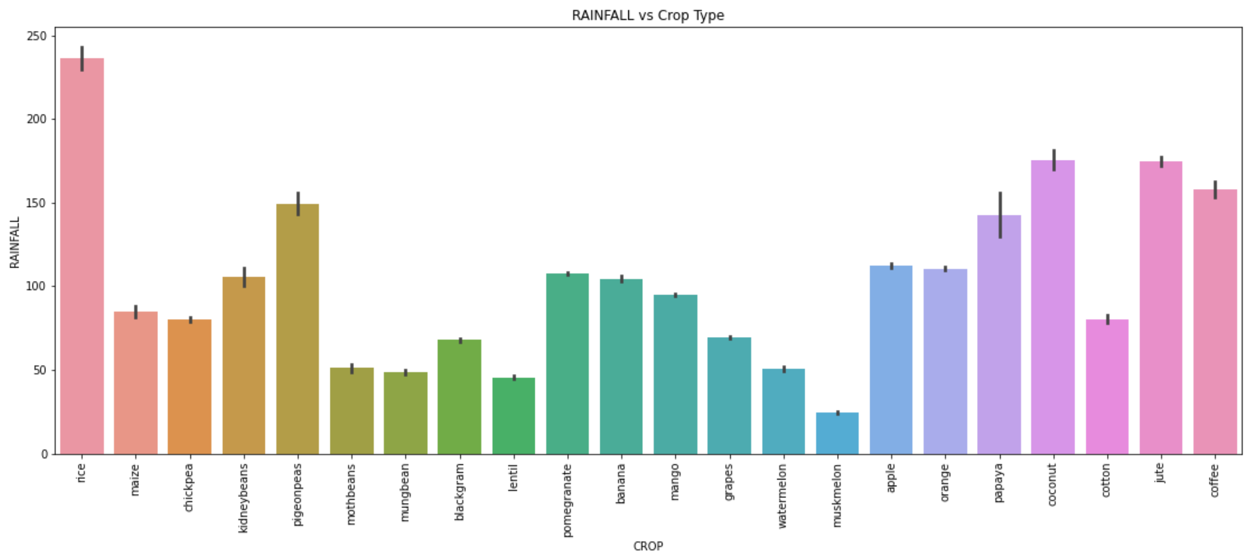


**Figure 13:** Crop Type vs Rainfall

## 3.2 DATA PRE-PROCESSING

### 3.2.1 CONSIDERATION OF FEATURES

Nitrogen, potassium, phosphorus, temperature, humidity, PH, and rainfall are the seven input features where crop is the expected output with different categories.

The output feature "CROP" consists of 22 categories those are rice, maize, chickpea, kidneybeans, pigeonpeas, mothbeans, mungbean, blackgram, lentil, pomegranate, banana, mango, grapes, watermelon, muskmelon, apple, orange, papaya, coconut, cotton, jute and coffee.

### 3.2.2 MISSING VALUES

```
NITROGEN       0
PHOSPHORUS     0
POTASSIUM      0
TEMPERATURE    0
HUMIDITY       0
PH             0
RAINFALL       0
CROP           0
dtype: int64
```

**Figure 14:** No. of missing values in each column

As there are no missing values found, missing values treatment is not required.

### 3.2.3 CONVERTING CATEGORICAL DATA TO NUMERIC

The output feature consists of categorical data with 22 different types of labels.

We can use LabelEncoder( ) function to convert these categorical data to numeric form, as there are 22 different labels the LabelEncoder( ) assigns each label with a unique number ranging between 0 to 21.

Assigned values are:

Rice = 20, maize = 11, chickpea = 3, kidneybeans = 9, pigeonpeas = 18, mothbeans = 13, mungbean =14, blackgram = 2, lentil = 10, pomegranate = 19, banana = 1, mango = 12, grapes = 7, watermelon = 21, muskmelon = 15, apple = 0, orange = 16, papaya = 17, coconut = 4, cotton = 6, jute = 8 and coffee = 5.

### 3.2.4 OVERSAMPLING THE DATA

As we have less number of data available that is about 2200 rows, we want to oversample it by 5 times, that give us a good amount of data about 11000 rows.

This can be achieved using SMOTE (Synthetic Minority Oversampling Technique) is a statistical technique for increasing the number of cases in the dataset in a balanced way. The component works by generating new instances from existing minority cases that you supply as input. It works by utilizing a K-nearest algorithm to create synthetic data.

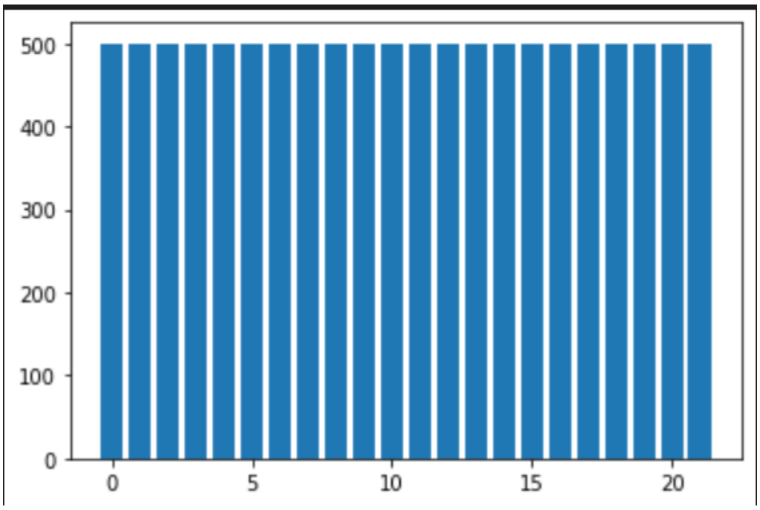


**Figure 15:** Oversampled data

## 3.3 OVERVIEW OF TECHNOLOGY

### 3.3.1 SPLITTING DATA

The data available is split into two parts, one is for training and other is for testing or validation purpose.

It can be achieved by using train_test_split( ) method with parameters like input features, output features and split size.

We have considered test_size = 0.4 i.e 40% of data is allocated for testing and remaining 60% of data is for training purpose.

### 3.3.2 IMPLEMENTATION OF DIFFERENT METHODS

This section talks about the algorithms used for the project. We used three different algorithms, they are Decision tree, Random forest, SVM.

**Decision Tree:** A Decision Tree is a supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node**.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

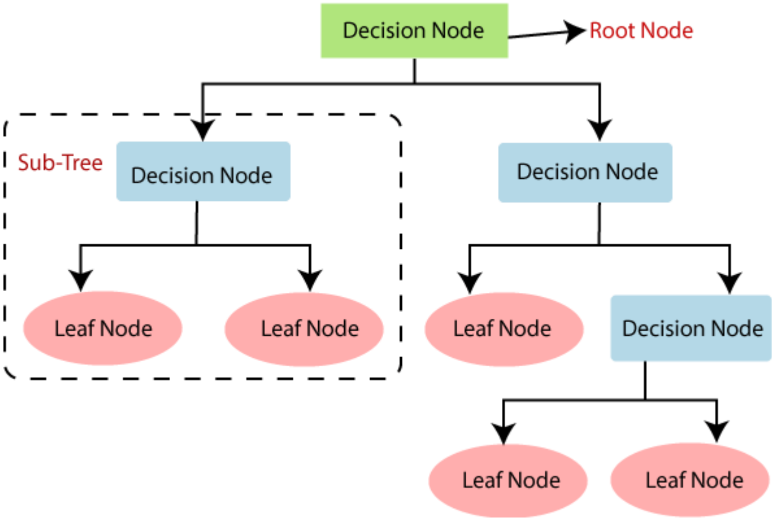


**Figure 16:** Decision tree flowchart

**Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.

**Step-2:** Find the best attribute in the dataset using the **Attribute Selection Measure (ASM).**

**Step-3:** Divide the S into subsets that contains possible values for the best attributes.

**Step-4:** Generate the decision tree node, which contain the best attribute.

**Step-5:** Recursively make new decision trees using the subsets of the dataset created in step - 3. Continue this process until a stage is reached where you cannot further classify the nodes and call the final node as a leaf node.

```python
# Create Decision Tree classifer object
dt = DecisionTreeClassifier(criterion="entropy", max_depth=9)

# Train Decision Tree Classifer
dt = dt.fit(X_train,y_train)

#Predict the response for test dataset
y_pred_test_dt = dt.predict(X_test)
y_pred_train_dt = dt.predict(X_train)

# Model Accuracy, how often is the classifier correct?
print("Testing Accuracy:",metrics.accuracy_score(y_test, y_pred_test_dt))
print("Training Accuracy:",metrics.accuracy_score(y_train, y_pred_train_dt))
```
✓ 0.8s

```
Testing Accuracy: 0.9920454545454546
Training Accuracy: 0.9971212121212121
```

**Figure 17:** Building Decision tree

**Random forest:** Random forest is used for both regression and classification-based applications. This algorithm is flexible and easy to use. Most of the time this algorithm gives accurate results even without hyper tuning the parameters. It builds many decision trees which on merging forms as a forest. While building the decision trees, adds more randomness to the model. This algorithm searches for the best feature in the random subset of features, which results in the formation of a better model.

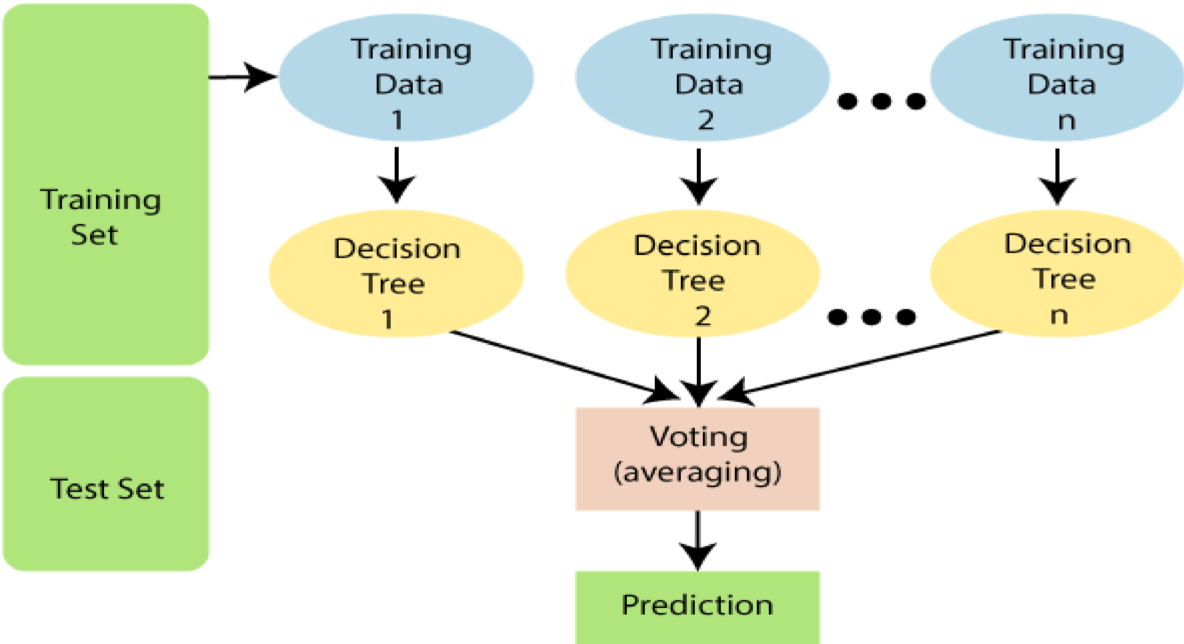


**Figure 18:** Random forest flowchart

Random Forest works in two-phase first is to create the random forest by combining the N decision tree, and the second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

```
# Create Random Forest classifer object
rf = RandomForestClassifier(criterion="entropy", max_depth=7)

# Train Random Forest Classifer
rf = rf.fit(X_train,y_train)

#Predict the response for train & test dataset
y_pred_test_rf = rf.predict(X_test)
y_pred_train_rf = rf.predict(X_train)

# Model Accuracy, how often is the classifier correct?
print("Testing Accuracy:",metrics.accuracy_score(y_test, y_pred_test_rf))
print("Training Accuracy:",metrics.accuracy_score(y_train, y_pred_train_rf))
```
✓ 1.1s

```
Testing Accuracy: 0.9972727272727273
Training Accuracy: 0.9981818181818182
```

**Figure 19:** Building Random forest

**Support Vector Machine (SVM):** The objective of the support vector machine algorithm is to find a hyperplane in N-dimensional space (N - the number of features) that distinctly classifies the data points. To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e. the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

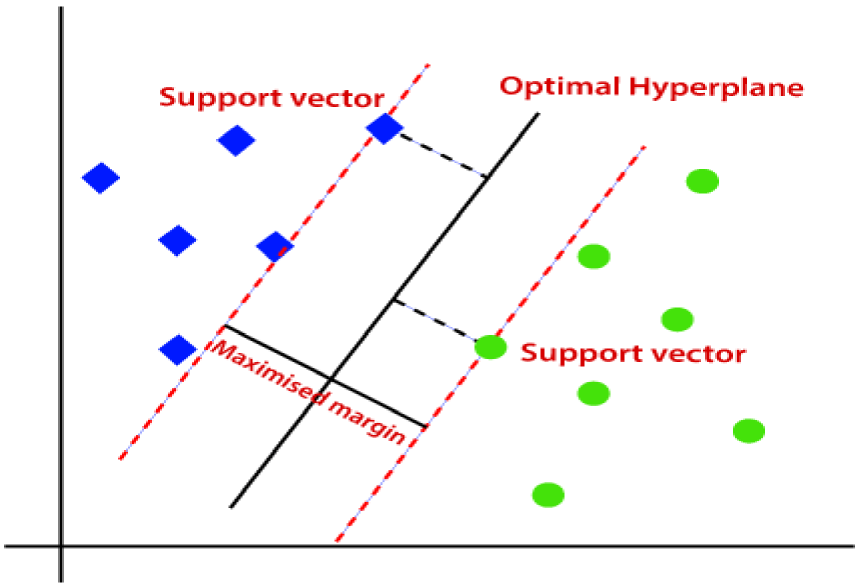


**Figure 20:** SVM

**Working of SVM:** An SVM model is basically a representation of different classes in a hyperplane in multidimensional space. The hyperplane will be generated in an iterative manner by SVM so that the error can be minimized. The goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH).

- **Support Vectors:** Data points that are closest to the hyperplane is called support vectors. Separating line will be defined with the help of these data points.
- **Hyperplane:** As we can see in the above diagram, it is a decision plane or space which is divided between a set of objects having different classes.
- **Margin:** It may be defined as the gap between two lines on the closet data points of different classes. It can be calculated as the perpendicular distance from the line to the support vectors. Large margin is considered as a good margin and small margin is considered as a bad margin.

The main goal of SVM is to divide the datasets into classes to find a maximum marginal hyperplane (MMH) and it can be done in the following two steps

- First, SVM will generate hyperplanes iteratively that segregates the classes in best way.
- Then, it will choose the hyperplane that separates the classes correctly.

```
# Create Support Vector Machine classifer object
svm = SVC()

# Train Support Vector Machine Classifer
svm = svm.fit(X_train,y_train)

#Predict the response for test dataset
y_pred_test_svm = svm.predict(X_test)
y_pred_train_svm = svm.predict(X_train)

# Model Accuracy, how often is the classifier correct?
print("Testing Accuracy:",metrics.accuracy_score(y_test, y_pred_test_svm))
print("Training Accuracy:",metrics.accuracy_score(y_train, y_pred_train_svm))
```
✓ 2.4s

```
Testing Accuracy: 0.985909090909091
Training Accuracy: 0.9889393939393939
```

**Figure 21:** Building SVM

# CHAPTER 4

# SIMULATION RESULTS & ANALYSIS

## 4.1 RESULTS

| | **Training Accuracy** | **Testing Accuracy** |
|---|---|---|
| **Decision tree classifier** | **99.71%** | **99.20%** |
| **Random forest classifier** | **99.81%** | **99.72%** |
| **SVM classifier** | **98.89%** | **98.59%** |

**Table 1:** Accuracy rates

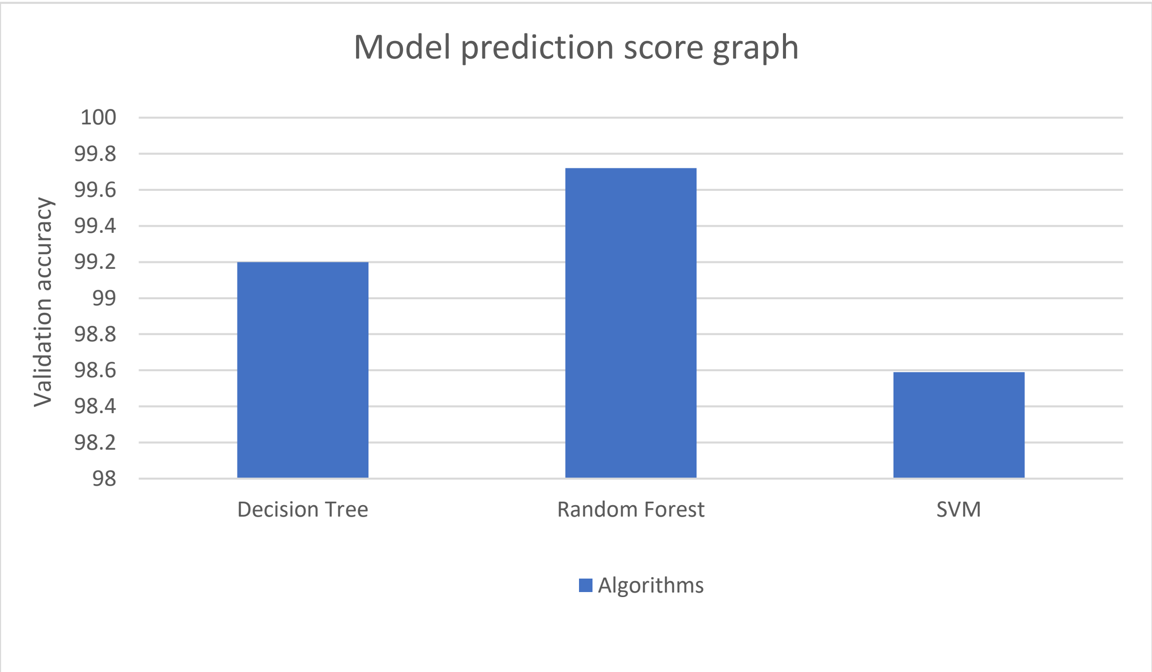


**Figure 22:** Graph about accuracies of different ML models

## 4.2 WEB APPLICATION

After completion of training and testing the machine learning models, we have chosen the model with highest accuracy that is random forest model and saved it as a pickle file, so that it can be further used to build an application.

Web application is built using flask by dumping the saved pickle file to it.

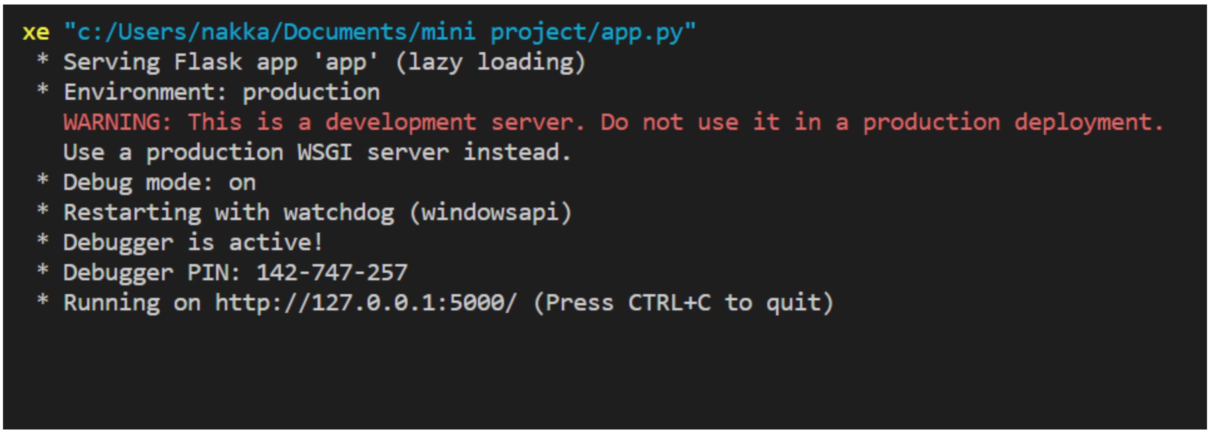


**Figure 23:** Getting the URL

After typing the acquired URL in the web browser our user interface will be appeared where you need to input the contents of soil and the result will be displayed onto the screen.

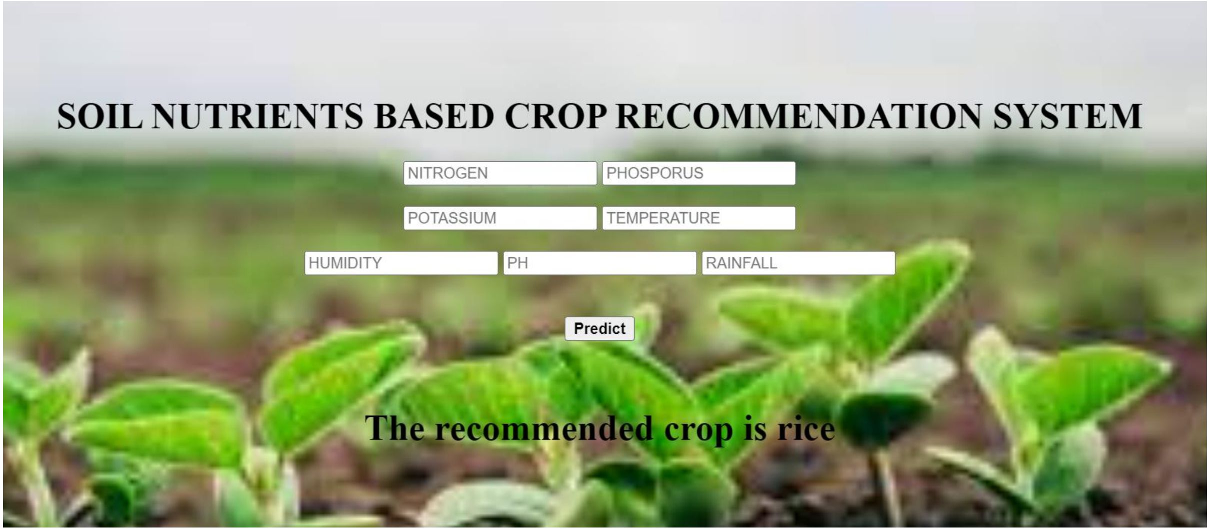


**Figure 24:** Output prediction through web application

# CHAPTER 5

# CONCLUSION AND FUTURE SCOPE

## 5.1 CONCLUSION

This study is based on the comparison of different machine learning algorithms, the dataset collected from the open source is initially divided into testing and training dataset. The ML model is provided with training dataset for generating the crop recommendation prediction model. The test data is given to the model, once the model is generated with minimum error and maximum accuracy. The inputs are fed to the generated model. Random Forest algorithm is the best classifier outperformed among all proposed algorithms, which predicts and suggests the crops to be sown with an accuracy of about 99.72%.

## 5.2 FUTURE SCOPE

The prediction of crop cultivation is critical to agriculture, with farmers keen to work out how much they can possibly expect to produce. In agriculture, several data mining strategies are used and analysed to predict crop cultivation in the future. This work has focused on comparing the different machine learning models and find the best model for prediction. The future work could be of exploring the observations by involving some more features which plays an important role in growth of plants and making more user friendly, and further we can add more features to application like crop disease prediction and giving instant solution for it, which will be an ecosystem for farmers.

# REFERENCES


[1] Raorane, A, A., Kulkarni, R, V., “Data Mining: An effective tool for yield estimation in the agricultural sector”, International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Volume.1, Issue.2, August 2012, pp: 75-79.

[2] M. Sebban and R. Nock, *A* hybrid filter/wrapper approach of feature selection using information theory, Pattern Recognit 35 (2002), pp. 835–846.

[3] P.S. Maya Gopal and R. Bhargavi., Feature selection for yield prediction in Boruta algorithm, International Journal of Pure and Applied Mathematics 118 (2018), pp. 139–144.

[4] Vidhya Sachithanandam, CROPS RECOMMENDATION BASED ON SOILS AND WEATHER PREDICTION International Conference on Innovations And Challenges In Engineering And Technology - ICICET ’18.

[5] Athani, S., Tejeshwar, C. H., Patil, M. M., Patil, P, & Kulkarni, R. (2017, February 1). Soil moisture monitoring using IoT enabled arduino sensors with neural networks for improving soil management for farmers and predict seasonal rainfall for planning future harvest in North Karnataka — India.

[6] H. Liu and H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, 1st ed., Springer, US, 1998.

[7] Kumar, R., Singh, M. P., Kumar, P., & Singh, J. P. (2015). Crop Selection Method to maximize crop yield rate using machine learning technique. IEEE Xplore.

[8] Ramesh A, Vijay. S. Rajpurohit, “A survey on Data Mining Techniques for Crop Yield Prediction”, “International Journal of Advance Research in Volume 2, Issue 9, September 2014”.

[9] Rajshekhar Borate, Rahul Ombale, Sagar Ahire, Manoj Dhawade, Mrs. Prof. R.P.Karande, “Applying Data Mining Techniques to Predict Annual Yield of Major Crops and Recommend Planting Different Crops in Different Districts in India”.