NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
: 08258 - 281039 – 281263, Fax: 08258 – 281265

**Department of Computer Science and Engineering**

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

Report on Mini Project

# CROP RECOMMENDATION

**Course Code: 18CS601**

**Course Name: MACHINE LEARNING**

Semester: VI SEM Section: C

**Submitted To:**

**Dr. Sarika Hegde**

Associate Professor
Department of Computer Science
And Engineering

**Submitted By:**

Saurav N Shetty (4nm18cs160)
Shashank (4nm18cs165)

**Date of submission: 22/05/2021**

# Problem Definition:

Precision agriculture is in trend nowadays. It helps the farmers to get informed decisions about the farming strategy. Prepare a machine learning model which would recommend the users the most suitable crops to grow in a particular farm based on various parameters.

# Dataset Explanation:

This dataset was built by augmenting datasets of rainfall, climate and fertilizer data available for India.

**Data fields**

- N - ratio of Nitrogen content in soil.
- P - ratio of Phosphorus content in soil.
- K - ratio of Potassium content in soil.
- temperature - temperature in degree Celsius.
- humidity - relative humidity in %.
- ph - ph value of the soil.
- rainfall - rainfall in mm.

The goal is to recommend the best crops based on the data available. Hence, this becomes a classification problem with multiple classes.

Here, the class labels represent the crop to be grown. In our dataset, there are 22 different class labels (crops), they are:

Rice, apple, orange, mize, coffee, banana, lentil, pomegranate, watermelon, black gram, papaya, cotton, grapes, coconut, pigeon peas, jute, muskmelon, mungbean, moth beans, chickpea, kidney beans, mango.

| 1 | N | P | K | temperatu | humidity | ph | rainfall | class |
|---|---|---|---|---|---|---|---|---|
| 2 | 90 | 42 | 43 | 20.87974 | 82.00274 | 6.502985 | 202.9355 | rice |
| 3 | 85 | 58 | 41 | 21.77046 | 80.31964 | 7.038096 | 226.6555 | rice |
| 4 | 60 | 55 | 44 | 23.00446 | 82.32076 | 7.840207 | 263.9642 | rice |
| 5 | 74 | 35 | 40 | 26.4911 | 80.15836 | 6.980401 | 242.864 | rice |
| 6 | 78 | 42 | 42 | 20.13017 | 81.60487 | 7.628473 | 262.7173 | rice |
| 7 | 69 | 37 | 42 | 23.05805 | 83.37012 | 7.073454 | 251.055 | rice |
| 8 | 69 | 55 | 38 | 22.70884 | 82.63941 | 5.700806 | 271.3249 | rice |
| 9 | 94 | 53 | 40 | 20.27774 | 82.89409 | 5.718627 | 241.9742 | rice |
| 10 | 89 | 54 | 38 | 24.51588 | 83.53522 | 6.685346 | 230.4462 | rice |
| 11 | 68 | 58 | 38 | 23.22397 | 83.03323 | 6.336254 | 221.2092 | rice |
| 12 | 91 | 53 | 40 | 26.52724 | 81.41754 | 5.386168 | 264.6149 | rice |
| 13 | 90 | 46 | 42 | 23.97898 | 81.45062 | 7.502834 | 250.0832 | rice |
| 14 | 78 | 58 | 44 | 26.8008 | 80.88685 | 5.108682 | 284.4365 | rice |
| 15 | 93 | 56 | 36 | 24.01498 | 82.05687 | 6.984354 | 185.2773 | rice |
| 16 | 94 | 50 | 37 | 25.66585 | 80.66385 | 6.94802 | 209.587 | rice |
| 17 | 60 | 48 | 39 | 24.28209 | 80.30026 | 7.042299 | 231.0863 | rice |
| 18 | 85 | 38 | 41 | 21.58712 | 82.78837 | 6.249051 | 276.6552 | rice |
| 19 | 91 | 35 | 39 | 23.79392 | 80.41818 | 6.97086 | 206.2612 | rice |
| 20 | 77 | 38 | 36 | 21.86525 | 80.1923 | 5.953933 | 224.555 | rice |
| 21 | 88 | 35 | 40 | 23.57944 | 83.5876 | 5.853932 | 291.2987 | rice |
| 22 | 89 | 45 | 36 | 21.32504 | 80.47476 | 6.442475 | 185.4975 | rice |
| 23 | 76 | 40 | 43 | 25.15746 | 83.11713 | 5.070176 | 231.3843 | rice |
| 24 | 67 | 59 | 41 | 21.94767 | 80.97384 | 6.012633 | 213.3561 | rice |
| 25 | 83 | 41 | 43 | 21.05254 | 82.6784 | 6.254028 | 233.1076 | rice |
| 26 | 98 | 47 | 37 | 23.48381 | 81.33265 | 7.375483 | 224.0581 | rice |
| 27 | 66 | 53 | 41 | 25.07564 | 80.52389 | 7.778915 | 257.0039 | rice |

## Algorithm:

---

BACKPROPAGATION(*training_examples*, $\eta$, $n_{in}$, $n_{out}$, $n_{hidden}$)

*Each training example is a pair of the form* $(\vec{x}, \vec{t})$, *where* $\vec{x}$ *is the vector of network input values, and* $\vec{t}$ *is the vector of target network output values.*

$\eta$ *is the learning rate (e.g., .05).* $n_{in}$ *is the number of network inputs,* $n_{hidden}$ *the number of units in the hidden layer, and* $n_{out}$ *the number of output units.*

*The input from unit* $i$ *into unit* $j$ *is denoted* $x_{ji}$, *and the weight from unit* $i$ *to unit* $j$ *is denoted* $w_{ji}$.

- Create a feed-forward network with $n_{in}$ inputs, $n_{hidden}$ hidden units, and $n_{out}$ output units.
- Initialize all network weights to small random numbers (e.g., between $-.05$ and .05).
- Until the termination condition is met, Do
  - For each $(\vec{x}, \vec{t})$ in *training_examples*, Do

*Propagate the input forward through the network:*

1. Input the instance $\vec{x}$ to the network and compute the output $o_u$ of every unit $u$ in the network.

*Propagate the errors backward through the network:*

2. For each network output unit $k$, calculate its error term $\delta_k$

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k) \tag{T4.3}$$

3. For each hidden unit $h$, calculate its error term $\delta_h$

$$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in outputs} w_{kh}\delta_k \tag{T4.4}$$

4. Update each network weight $w_{ji}$

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

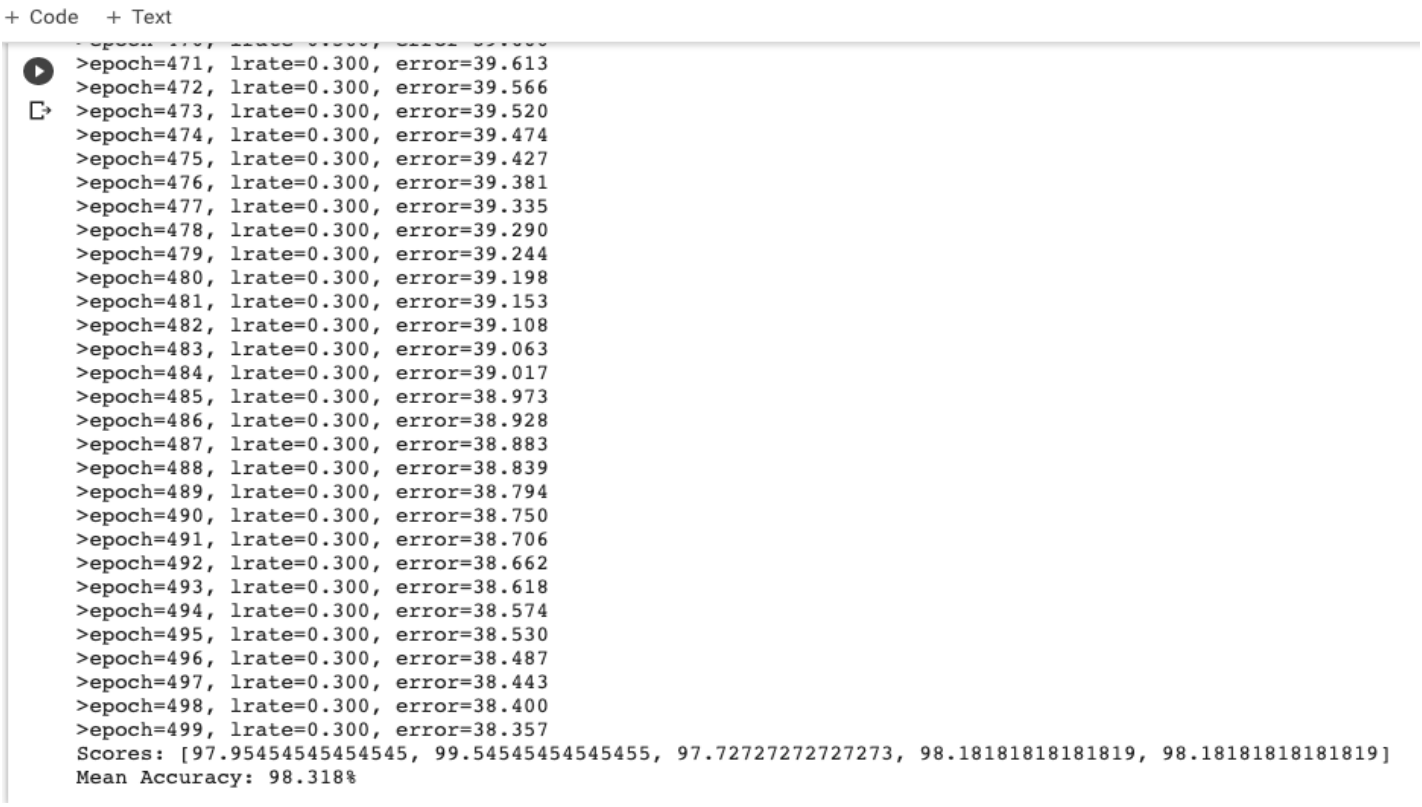
where

$$\Delta w_{ji} = \eta \, \delta_j \, x_{ji} \tag{T4.5}$$

---

- Stochastic gradient descent algorithm is used to calculate the error and update the weights.

**Outcome:**




The model of feed forward neural network used in our project has 8 input nodes, 15 nodes in the hidden layer and 22 nodes in the output layer.

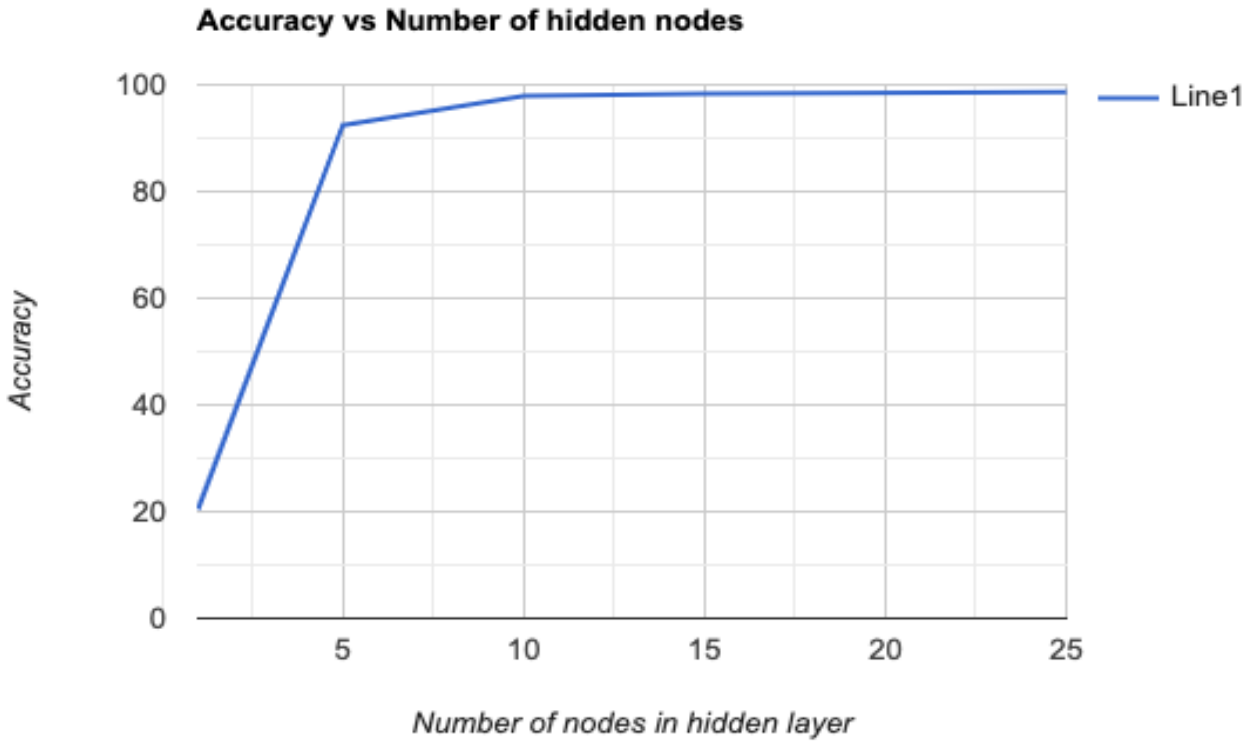
The input layer basically corresponds to a row in the dataset; each node in this layer has only one input.

The output layer consists of 22 nodes, each corresponding to one out of the 22 different class labels.

The hidden layer in our final model had 15 nodes; this was decided after experimenting with different numbers of nodes in the hidden layer.

Using these metrics we obtained a mean accuracy of 98.3%. The screenshot of the results is shown below.

The graph below shows the accuracies obtained for different numbers of hidden nodes used in the model. We can see that as the number of nodes in the hidden layer increases the accuracy also increases drastically. Accuracy reached a peak at 98% and remained constant for further increase in number of hidden nodes.



## Conclusion:

Precision agriculture is in trend nowadays. Precision agriculture is a modern farming technique that uses the data of soil characteristics, soil types, crop yield data, and weather conditions and suggests the farmers with the most optimal crop to grow in their farms for maximum yield and profit. This technique can reduce crop failures and will help the farmers to make informed decisions about their farming strategy.

In order to mitigate the agrarian crisis in the current status quo, there is a need for better recommendation systems to alleviate the crisis by helping the farmers to make an informed decision before starting the cultivation of crops.