AN EFFICIENT ARCHITECTURE FOR THE ACCURATE PREDICTION OF CROP USING ENSEMBLE LEARNING

A MINOR PROJECT REPORT

Submitted by

ANURAG GAUTAM [Reg No: RA1811003010703]

SALONI JAIN [Reg No: RA1811003010704]

Under the guidance of

Ms. S. Priya

(Assistant Professor (Sr.G), Department of Computing Technologies)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

INSTITUTE OF SCIENCE & TECHNOLOGY

Deemed to be University u/s 3 of UGC Act, 1956

S.R.M Nagar, Kattankulathur, Chengalpattu District

NOVEMBER 2021

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that 18CSP107L minor project report titled "**AN EFFICIENT ARCHITECTURE FOR THE ACCURATE PREDICTION OF CROP USING ENSEMBLE LEARNING**" is the bonafide work of "**ANURAG GAUTAM [RA1811003010703], SALONI JAIN [RA1811003010704]**" who carried out the minor project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Mrs. S. Priya

GUIDE

Assistant Professor (Sr. G)

Dept. of Computing Technologies

SIGNATURE

Dr. M. Pushpalatha

Head of Department

Dept. of Computing Technologies

ABSTRACT

Agriculture is not just a supply of resources but a way of life. It is the main source of food, fibers, and other raw material. 70% of the Indian population directly relies on agriculture. The most common problem faced by Indian farmers is productivity. This issue arises because the farmers do not choose the crops based on the quality of their own soil and other climatic factors clubbed together. This concern can be solved with the help of machine learning algorithms which is found to be an effective method for predictive analysis. This study gives solutions like proposing a recommendation system to recommend suitable crops based on soil and climatic parameters with high specific accuracy using the ensemble learning algorithm. Thus, the system will greatly help the farmers to make valuable decisions.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AI- Artificial Intelligence
GUI- Graphic User Interface
GVA- Gross Value Added
KNN- K Nearest Neighbors
ML- Machine Learning
NPK- Nitrogen, Phosphorus, Potassium
pH- Power of Hydrogen
SVM- Support Vector Machine

CHAPTER 1

INTRODUCTION

A farmer's call regarding which crop to grow is usually clouded by his intuition and alternative factors like creating instant profits, lack of awareness regarding market demand, overestimating a soil's potential to support a specific crop, and so on. A misguided call on the part of the farmer may place a major strain on his family's financial status. Maybe this might be one in every one of the numerous reasons conducive to the innumerous suicide cases of farmers that we tend to hear from the media.

With existing forecasting and monitoring methods, we can address this problem. While these methods are helpful, there is no optimal solution to suggest a harvest. Some of the disadvantages found with the existing system are inadequate analysis, the selection of effective algorithms and the efficient selection of attributes, all these parameters can affect the crop yield. The proposed system helps to overcome the disadvantages of the existing system. With this in mind, we tend to propose a system that may contemplate environmental parameters (temperature, rainfall, humidity) and soil characteristics (pH value, Nitrogen value, Phosphorus value, Potassium value) before recommending the prime appropriate crop to the user.

This research work proposes an ensemble learning-based crop recommendation system using methods such as bagging and boosting. The performance of the proposed model is effective in recommending cultivation based on various input parameters. The different steps of this research work are data collection, pre-processing, building a recommendation model, training, and testing the recommendation model.

The first step in pre-processing is the selection of features and parameters. Since the data set is an unbalanced data set, the second pre-processing step is to balance the data set. The proposed recommendation model has been developed, trained and tested to recommend an appropriate crop to the farmer. This research aims to propose an ensemble learning-based recommendation model that recommends the most appropriate culture for the given inputs. The performance of the model is validated against various parameters, and the ensemble learning-based recommendation model is effective in recommending items to the user.

1.1 PROBLEM STATEMENT

In a country like India, where agriculture and also the connected sectors contribute so much to its Gross Value Added (GVA), an incorrect judgment about which crop to grow to minimize losses and maximize benefits would have negative implications on not simply the farmer’s family, however the whole economy of a district. For this reason, we've known a farmer’s perplexity regarding which crop to grow as per the soil and environmental condition parameters, as a great one. The necessity of the hour is to style a system that would offer prognostic insights to the Indian farmers, thereby serving them to create an informed call regarding which crop to grow.

1.2 OBJECTIVES

- Our objective is to analyse soil contents and weather conditions available in the field to further recommend a crop using that analysis.
- It will minimize losses in agriculture due to unfavourable conditions.
- It will give the maximum benefit to the farmers.
- It will also assure that the fertilizers are used appropriately as per the requirement.

CHAPTER 2

LITERATURE SURVEY

[1] Viviliya, B. and Vaidhehi, V., "The Design of Hybrid Crop Recommendation System using Machine Learning Algorithms"

This research article proposes a hybrid crop recommendation system using classifiers such as Naive Bayes, J48 and association rules. The recommendation system is designed based on the certain attributes such as geographic and climatic parameters with the region, type of soil, content of soil nutrients, temperature and water table of the crop.

[2] Dhruv Piyush Parikh, Jugal Jain, Tanishq Gupta and Rishit Hemant Dabhade, "Machine Learning Based Crop Recommendation System"

This research paper uses scikit-learn to implement machine learning algorithm and GUI implementation to predict the best suitable crop based on different input parameters. Using Tkinter gives a graphical user interface for the user to interact with the model more easily.

[3] Kevin Tom Thomas, Varsha S, Merin Mary Saji, Lisha Varghese, Er. Jinu Thomas, "Crop Prediction Using Machine Learning"

The paper takes N, P, K and pH values into consideration to determine the best crop for the given soil conditions. It uses various algorithms like K Nearest Neighbors, Decision Tree, Naïve Bayes and Support Vector Machines to recommend the most suitable crop for the input soil parameters.

[4] S.Veenadhari, Dr. Bharat Misra, Dr. CD Singh, “Machine Learning approach for Forecasting Crop Yield based on Climatic Parameters”

In the present study, a software tool called "Crop Advisor" is developed to predict the influence of climatic parameters on crop yield. Algorithm C .5 is used to determine the climatic parameters with the greatest impact on the crop yield of selected crops. This software provides an indication of the relative influence of different climatic parameters on crop yield, as the application of these input parameters varies with the individual fields in space and time.

[5] Dr.A.K.Mariappan, Ms C. Madhumitha, Ms P. Nishitha, Ms S. Nivedhitha, “Crop Recommendation System through Soil Analysis Using Classification in Machine Learning”

The Crop Recommendation system uses KNN classification in Supervised Machine Learning Algorithm to recommend suitable crops with higher accuracy and efficiency. The system lists the suitable crops based on the soil and leaves it upon the farmers to decide on the crop to be sown.

CHAPTER 3

SYSTEM ARCHITECHTURE AND DESIGN

The proposed system is depicted using an architecture diagram as shown in figure 3, where we have a database with pre-processed data which will be split into training and testing data. The training data will be used to train the ensemble model which will result in a recommendation model which will be further tested for accuracy and also undergo cross-validation tests. The final model will use the testing data to make concluding predictions.

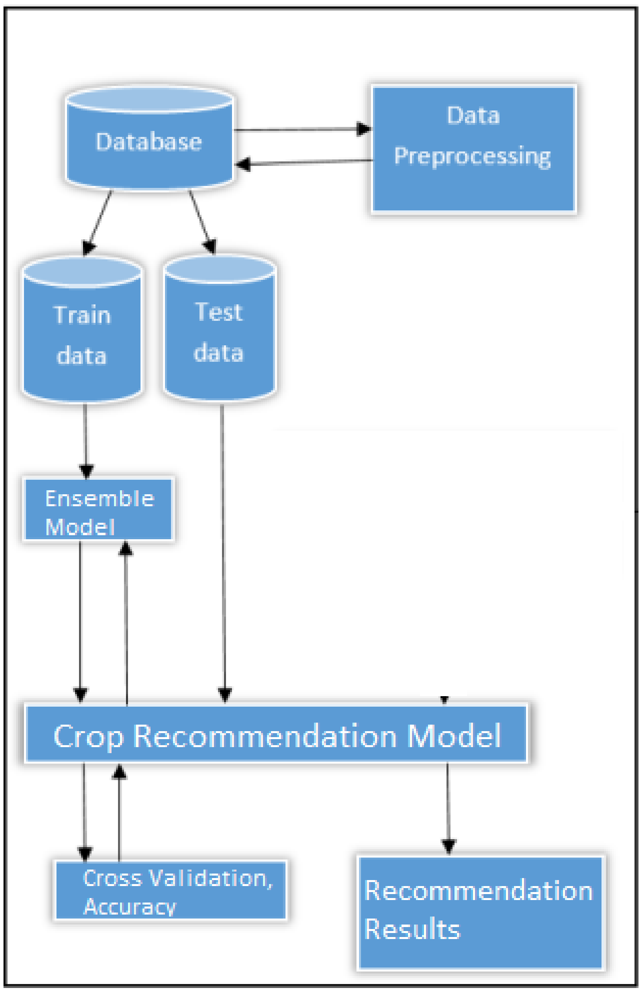


Figure 3: Architecture Diagram

CHAPTER 4

METHODOLOGY

4.1 SOFTWARE USED

4.1.1 Google Colaboratory

Colaboratory, abbreviated as "Colab", is a product of Google Research. Colab allows anyone to write and execute arbitrary Python code through the browser, and is particularly well suited for machine learning, data analysis, and education. Technically, Colab is a hosted Jupyter notebook service that requires no configuration to be used and offers free access to computing resources, including GPUs.

4.2 TECHNOLOGIES USED

4.2.1 Python 3.6.9

Python is a modern-day programming language that is used in various fields due to its vast set of libraries and functionalities. Its usage starts from basic programming to complex software development. It is a user-friendly language that makes it widely used for machine learning and data analysis applications.

4.2.2 Machine Learning

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to be more precise in predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical input data to predict new output values.

4.2.3 Ensemble Learning

Ensemble learning is a popular machine learning technique that combines multiple models to improve the overall accuracy of machine learning algorithms.[6] Ensemble modelling is a great way to enhance the model presentation. It is a repetitive cycle in which different models are intentionally created and combined to deal with a particular computing problem.

4.2.4 Bagging

Bagging stands for "Bootstrap Aggregation" and is used to reduce variance in the forecast model. Grouping generates additional training data from the dataset. This is achieved by random sampling with the replacement of the original data set. Substitute sampling can repeat some observations on each new training dataset. All Bagging items have an equal chance of appearing in a new dataset. These multiple datasets are used to train multiple models in parallel. All forecasts from different ensemble models are averaged. The majority vote obtained by the voting mechanism is taken into account in the ranking.

Bagging reduces variance and adjusts the prediction to an expected outcome. The Random Forest model uses bagging, where there are decision tree models with higher variance. Random Forest makes a random selection of features to grow trees. Several random trees form a random forest.

4.2.5 Boosting

Boosting is a sequential ensemble method that iteratively adjusts the observation weight according to the latest classification. If an observation is misclassified, weight is added to that observation. The term "boosting" in common parlance refers to algorithms that turn a weak learner into a stronger one. Boosting reduces bias error and builds robust predictive models.

Data points that were erroneously predicted in each iteration are recognized and their weight is increased. The boosting algorithm assigns weights to each resulting model during training. A learner with good predictive results from the training data is assigned a higher weight. When evaluating a new learner, boosting tracks learner mistakes. XGBoost is a boosting algorithm, which stands for eXtreme Gradient Boosting. It is an extension to gradient boosted decision trees and are designed to improve speed and performance.

CHAPTER 5

CODING AND TESTING

Data Pre-processing

```
[ ] import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns
```

```
[ ] merge_fert = pd.read_csv('FertilizerData.csv')
    merge_fert.head()
```

	Unnamed: 0	Crop	N	P	K	pH
0	0	rice	80	40	40	5.5
1	3	maize	80	40	20	5.5
2	5	chickpea	40	60	80	5.5
3	12	kidneybeans	20	60	20	5.5
4	13	pigeonpeas	20	60	20	5.5

```
[ ] merge_fert.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Crop    22 non-null     object
 1   N       22 non-null     int64
 2   P       22 non-null     int64
 3   K       22 non-null     int64
 4   pH      22 non-null     float64
dtypes: float64(1), int64(3), object(1)
memory usage: 1008.0+ bytes
```

```
[ ] del merge_fert['Unnamed: 0']

    merge_fert.describe()
```

	N	P	K	pH
count	22.000000	22.000000	22.000000	22.000000
mean	50.454545	45.681818	48.181818	5.409091
std	36.315715	32.634172	51.698426	0.590326
min	20.000000	10.000000	10.000000	4.000000
25%	20.000000	20.000000	20.000000	5.500000
50%	30.000000	40.000000	30.000000	5.500000
75%	80.000000	60.000000	50.000000	5.500000
max	120.000000	125.000000	200.000000	6.500000

```
[ ] merge_fert['Crop'].unique()
```

```
array(['rice', 'maize', 'chickpea', 'kidneybeans', 'pigeonpeas',
       'mothbeans', 'mungbean', 'blackgram', 'lentil', 'pomegranate',
       'banana', 'mango', 'grapes', 'watermelon', 'muskmelon', 'apple',
       'orange', 'papaya', 'coconut', 'cotton', 'jute', 'coffee'],
      dtype=object)
```

```
[ ] merge_crop = pd.read_csv('MergeFileCrop.csv')
    reco_fert = merge_fert
```

```
merge_crop.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   N            2200 non-null   object 
 1   P            2200 non-null   object 
 2   K            2200 non-null   object 
 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   label        2200 non-null   object 
dtypes: float64(4), object(4)
memory usage: 137.6+ KB
```

```
merge_crop.shape[0]
```

```
2200
```

```
[ ] import random
    temp = pd.DataFrame(columns = ['N','P','K'])
    for i in range(0,merge_crop.shape[0]):
        crop = merge_crop.label.iloc[i]
        N = reco_fert[reco_fert['Crop'] == crop]["N"].iloc[0] + random.randint(-20,20)
        P = reco_fert[reco_fert['Crop'] == crop]["P"].iloc[0] + random.randint(-5,20)
        K = reco_fert[reco_fert['Crop'] == crop]["K"].iloc[0] + random.randint(-5,5)
        d = {"N":N,"P":P,"K":K}
        temp = temp.append(d,ignore_index = True)
    temp
```

	N	P	K
0	93	40	39
1	91	38	35
2	87	56	45
3	92	59	43
4	99	35	40
...	...	...	...
2195	96	19	27
2196	119	19	34
2197	81	30	30
2198	97	33	25
2199	114	24	26

2200 rows × 3 columns

```
[ ] merge_crop['N'] = temp['N']
    merge_crop['P'] = temp['P']
    merge_crop['K'] = temp['K']

    merge_crop
```

	Unnamed: 0	temperature	humidity	ph	rainfall	label	N	P	K
0	0	20.879744	82.002744	6.502985	202.935536	rice	95	58	35
1	1	21.770462	80.319644	7.038096	226.655537	rice	69	43	37
2	2	23.004459	82.320763	7.840207	263.964248	rice	95	39	38
3	3	26.491096	80.158363	6.980401	242.864034	rice	62	43	35
4	4	20.130175	81.604873	7.628473	262.717340	rice	76	60	45
...	...	...	...	...	...	...	...	...	...
2195	895	26.774637	66.413269	6.780064	177.774507	coffee	97	32	25
2196	896	27.417112	56.636362	6.086922	127.924610	coffee	104	28	29
2197	897	24.131797	67.225123	6.362608	173.322839	coffee	93	30	26
2198	898	26.272418	52.127394	6.758793	127.175293	coffee	110	33	25
2199	899	23.603016	60.396475	6.779833	140.937041	coffee	109	36	31

2200 rows × 9 columns

```
[ ] del merge_crop['Unnamed: 0']

    merge_crop
```

	temperature	humidity	ph	rainfall	label	N	P	K
0	20.879744	82.002744	6.502985	202.935536	rice	95	58	35
1	21.770462	80.319644	7.038096	226.655537	rice	69	43	37
2	23.004459	82.320763	7.840207	263.964248	rice	95	39	38
3	26.491096	80.158363	6.980401	242.864034	rice	62	43	35
4	20.130175	81.604873	7.628473	262.717340	rice	76	60	45
...	...	...	...	...	...	...	...	...
2195	26.774637	66.413269	6.780064	177.774507	coffee	97	32	25
2196	27.417112	56.636362	6.086922	127.924610	coffee	104	28	29
2197	24.131797	67.225123	6.362608	173.322839	coffee	93	30	26
2198	26.272418	52.127394	6.758793	127.175293	coffee	110	33	25
2199	23.603016	60.396475	6.779833	140.937041	coffee	109	36	31

2200 rows × 8 columns

```
[ ] merge_crop = merge_crop[[ 'N', 'P', 'K','temperature', 'humidity', 'ph', 'rainfall', 'label']]

    merge_crop.to_csv("crop_recommendation.csv",index=False)

    # Checking if everything went fine
    df = pd.read_csv('crop_recommendation.csv')

    df.head()
```

	N	P	K	temperature	humidity	ph	rainfall	label
0	95	58	35	20.879744	82.002744	6.502985	202.935536	rice
1	69	43	37	21.770462	80.319644	7.038096	226.655537	rice
2	95	39	38	23.004459	82.320763	7.840207	263.964248	rice
3	62	43	35	26.491096	80.158363	6.980401	242.864034	rice
4	76	60	45	20.130175	81.604873	7.628473	262.717340	rice

```
[ ] df.shape

    (2200, 8)
```

Model Building

```
[ ] # Importing libraries
    from sklearn.model_selection import cross_val_score
    from __future__ import print_function
    import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.metrics import classification_report
    from sklearn import metrics
    from sklearn import tree
    import warnings
    warnings.filterwarnings('ignore')
```

```
[ ] df = pd.read_csv('crop_recommendation.csv')

    df.head()
```

	N	P	K	temperature	humidity	ph	rainfall	label
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

```
[ ] df.tail()
```

	N	P	K	temperature	humidity	ph	rainfall	label
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

```
[ ] df.size
```

```
17600
```

```
[ ] df.shape
```

```
(2200, 8)
```

```
[ ] df.columns
```

```
Index(['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'rainfall', 'label'], dtype='object')
```

```
[ ] df['label'].value_counts()
```

```
watermelon     100
blackgram      100
mothbeans      100
muskmelon      100
jute           100
coconut        100
mango          100
grapes         100
lentil         100
mungbean       100
coffee         100
cotton         100
pigeonpeas     100
kidneybeans    100
rice           100
banana         100
apple          100
maize          100
papaya         100
pomegranate    100
chickpea       100
orange         100
Name: label, dtype: int64
```

Seperating features and target label

```
[ ] features = df[['N', 'P','K','temperature', 'humidity', 'ph', 'rainfall']]
    target = df['label']
    labels = df['label']
```

```
[ ] # Initialzing empty lists to append all model's name and corresponding name
    acc = []
    model = []
```

```
[ ] # Splitting into train and test data

    from sklearn.model_selection import train_test_split
    Xtrain, Xtest, Ytrain, Ytest = train_test_split(features,target,test_size = 0.2,random_state =2)
```

BAGGING: Random Forest

```
[ ]  from sklearn.ensemble import RandomForestClassifier

     RF = RandomForestClassifier(n_estimators=20, random_state=0)
     RF.fit(Xtrain,Ytrain)

     predicted_values = RF.predict(Xtest)

     x = metrics.accuracy_score(Ytest, predicted_values)
     acc.append(x)
     model.append('RF')
     print("RF's Accuracy is: ", x)

     print(classification_report(Ytest,predicted_values))
```

```
RF's Accuracy is:  0.990909090909091
              precision    recall  f1-score   support

       apple       1.00      1.00      1.00        13
      banana       1.00      1.00      1.00        17
   blackgram       0.94      1.00      0.97        16
    chickpea       1.00      1.00      1.00        21
     coconut       1.00      1.00      1.00        21
      coffee       1.00      1.00      1.00        22
      cotton       1.00      1.00      1.00        20
      grapes       1.00      1.00      1.00        18
        jute       0.90      1.00      0.95        28
 kidneybeans       1.00      1.00      1.00        14
      lentil       1.00      1.00      1.00        23
       maize       1.00      1.00      1.00        21
       mango       1.00      1.00      1.00        26
   mothbeans       1.00      0.95      0.97        19
    mungbean       1.00      1.00      1.00        24
   muskmelon       1.00      1.00      1.00        23
      orange       1.00      1.00      1.00        29
      papaya       1.00      1.00      1.00        19
  pigeonpeas       1.00      1.00      1.00        18
 pomegranate       1.00      1.00      1.00        17
        rice       1.00      0.81      0.90        16
  watermelon       1.00      1.00      1.00        15

    accuracy                           0.99       440
   macro avg       0.99      0.99      0.99       440
weighted avg       0.99      0.99      0.99       440
```

```
[ ] # Cross validation score (Random Forest)
    score = cross_val_score(RF,features,target,cv=5)
    score
```

```
array([0.99772727, 0.99545455, 0.99772727, 0.99318182, 0.98863636])
```

```
[ ] import pickle
    # Dump the trained Naive Bayes classifier with Pickle
    RF_pkl_filename = 'RandomForest.pkl'
    # Open the file to save as pkl file
    RF_Model_pkl = open(RF_pkl_filename, 'wb')
    pickle.dump(RF, RF_Model_pkl)
    # Close the pickle instances
    RF_Model_pkl.close()
```

BOOSTING: XGBOOST

```
[ ] import xgboost as xgb
    XB = xgb.XGBClassifier()
    XB.fit(Xtrain,Ytrain)

    predicted_values = XB.predict(Xtest)

    x = metrics.accuracy_score(Ytest, predicted_values)
    acc.append(x)
    model.append('XGBoost')
    print("XGBoost's Accuracy is: ", x)

    print(classification_report(Ytest,predicted_values))
```

```
XGBoost's Accuracy is:  0.9931818181818182
              precision    recall  f1-score   support

       apple       1.00      1.00      1.00        13
      banana       1.00      1.00      1.00        17
   blackgram       1.00      1.00      1.00        16
    chickpea       1.00      1.00      1.00        21
     coconut       1.00      1.00      1.00        21
      coffee       1.00      1.00      1.00        22
      cotton       1.00      1.00      1.00        20
      grapes       1.00      1.00      1.00        18
        jute       0.96      0.93      0.95        28
 kidneybeans       1.00      1.00      1.00        14
      lentil       1.00      1.00      1.00        23
       maize       1.00      1.00      1.00        21
       mango       1.00      1.00      1.00        26
   mothbeans       1.00      1.00      1.00        19
    mungbean       1.00      1.00      1.00        24
   muskmelon       1.00      1.00      1.00        23
      orange       1.00      1.00      1.00        29
      papaya       1.00      1.00      1.00        19
  pigeonpeas       1.00      1.00      1.00        18
 pomegranate       1.00      1.00      1.00        17
        rice       0.88      0.94      0.91        16
  watermelon       1.00      1.00      1.00        15

    accuracy                           0.99       440
   macro avg       0.99      0.99      0.99       440
weighted avg       0.99      0.99      0.99       440
```

```
# Cross validation score (XGBoost)
score = cross_val_score(XB,features,target,cv=5)
score
```

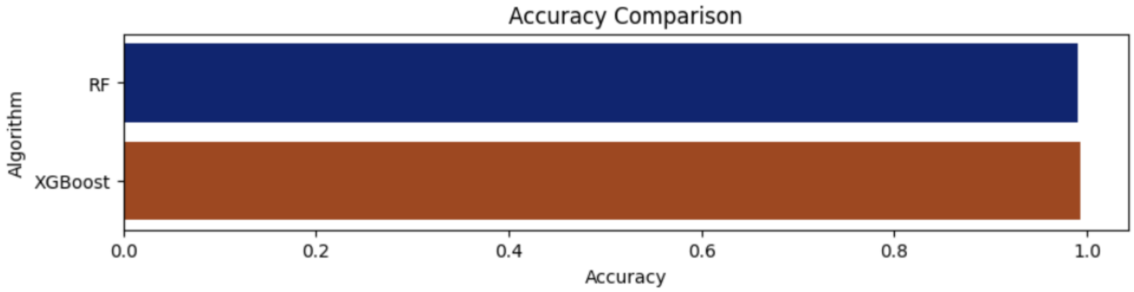
```
array([0.98636364, 0.99318182, 0.99545455, 0.99090909, 0.98409091])
```

```
import pickle
# Dump the trained Naive Bayes classifier with Pickle
XB_pkl_filename = 'XGBoost.pkl'
# Open the file to save as pkl file
XB_Model_pkl = open(XB_pkl_filename, 'wb')
pickle.dump(XB, XB_Model_pkl)
# Close the pickle instances
XB_Model_pkl.close()
```

Accuracy Comparison

```
[ ] plt.figure(figsize=[10,2],dpi = 100)
    plt.title('Accuracy Comparison')
    plt.xlabel('Accuracy')
    plt.ylabel('Algorithm')
    sns.barplot(x = acc,y = model,palette='dark')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2c8fd63b10>
```




```
[ ] accuracy_models = dict(zip(model, acc))
    for k, v in accuracy_models.items():
        print (k, '-->', v)
```

```
RF --> 0.990909090909091
XGBoost --> 0.9931818181818182
```

Prediction Making

```
[ ] data = np.array([[104,18, 30, 23.603016, 60.3, 6.7, 140.91]])
    prediction = RF.predict(data)
    print(prediction)
```

```
['coffee']
```

```
[ ] data = np.array([[83, 45, 60, 28, 70.3, 7.0, 150.9]])
    prediction = RF.predict(data)
    print(prediction)
```

```
['jute']
```

CHAPTER 6

RESULTS AND DISCUSSIONS

Figure 6.1 shows the classification reports of both bagging and boosting methods. Figure 6.2 gives a comparison between the accuracy of the two algorithms. As evident, the boosting method using XGBoost algorithm yields better accuracy as compared to all other methods.

RF's Accuracy is: 0.990909090909091

	precision	recall	f1-score	support
apple	1.00	1.00	1.00	13
banana	1.00	1.00	1.00	17
blackgram	0.94	1.00	0.97	16
chickpea	1.00	1.00	1.00	21
coconut	1.00	1.00	1.00	21
coffee	1.00	1.00	1.00	22
cotton	1.00	1.00	1.00	20
grapes	1.00	1.00	1.00	18
jute	0.90	1.00	0.95	28
kidneybeans	1.00	1.00	1.00	14
lentil	1.00	1.00	1.00	23
maize	1.00	1.00	1.00	21
mango	1.00	1.00	1.00	26
mothbeans	1.00	0.95	0.97	19
mungbean	1.00	1.00	1.00	24
muskmelon	1.00	1.00	1.00	23
orange	1.00	1.00	1.00	29
papaya	1.00	1.00	1.00	19
pigeonpeas	1.00	1.00	1.00	18
pomegranate	1.00	1.00	1.00	17
rice	1.00	0.81	0.90	16
watermelon	1.00	1.00	1.00	15
accuracy			0.99	440
macro avg	0.99	0.99	0.99	440
weighted avg	0.99	0.99	0.99	440

XGBoost's Accuracy is: 0.9931818181818182

	precision	recall	f1-score	support
apple	1.00	1.00	1.00	13
banana	1.00	1.00	1.00	17
blackgram	1.00	1.00	1.00	16
chickpea	1.00	1.00	1.00	21
coconut	1.00	1.00	1.00	21
coffee	1.00	1.00	1.00	22
cotton	1.00	1.00	1.00	20
grapes	1.00	1.00	1.00	18
jute	0.96	0.93	0.95	28
kidneybeans	1.00	1.00	1.00	14
lentil	1.00	1.00	1.00	23
maize	1.00	1.00	1.00	21
mango	1.00	1.00	1.00	26
mothbeans	1.00	1.00	1.00	19
mungbean	1.00	1.00	1.00	24
muskmelon	1.00	1.00	1.00	23
orange	1.00	1.00	1.00	29
papaya	1.00	1.00	1.00	19
pigeonpeas	1.00	1.00	1.00	18
pomegranate	1.00	1.00	1.00	17
rice	0.88	0.94	0.91	16
watermelon	1.00	1.00	1.00	15
accuracy			0.99	440
macro avg	0.99	0.99	0.99	440
weighted avg	0.99	0.99	0.99	440

Fig. 6.1: Classification reports of Random Forest (Bagging) and XGBoost (Boosting) models

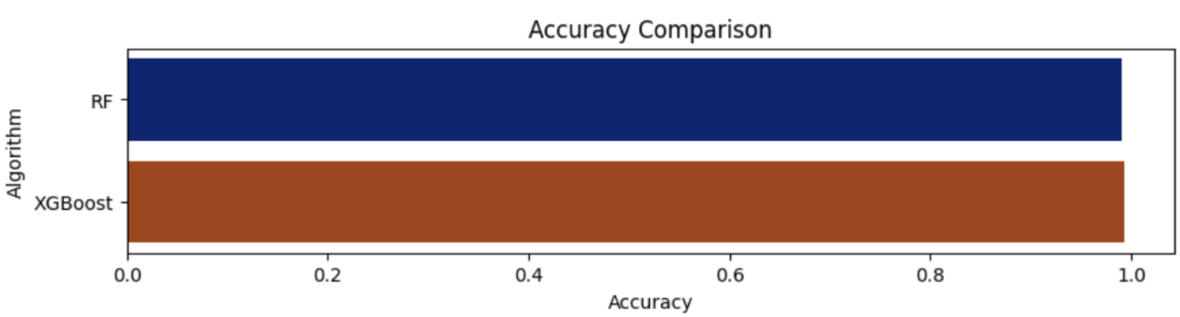


Fig. 6.2: Accuracy comparison of Random Forest (Bagging) and XGBoost (Boosting) models

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

The dataset collected from the open source is initially divided into train and test dataset. The ensemble model is provided with training dataset for generating the crop suggestion 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. The model then predicts and suggests the crops to be sown with an accuracy of about 99.31%.

In future, this project can be enhanced to add various other features. Some possible future enhancements include:

- Recommendation of fertilizers and micronutrients
- Crop prevention from diseases
- Implementation in different regional languages

REFERENCES

[1] Viviliya, B. and Vaidhehi, V., “The Design of Hybrid Crop Recommendation System using Machine Learning Algorithms”

[2] Dhruv Piyush Parikh, Jugal Jain, Tanishq Gupta and Rishit Hemant Dabhade, “Machine Learning Based Crop Recommendation System”

[3] Kevin Tom Thomas, Varsha S, Merin Mary Saji, Lisha Varghese, Er. Jinu Thomas, “Crop Prediction Using Machine Learning”

[4] S.Veenadhari, Dr. Bharat Misra, Dr. CD Singh, “Machine Learning approach for Forecasting Crop Yield based on Climatic Parameters”

[5] Dr.A.K.Mariappan, Ms C. Madhumitha, Ms P. Nishitha, Ms S. Nivedhitha, “Crop Recommendation System through Soil Analysis Using Classification in Machine Learning”

[6] https://bdtechtalks.com/2020/11/12/what-is-ensemble-learning/

[7] https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397

[8] Anguraj.K, Thiyaneswaran.B, Megashree.G, Preetha Shri.J.G, Navya.S, Jayanthi. J, “Crop Recommendation on Analyzing Soil Using Machine Learning”

[9] Ishita Dasgupta, Jayit Saha, Pattabiraman Venkatasubbu, Parvathi Ramasubramanian, “AI Crop Predictor and Weed Detector Using Wireless Technologies: A Smart Application for Farmers”

[10] Madhusree Kuanr, Bikram Kesari Rath, Sachi Nandan Mohanty, “Crop Recommender System for the Farmers using Mamdani Fuzzy Inference Model”

[11] K. Shinde, J. Andrei, and A. Oke., “Web Based Recommendation System for Farmers”

[12] G.Najeeb Ahmed, Dr.S.Kamalakkannan, “Survey for predicting agriculture cultivation using IoT”

[13] Sagar S. Badhiye, Nilesh U. Sambhe, P. N. Chatur, “KNN Technique for Analysis and Prediction of Temperature and Humidity Data”

[14] Priyadharshini A, Swapneel Chakraborty, Aayush Kumar, Omen Rajendra Pooniwala, “Intelligent Crop Recommendation System using Machine Learning”

[15] Rohit Kumar Rajak, Ankit Pawar, Mitalee Pendke, Pooja Shinde, Suresh Rathod, Avinash Devare, “Crop Recommendation System to Maximize Crop Yield using Machine Learning Technique”

APPENDIX: PLAGIARISM REPORT