Industrial Internship Report on

"Prediction of Agriculture Crop Production in India"

Prepared by

Name: Krishna Kant

Branch:Computer Science

Semester:6th

Mobile Number:7258989036

Email id:krishnakanty058@gmail.com

College Name:Birla Institute Of Technology

(Patna Campus)

Executive Summary

This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT).

This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks' time.

My project was based on Data Science and Machine learning and my assigned project was

"Prediction of Agriculture Crop Production in India".

Context:

(Agriculture Production in india from 2001-2014)

This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

TABLE OF CONTENTS

1 Preface

I was very excited about the internship from starting period of time and I had completed all the task that they had assigned to me on the weekly basics.in first week they had assigned the task related to uct means about their domain, the technologies they used,major achievement,etc and also gave a task related to projects selection according to their own interest from the various projects lists that they provided me,after that I explored a lot about that and selected agriculture related projects.In second week I had learned about lots of machine learning algorithms and selected the appropriate ones for my project work.In third week I had started the implementation parts according to strategy that I had made and also maked some use case ,block diagram .In fourth week I was continuing my work on implentation part and maked the various machine learning models for various predictions .In fifth week I had checked the performance of various models and also improved the performance of various model who doesnot work very well and In sixth week I had compiled all work and make the final report of my project.that was the all about summary of my works of 6 weeks

relevant Internship play a significant role in career development for many reasons like:

1.Gain Practical Experience,2.Skill Development,3.help me to connect with various people of my professional background 4.Resume Enhancement 5.Help me in personal growth.

My project/problem statement was **Prediction of Agriculture Crop Production in India**

Context

Agricuture Production in India from 2001-2014

Content

This Dataset Describes the Agricuture Crops Cultivation/Production in india. This is

from https://data.gov.in/ fully Licensed Acknowledgements

This Dataset can solves the problems of various crops Cultivation/production in india.

Columns

Crop: string, crop name Variety:string,crop subsidary name

state: string,Crops Cultivation/production Place Quantity:Integer,no of Quintals/Hectars production:Integer,no of years Production Season:DateTime,medium(no of days),long(no of days) Unit:String , Tons

Cost:Integer, cost of cutivation and Production Recommended
Zone:String ,place(State,Mandal,Village)

Inspiration

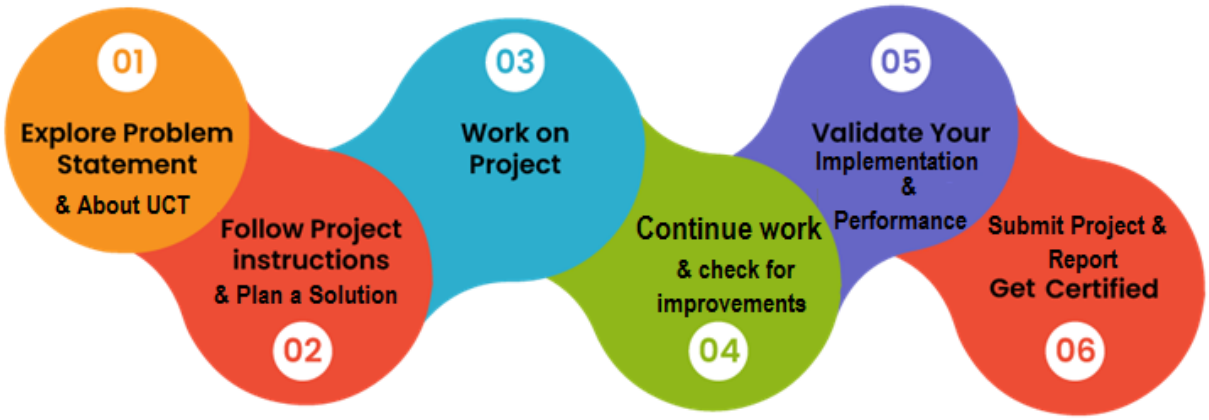
Across The Globe India Is the Second Largest Country having People more than 1.3 Billion.
Many People Are Dependent On The Agricuture And it is the Main Resource.
In Agricuturce Cultivation/Production Having More Problems.
I want to solve the Big problem in india and usefull to many more people

Data set Link:

https://drive.google.com/file/d/1zfqvs8-mAO6E0JpgvhBdueNx8Th03pUp/view?usp=sharing

Opportunity given by USC/UCT.

How Program was planned



I learned lots of things from upskill campus,UCT and mentors through this internship program.i gained practical experience from this internship and gets lots of exposure from the members of UCT .my overall experience is very good with upskill campus and UCT.

Thank to all (upskill campus,UCT ,Kaushlendra Singh Sisodia sir,coordinators,mentors) for helping me directly or indirectly in this internship journey of 42 days.i was lucky to got the support from you all.

Once again Thank You all of you for brought new changes in my life/carrer.

Dear juniors and peers,

I wanted to share my thoughts and experiences as an intern,hoping to provide you with some valuable insights and encoragement for your own carrer journeys.first and foremost ,embrace the opportunities that internships offer.They are stepping stones towards your professional growth and development.Treat yours internships as valuable learning experiences ,where you can apply your academic knowledge in real

-world settings.Networking is key during your internship.Take adavntage of the connections you make with your supervisors,colleagues and industry professionals.These connections can lead to mentorship opportunities ,potential job offers,and a broader professioanl network.

Wishing you all the best in your internships and beyond.Embrace the learning,Seize the opportunities,and let your internships be stepping stones towards a fulfilling and successful career.

Best regards,

Krishna Kant

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

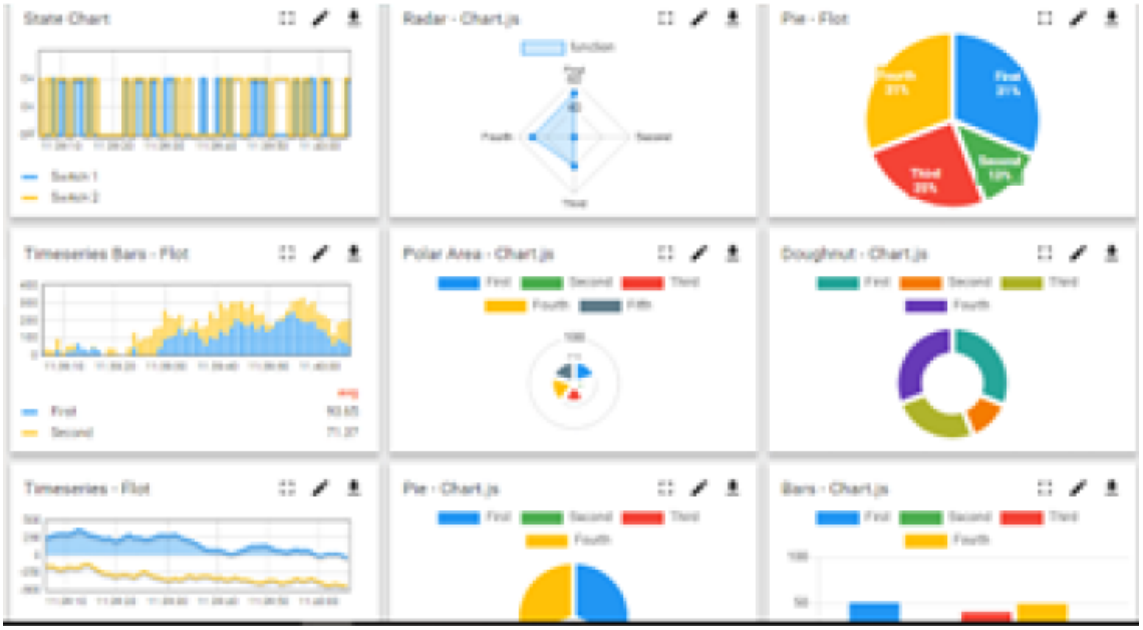
i. UCT IoT Platform (UCT Insight)

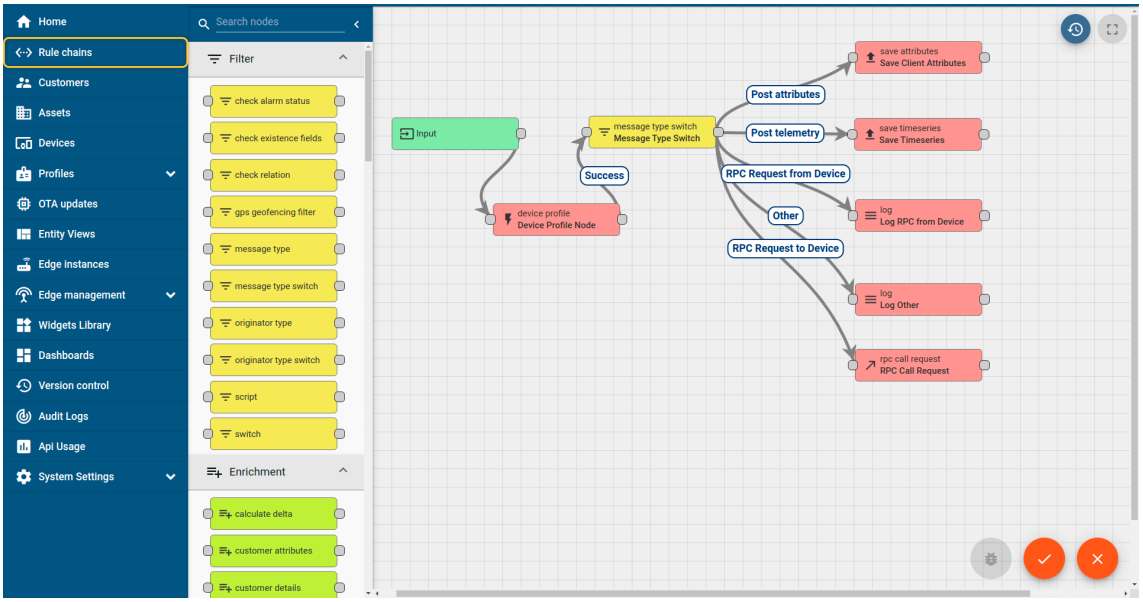
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

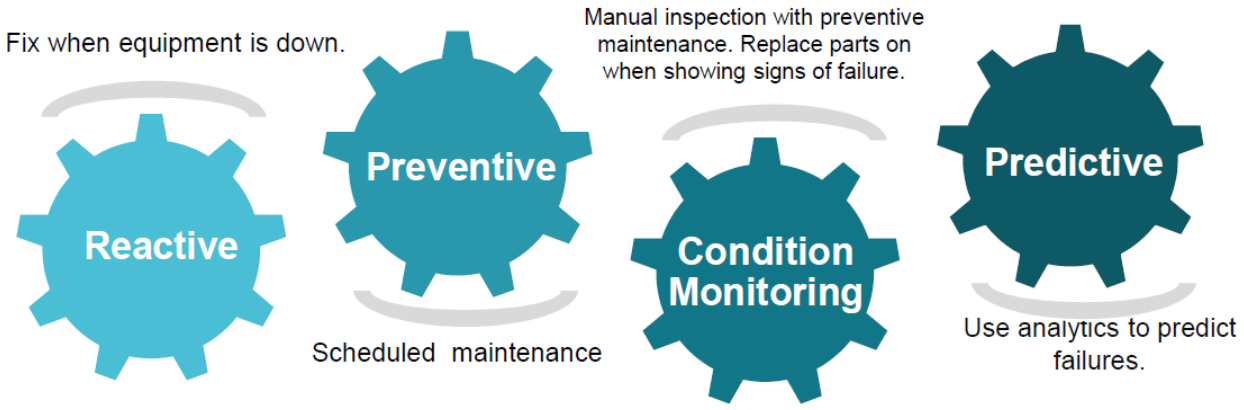
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

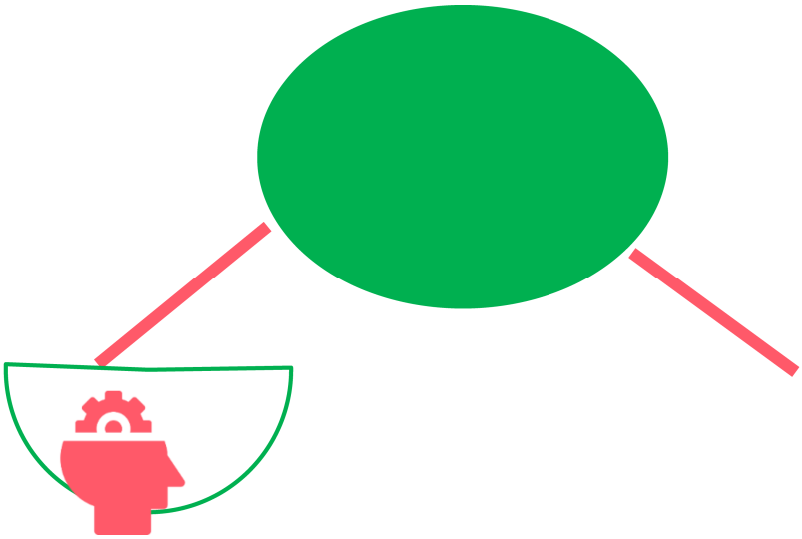
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

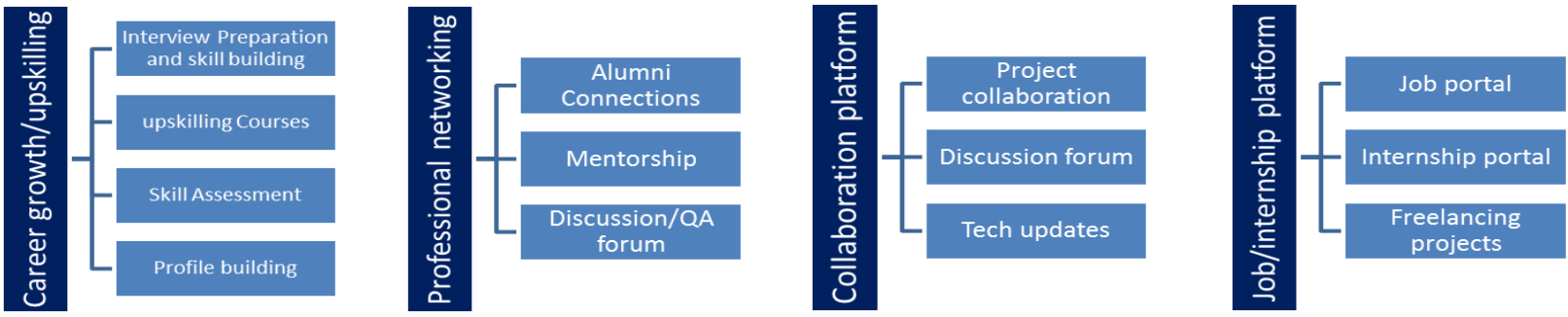
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

- ☛ get practical experience of working in the industry.
- ☛ to solve real world problems.
- ☛ to have improved job prospects.
- ☛ to have Improved understanding of our field and its applications.
- ☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1]Google

[2] Youtube

[3] javapoint

2.6 Glossary

Terms	Acronym

accuracy	The number of correct classification **predictions** divided by the total number of predictions
bagging	A method to **train** an **ensemble** where each constituent **model** trains on a random subset of training examples **sampled with replacement**
boosting	A machine learning technique that iteratively combines a set of simple and not very accurate classifiers (referred to as "weak" classifiers) into a classifier with high accuracy (a "strong" classifier) by **upweighting** the examples that the model is currently misclassifying
categorical data	Categorical features are sometimes called **discrete features**
confusion matrix	An NxN table that summarizes the number of correct and incorrect predictions that a **classification model** made.

data analysis	Obtaining an understanding of data by considering samples, measurement, and visualization. Data analysis can be particularly useful when a dataset is first received, before one builds the first **model**
data set or dataset	A collection of raw data, commonly (but not exclusively) organized in one of the following formats: •a spreadsheet •a file in CSV (comma-separated values) format
decision tree	A supervised learning model composed of a set of **conditions** and **leaves** organized hierarchically
gradient boosting	A training algorithm where weak models are trained to iteratively improve the quality (reduce the loss) of a strong model. For example, a weak model could be a linear or small decision tree model. The strong model becomes the sum of all the previously trained weak models.
matplotlib	An open-source Python 2D plotting library. matplotlib helps you visualize different aspects of machine learning

3 Problem Statement

My project/problem statement was **Prediction of Agriculture Crop Production in India**

Context

Agricuture Production in India from 2001-2014

Content

This Dataset Describes the Agricuture Crops Cultivation/Production in india. This is

from https://data.gov.in/ fully Licensed

Acknowledgements

This Dataset can solves the problems of various crops Cultivation/production in india.

Columns

Crop: string, crop name Variety:string,crop subsidary name

state: string,Crops Cultivation/production Place Quantity:Integer,no of Quintals/Hectars production:Integer,no of years Production Season:DateTime,medium(no of days),long(no of days) Unit:String , Tons

Cost:Integer, cost of cutivation and Production Recommended
Zone:String ,place(State,Mandal,Village)

Inspiration

Across The Globe India Is the Second Largest Country having People more than 1.3 Billion.
Many People Are Dependent On The Agricuture And it is the Main Resource.
In Agricuturce Cultivation/Production Having More Problems.
I want to solve the Big problem in india and usefull to many more people

Data set Link:

https://drive.google.com/file/d/1zfqvs8-mAO6E0JpgvhBdueNx8Th03pUp/view?usp=sharing

BASIC OVERVIEW:

For most developing countries, agriculture is their primary source of revenue. Modern agriculture is a constantly growing approach for agricultural advances and farming techniques. It becomes challenging for the farmers to satisfy our planet's evolving requirements and the expectations of merchants,

customers, etc. Some of the challenges the farmers face are-(i) Dealing with climatic changes because of soil erosion and industry emissions (ii) Nutrient deficiency in the soil, caused by a shortage of crucial minerals such as potassium, nitrogen, and phosphorus can result in reduced crop growth. (iii) Farmers make a mistake by cultivating the same crops year after year without experimenting with different varieties. They add fertilizers randomly without understanding the inferior quality or quantity.

Agriculture gave birth to civilization. India is an agrarian country and its economy largely based upon crop productivity. Thus agriculture is the backbone of all business in India. Now India stands in second rank in worldwide in farm production, Agriculture and allied sectors like forestry and fisheries considered for 14.5% of the GDP in 2015 and about 50% of the total manpower. The economy improvement of agriculture towards India's GDP is strongly declining growth still agriculture is statistically the broadest economic background and plays a significant role in the various socio economic frame work of India. Indian agriculture is affected by various factors such as climate, due to topography, historical, geographical, biological, political, and institutional and socio economic factors. As time passed there are variations in natural factors and nature of technology so policies also changed. So agriculture production performance also changes in drastic path and large gap in different geographic locations of country. The factors that affect agriculture are independent of one another. So this arise risk and the consistent output of food also affected. Agricultural production is mostly affected by environmental factors. Weather influences crop growth and development, causing large intra-seasonal yield variability. In addition, spatial variability of soil properties, interacting with the weather, cause spatial yield variability. Crop agronomic management that is planting, fertilizer application, irrigation, tillage etc., can be used to offset the loss in yield due to effects of weather. As a result, yield forecasting represents an important tool for optimizing crop yield and to evaluate the crop-area insurance contracts. As the climate changes time to time due to the pollution, population, solid waste management, surface and ground water hydrology etc and the impact of climate change in the developing world described by G Yamuna. i get the Motivation from the farmers and i talked to few of them about the Agriculture related things,there problems,how can they managed to Cultivate particular crop in a region,how can they decide to cultivate a particular crops that give maximum yields means about parameters that they used to decide that...so after Knowing lots of things from farmer i have decided to design a machine learning model that make his task easier to decide the best crops for their land,what is yield of that crop,what is cost of cultivation ,what is cost of production ,etc.so,after getting Motivation from their lifestyle i have thinking about that model...

4 My Proposed solution

IN THIS PROJECT I HAD MADE FIVE DIFFERENT DIFFERENT MACHINE LEARNING MODEL TO PREDICT (CROP NAME BY THE HELP OF PRODUCTION COST,YIELD,STATE,CULTIVATION COST),(CROP YIELD BY THE HELP OF STATE NAME,CROPS NAME, CULTIVATION COST,PRODUCTION COST),(CROP NAME BY THE HELP OF RECOMMENDED ZONE,VARITY,DURATION),CULTIVATION COST BY THE HELP OF CROP NAME,STATE NAME,CULTIVATION COST(C2),PRODUCTION COST,YIELD AND 5TH ONE CULTIVATION COST BY CROP NAME, STATE NAME,PRODUCTION COST AND YIELD .

Following are the five machine learning models that i have designed :-

MACHINE LEARNING MODEL NO 1: Design machine Learning model for prediction of crops on the basis of State name,Cost of Cultivation (/Hectare) A2+FL,Cost of Cultivation (/Hectare) C2,Yield (Quintal/ Hectare) ,dataset used:datafile (1).csv

MACHINE LEARNING MODEL NO 2: Recommend crop name by the attributes variety,duration and Recommended Zone .dataset used:datafile (3).csv

MACHINE LEARNING MODEL NO 3: Predict crops yield by the crops name,state name,Cost of Cultivation (/Hectare) A2+FL,Cost of Cultivation (/Hectare) C,Cost of Production (`/Quintal) C2.i have used datset datafile (1).csv

MACHINE LEARNING MODEL NO 4: Predict Cost Of Cultivation (/Hectare)--A2+FL by using the Attribute name Crops,Cost of Cultivation (/Hectare) C2,Cost of Production (/Quintal) C2,Yield (Quintal/ Hectare) and state name.i have used dataset datafile (1).csv

MACHINE LEARNING MODEL 5: Prediction of Cost of Production (/Quintal) C2 FROM THE attributes CROPS NAME,STATE NAME,Cost of Cultivation (/Hectare) A2+FL,Cost of Cultivation (`/Hectare) C2,Yield (Quintal/ Hectare).i have used dataset datafile (1).csv

4.1 Code submission (Github link):-click here for code and report

4.2 Report submission (Github link) :click here for report and code

OR

Github link for both code and report:click here for code and report

or

direct link for both report and code:

https://github.com/Krishna24072003/prediction-of-agriculture-crops-in-india/tree/5e94f23b945d3e4635f99fb229f92440fd724667/uploadagricultureinindia

5 Proposed Design/ Model

First I have done the Data cleaning and data preprocessing part means EDA parts.then after i have done the visualization part in which i have shows the different different scope of agriculture things like leading producer state,crop yield ,crop production in various year,state wise crop production,cultivation cost,recommended zone for various crops and its variety ,year wise productions of different different crops,etc and then after i have done machine learning model designing ,training and testing parts(implementation parts) of different different model and check how efficiently they work and also checked accuracy,precision,recall,f1 score ,R2 score,etc to evaluate the performance of different different machine learning models.in this project i have used various supervised learning Algorithms (classification and regression both) for designing the different different machine learning sub project.

Various Machine Learning Algorithms that i have used in this project are as follows:

1.Linear Regression :Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

2.Logistic regression :Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

3.Support Vector Machine:Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

4.Naïve Bayes algorithm:Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.It is mainly used in text classification that includes a high-dimensional training dataset.Naïve Bayes Classifier is one of the simple and most effective

Classification algorithms which helps in building the fast machine learning models that can make quick predictions.It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

5.Decision Tree: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.The decisions or the test are performed on the basis of features of the given dataset.It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

6.Random Forest:Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

7.Ensemble learning:Ensemble learning gives us credence to the idea of the "wisdom of crowds," it suggests that the choice-making for a more extensive organization of humans is usually higher than that of an individual professional. Another side, ensemble learning refers to a collection (or ensemble) of base newbies or fashions, which are paintings collectively to attain a better very last of the prediction. A single model, also called a base or susceptible learner, may not perform well due to high variance or bias. But, while vulnerable learners are aggregated, they could shape a sturdy learner, as their combination reduces bias or variance, yielding higher model performance. Ensemble learning is a widely used and desired tool learning technique in which more than one person models, often referred to as base models, are blended to produce a powerful ideal of the prediction version. An example of ensemble learning is the Random Forest algorithm.Ensemble learning is frequently illustrated using selection timber as this algorithm may be liable to overfitting (excessive variance and low bias) when it has not been pruned. It could additionally lend itself to underfitting (low variance and extreme bias) when it is very small, like a

decision stump, a decision tree with one stage. While an algorithm overfits or fits its education set, it cannot generalize nicely to new datasets, so ensemble strategies are used to counteract this conduct to allow for the generalization of the model to new datasets. While selection timber can showcase excessive variance or high bias, it is worth noting that it is not the best modelling approach that leverages ensemble learning to find the "sweet spot" in the bias-variance trade-off.

5.1 UseCase Diagram

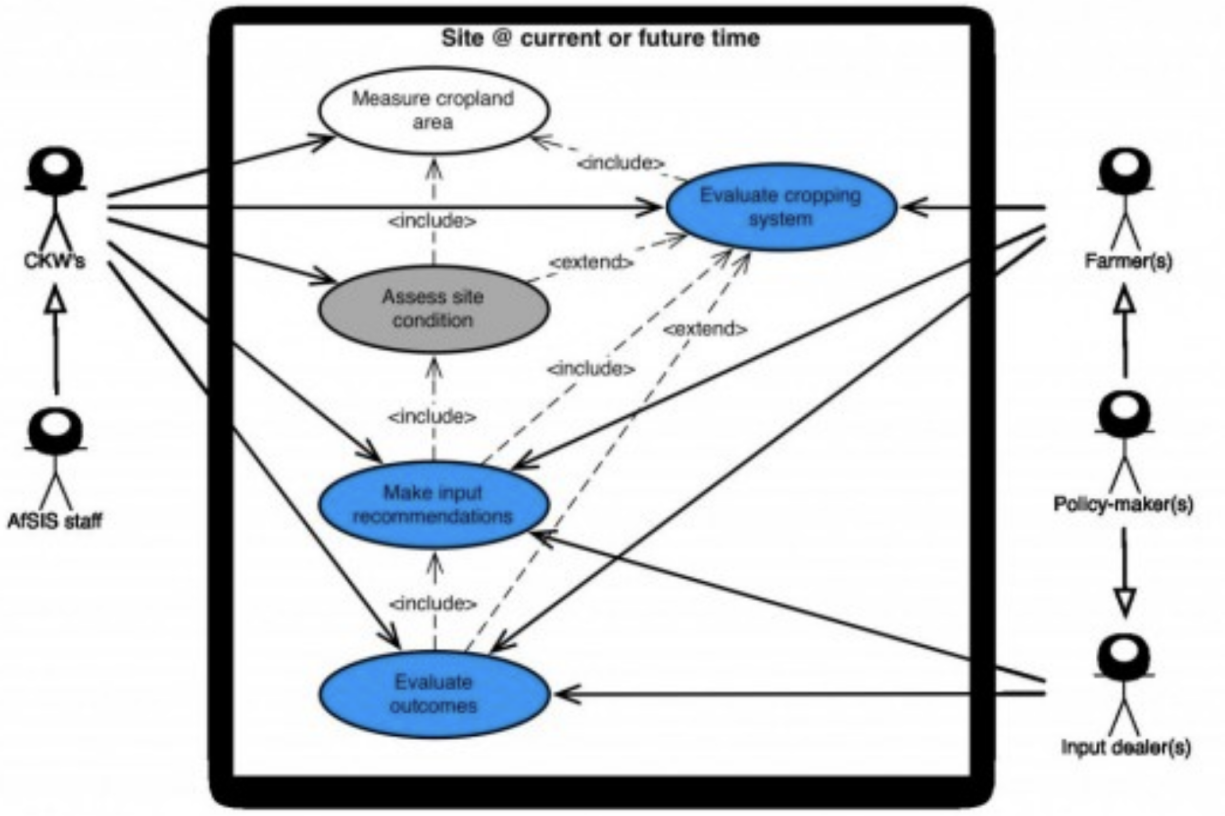


Figure 1: Use Case Diagram of Yield Prediction

5.2 Flow Diagram of Crop Yield predictions

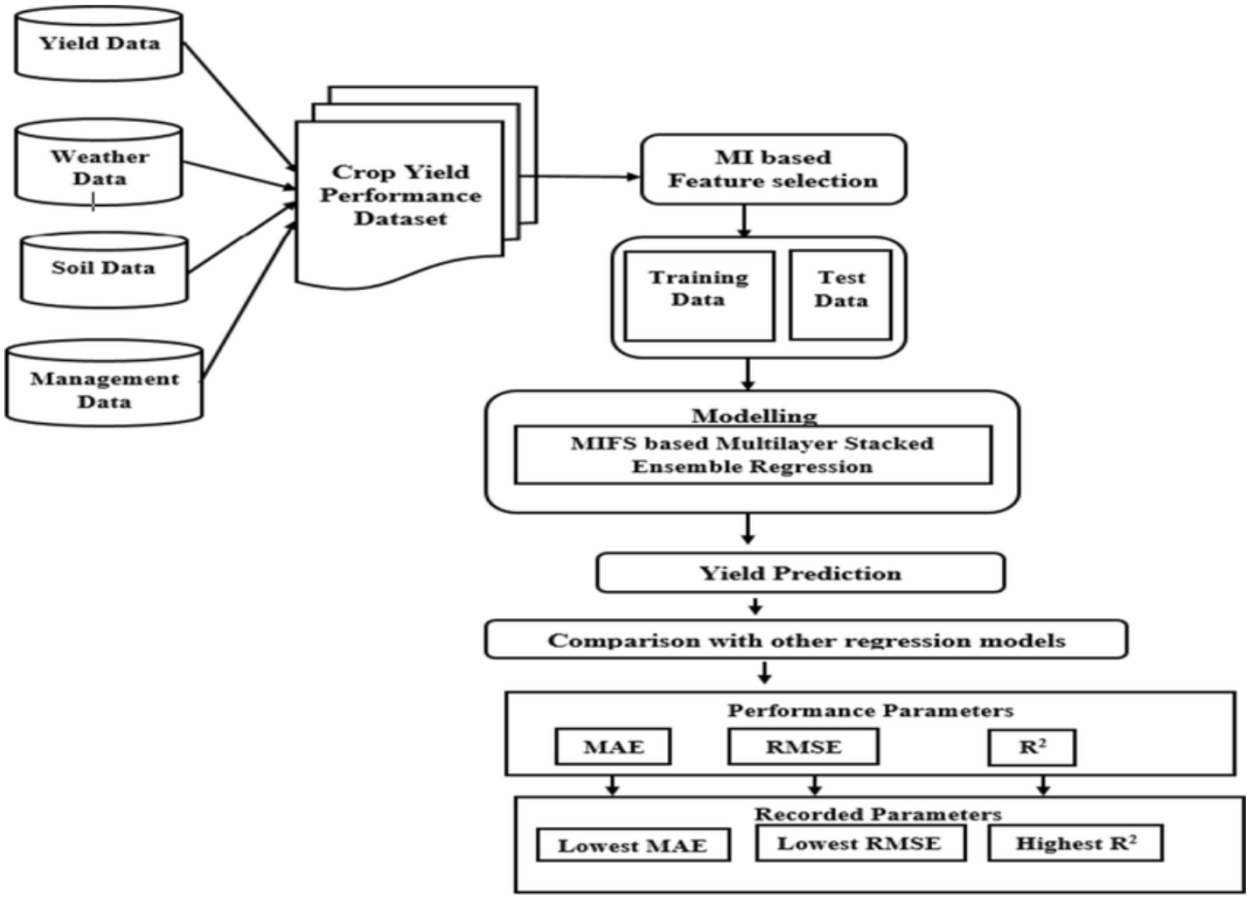
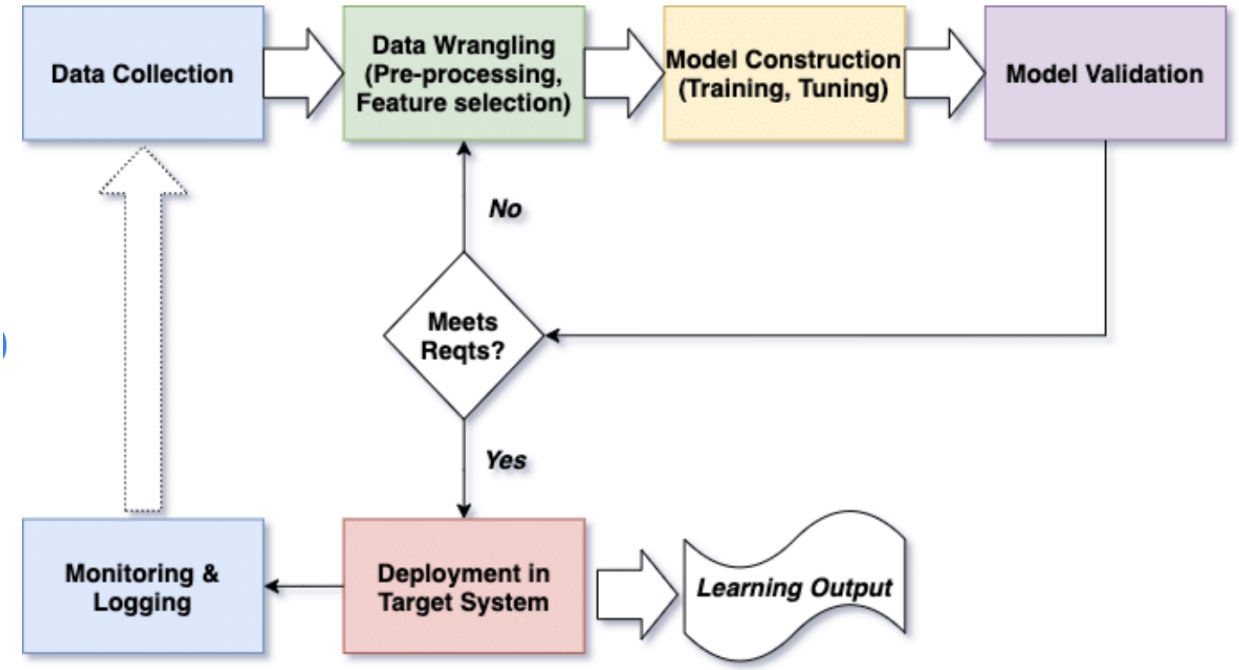
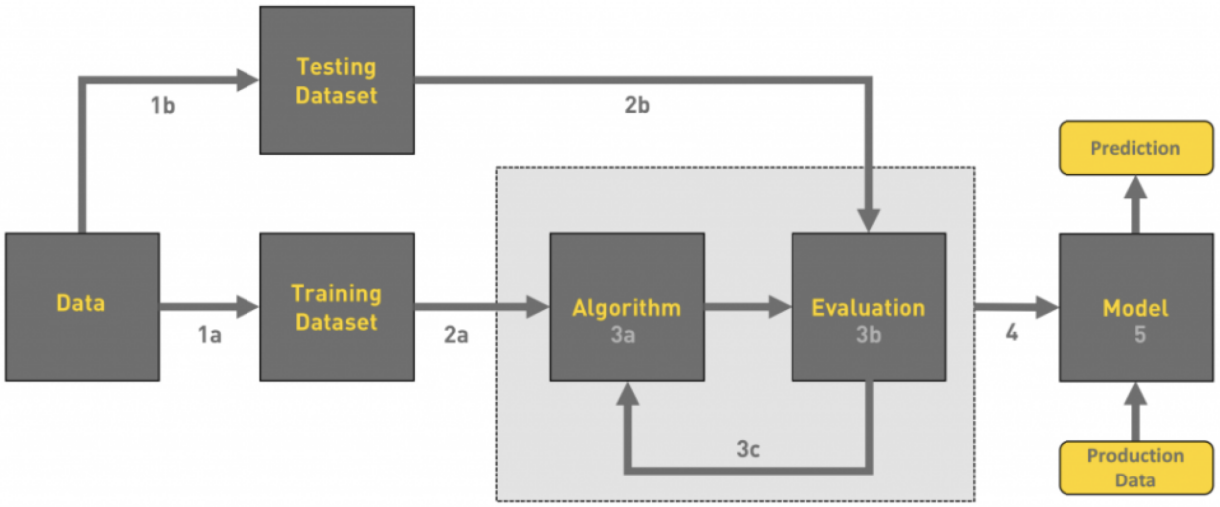


Fig.2 Flow Diagram of Crop Yield predictions

5.3 Interfaces



Block diagram of a generic ML-based system and important activities.



Basic block diagram of machine learning model.

1.Gathering data

2.Data pre-processing

3.Researching the model that will be best for the type of data

4.Training and testing the model

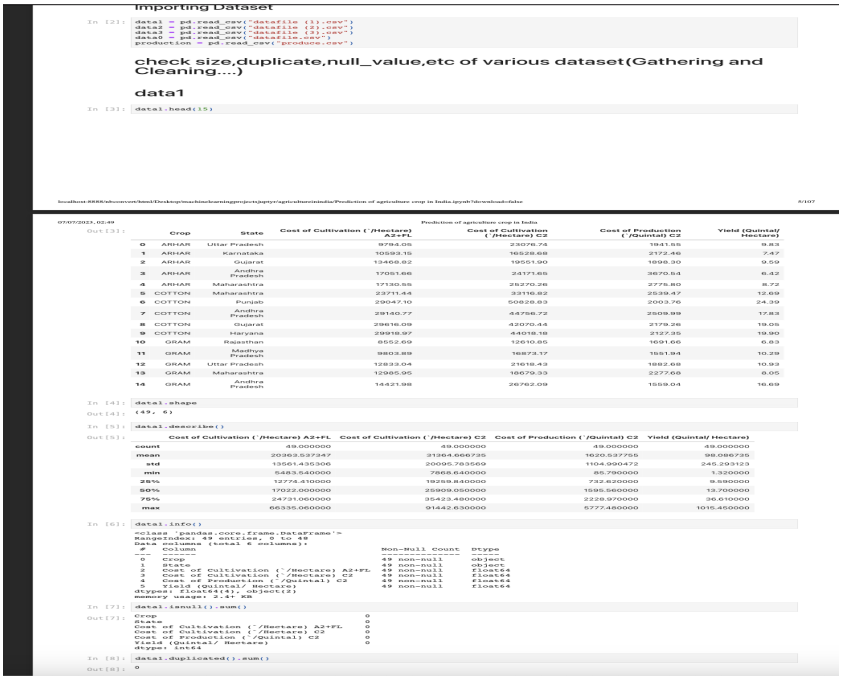
5.Evaluation....

Note:for implementation of model I used Juptyer Notebook

Data pre processing:

Data pre-processing is a process of cleaning the raw data i.e. the data is collected in the real world and is converted to a clean data set. In other words, whenever the data is gathered from different sources it is collected in a raw format and this data isn't feasible for the analysis.
Therefore, certain steps are executed to convert the data into a small clean data set, this part of the process is called as data pre-processing.

I have attached some screenshot of Data pre processing of my model for better understanding.



Data2

```
In [9]: data2.head()
```


Out[9]:

	Crop	Production 2006-07	Production 2007-08	Production 2008-09	Production 2009-10	Production 2010-11	Area 2006-07	Area 2007-08	Area 2008-09	Area 2009-10	Area 2010-11	Yield 2006-07	Yield 2007-08	Yield 2008-09	Yield 2009-10
0	Total Foodgrains	158.8	168.6	171.3	159.4	178.9	128.5	128.8	127.6	126.0	131.7	123.6	130.9	134.3	126.5
1	Rice	200.8	207.9	213.3	191.6	206.4	168.5	168.9	175.1	161.2	164.8	119.2	123.1	121.8	118.5
2	Wheat	131.9	136.4	140.1	140.3	150.8	115.0	115.2	114.0	116.9	119.5	114.4	118.4	122.8	120.5
3	Jowar	124.3	137.8	126.0	116.5	121.8	120.7	110.6	107.3	111.0	105.2	103.0	124.6	117.4	105.5
4	Bajra	136.4	161.5	143.9	105.4	167.9	94.5	95.1	87.0	88.5	95.6	144.3	169.7	165.4	119.5

```
In [10]: data2.shape
```

Out[10]: (55, 16)

```
In [11]: data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55 entries, 0 to 54
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Crop                   55 non-null     object 
 1   Production 2006-07     55 non-null     float64
 2   Production 2007-08     55 non-null     float64
 3   Production 2008-09     55 non-null     float64
 4   Production 2009-10     55 non-null     float64
 5   Production 2010-11     55 non-null     float64
 6   Area 2006-07           55 non-null     float64
 7   Area 2007-08           55 non-null     float64
 8   Area 2008-09           55 non-null     float64
 9   Area 2009-10           55 non-null     float64
 10  Area 2010-11           55 non-null     float64
 11  Yield 2006-07          55 non-null     float64
 12  Yield 2007-08          55 non-null     float64
 13  Yield 2008-09          55 non-null     float64
 14  Yield 2009-10          55 non-null     float64
 15  Yield 2010-11          55 non-null     float64
dtypes: float64(15), object(1)
memory usage: 7.0+ KB
```

```
In [12]: data2.describe()
```

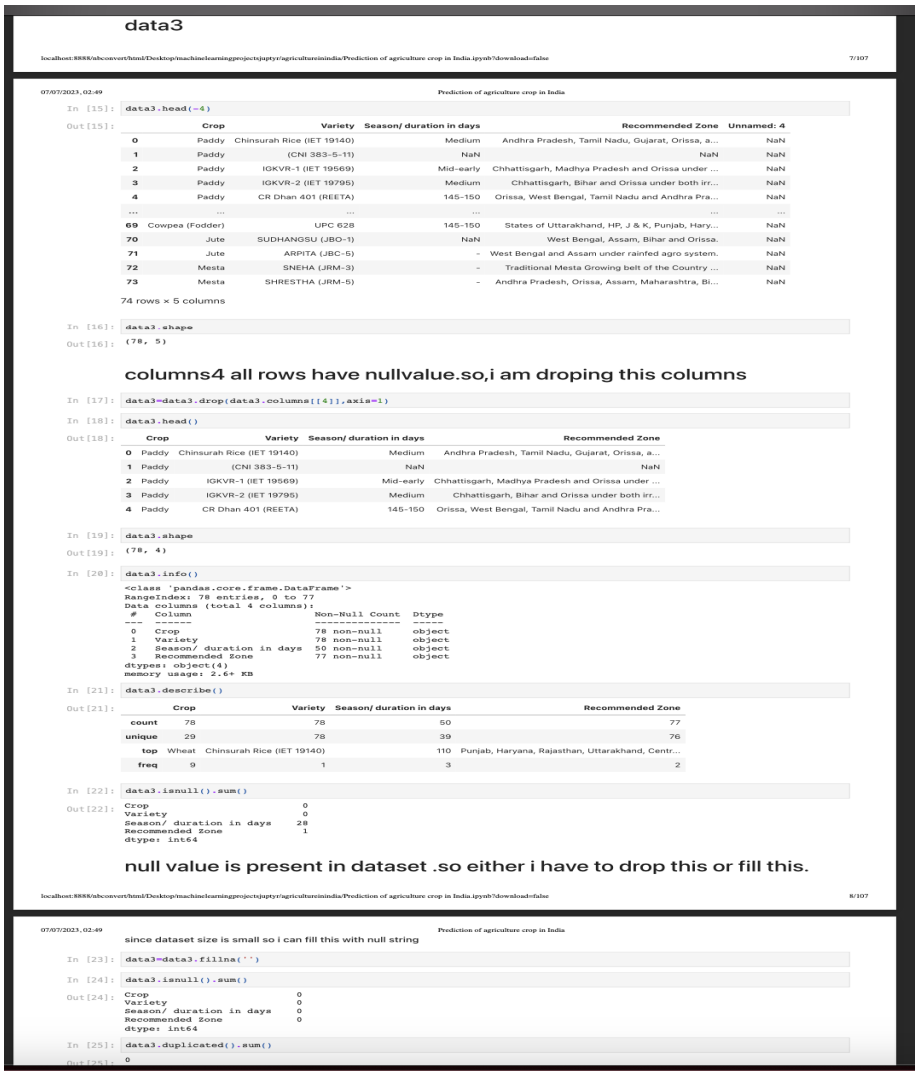
Production 2006-07	Production 2007-08	Production 2008-09	Production 2009-10	Production 2010-11	Area 2006-07	Area 2007-08	Area 2008-09	Area 2009-10	Area 2010-11	Yield 2006-07	Yield 2007-0
55.000000	55.000000	55.000000	55.000000	55.000000	55.000000	55.000000	55.000000	55.000000	55.000000	55.000000	55.00000
168.698182	182.629091	179.952727	173.749091	210.181818	116.025455	118.403636	120.530909	118.203636	127.216364	146.829091	153.77454
179.503163	199.671545	187.409253	184.231322	232.535985	37.990960	41.287797	44.460177	44.976615	54.842653	144.836331	153.24310
53.900000	54.400000	50.000000	42.900000	42.100000	47.900000	47.400000	43.600000	37.500000	36.100000	86.000000	75.30000
124.500000	122.200000	116.050000	105.350000	126.900000	92.550000	93.300000	93.200000	88.650000	90.650000	106.800000	116.80000
142.700000	160.100000	156.500000	140.300000	169.000000	120.700000	121.700000	121.900000	120.900000	129.400000	118.700000	124.50000
184.300000	200.100000	188.550000	197.050000	217.350000	131.050000	135.800000	136.250000	134.800000	147.950000	137.600000	154.40000
1427.700000	1571.500000	1463.000000	1430.300000	1790.600000	222.700000	241.600000	254.500000	260.300000	314.700000	1176.600000	1247.70000

```
In [13]: data2.isnull().sum()
```

```
Crop                   0
Production 2006-07     0
Production 2007-08     0
Production 2008-09     0
Production 2009-10     0
Production 2010-11     0
Area 2006-07           0
Area 2007-08           0
Area 2008-09           0
Area 2009-10           0
Area 2010-11           0
Yield 2006-07          0
Yield 2007-08          0
Yield 2008-09          0
Yield 2009-10          0
Yield 2010-11          0
dtype: int64
```

```
In [14]: data2.duplicated().sum()
```

Out[14]: 0



data3

localhost:8888/nbconvert/html/Desktop/machinelearningprojectjupyter/agricultureinindia/Prediction of agriculture crop in India.ipynb?download=false 7/107

07/07/2023, 02:49 Prediction of agriculture crop in India

In [15]:
```
data3.head(-4)
```

Out[15]:

	Crop	Variety	Season/ duration in days	Recommended Zone	Unnamed: 4
0	Paddy	Chinsurah Rice (IET 19140)	Medium	Andhra Pradesh, Tamil Nadu, Gujarat, Orissa, a...	NaN
1	Paddy	(CNI 383-5-11)	NaN	NaN	NaN
2	Paddy	IGKVR-1 (IET 19569)	Mid-early	Chhattisgarh, Madhya Pradesh and Orissa under ...	NaN
3	Paddy	IGKVR-2 (IET 19795)	Medium	Chhattisgarh, Bihar and Orissa under both irr...	NaN
4	Paddy	CR Dhan 401 (REETA)	145-150	Orissa, West Bengal, Tamil Nadu and Andhra Pra...	NaN
...	...	...	...	...	...
69	Cowpea (Fodder)	UPC 628	145-150	States of Uttarakhand, HP, J & K, Punjab, Hary...	NaN
70	Jute	SUDHANGSU (JBO-1)	NaN	West Bengal, Assam, Bihar and Orissa.	NaN
71	Jute	ARPITA (JBC-5)	-	West Bengal and Assam under rainfed agro system.	NaN
72	Mesta	SNEHA (JRM-3)	-	Traditional Mesta Growing belt of the Country ...	NaN
73	Mesta	SHRESTHA (JRM-5)	-	Andhra Pradesh, Orissa, Assam, Maharashtra, Bi...	NaN

74 rows × 5 columns

In [16]:
```
data3.shape
```

Out[16]:
```
(78, 5)
```

columns4 all rows have nullvalue.so,i am droping this columns

In [17]:
```
data3=data3.drop(data3.columns[[4]],axis=1)
```

In [18]:
```
data3.head()
```

Out[18]:

	Crop	Variety	Season/ duration in days	Recommended Zone
0	Paddy	Chinsurah Rice (IET 19140)	Medium	Andhra Pradesh, Tamil Nadu, Gujarat, Orissa, a...
1	Paddy	(CNI 383-5-11)	NaN	NaN
2	Paddy	IGKVR-1 (IET 19569)	Mid-early	Chhattisgarh, Madhya Pradesh and Orissa under ...
3	Paddy	IGKVR-2 (IET 19795)	Medium	Chhattisgarh, Bihar and Orissa under both irr...
4	Paddy	CR Dhan 401 (REETA)	145-150	Orissa, West Bengal, Tamil Nadu and Andhra Pra...

In [19]:
```
data3.shape
```

Out[19]:
```
(78, 4)
```

In [20]:
```
data3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 78 entries, 0 to 77
Data columns (total 4 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Crop                     78 non-null     object
 1   Variety                  78 non-null     object
 2   Season/ duration in days  50 non-null     object
 3   Recommended Zone         77 non-null     object
dtypes: object(4)
memory usage: 2.6+ KB
```

In [21]:
```
data3.describe()
```

Out[21]:

	Crop	Variety	Season/ duration in days	Recommended Zone
count	78	78	50	77
unique	29	78	39	76
top	Wheat	Chinsurah Rice (IET 19140)	110	Punjab, Haryana, Rajasthan, Uttarakhand, Centr...
freq	9	1	3	2

In [22]:
```
data3.isnull().sum()
```

Out[22]:
```
Crop                      0
Variety                   0
Season/ duration in days    28
Recommended Zone          1
dtype: int64
```

null value is present in dataset .so either i have to drop this or fill this.

localhost:8888/nbconvert/html/Desktop/machinelearningprojectjupyter/agricultureinindia/Prediction of agriculture crop in India.ipynb?download=false 8/107

07/07/2023, 02:49 Prediction of agriculture crop in India

since dataset size is small so i can fill this with null string

In [23]:
```
data3=data3.fillna('')
```

In [24]:
```
data3.isnull().sum()
```

Out[24]:
```
Crop                      0
Variety                   0
Season/ duration in days    0
Recommended Zone          0
dtype: int64
```

In [25]:
```
data3.duplicated().sum()
```

Out[25]:
```
0
```

Data Visualization:

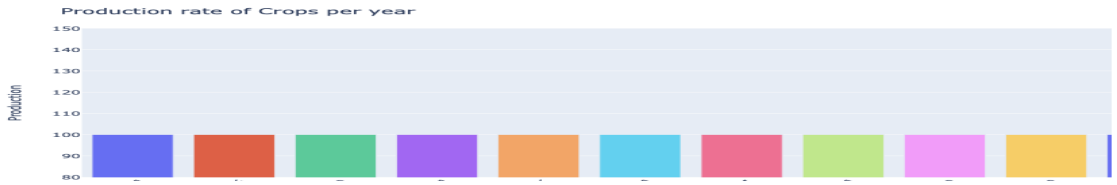
I attached some screenshot of data visualization for better understanding.

Data visualization of all the dataset is started from here:

```
In [66]: data0.shape[0]
Out[66]: 12
```

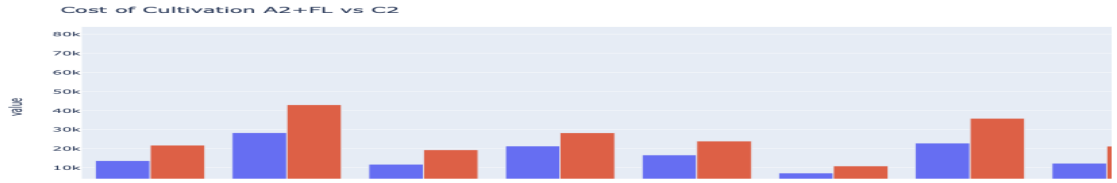
PRODUCTION RATE OF CROPS PER YEAR for this i am using data0

```
In [67]: #PRODUCTION ON RATE OF CROPS PER YEAR -
         #I WILL USE DATASET FRAME data0
         df = data0.copy()
         crops = []
         productions = []
         for i in range(12):
             for _ in range(8):
                 crops.append(df['Crop'][i])
             productions = productions + df.loc[i][1:].tolist()

         years = df.drop('Crop',axis=1).columns.tolist() * 12

         df = pd.DataFrame({'Crop':crops, 'Year': years, 'Production': productions})
         df = df.dropna().reset_index(drop=True)

         fig=px.bar(df,x='Crop', y="Production", animation_frame="Year",
                    animation_group="Crop", color="Crop", hover_name="Crop", range_y=[80,150])
         fig.update_layout(title="Production rate of Crops per year")
         fig.show()
```

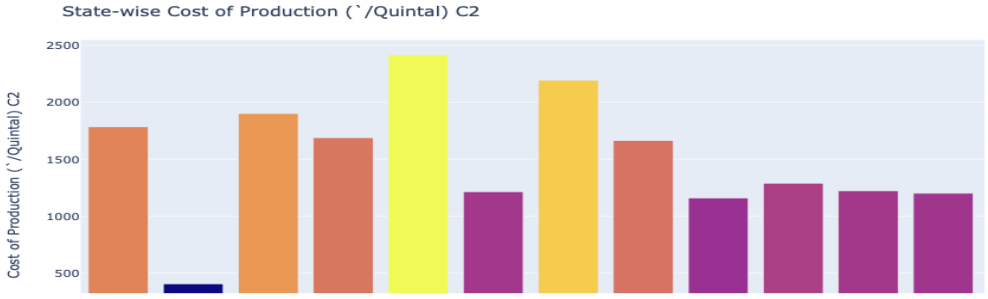
CROP WISE PRODUCTION WHERE I IGNORED STATEDATA

```
In [68]: bargp = data1[["Crop","Cost of Cultivation (`/Hectare) A2+FL","Cost of Cultivation (`/Hectare) C2"]]
         df = bargp.groupby('Crop')[["Cost of Cultivation (`/Hectare) A2+FL","Cost of Cultivation (`/Hectare) C2"]].mean().rese
         fig = df.plot(kind='bar',x='Crop',y=["Cost of Cultivation (`/Hectare) A2+FL","Cost of Cultivation (`/Hectare) C2"],bar
         fig.update_layout(showlegend=False, title='Cost of Cultivation A2+FL vs C2')
         fig.show()
```



NOW PRODUCTION OF CROPS WITH RESPECT OF STATE

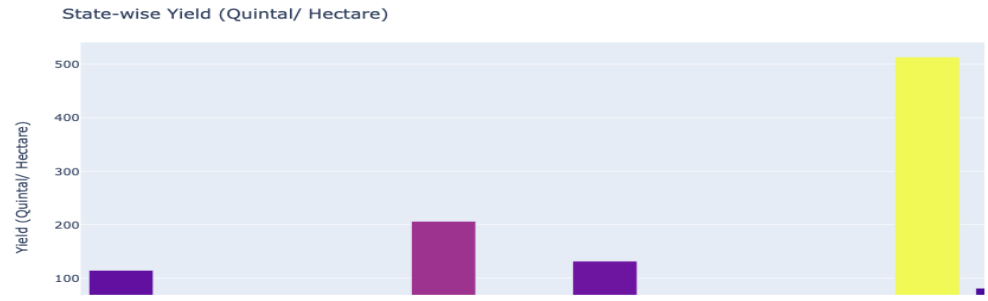
In [69]:
```
#NOW PRODUCTION OF CROPS WITH RESPECT OF STATE
statewise = data1[["State","Cost of Production (`/Quintal) C2"]]
df = statewise.groupby('State')[["Cost of Production (`/Quintal) C2"]].mean().reset_index()
fig = df.plot(kind='bar',x='State',y="Cost of Production (`/Quintal) C2", color="Cost of Production (`/Quintal) C2")
fig.update_layout(title="State-wise Cost of Production (`/Quintal) C2")
fig.show()
```

karnataka has highest cost of production

Yield vs state plot

In [70]:
```
# Yield vs state plot
Yield = data1[["State","Yield (Quintal/ Hectare) "]]
df = Yield.groupby('State')[["Yield (Quintal/ Hectare) "]].mean().reset_index()
fig = df.plot(kind='bar',x='State',y="Yield (Quintal/ Hectare) ", color="Yield (Quintal/ Hectare) ")
fig.update_layout(title="State-wise Yield (Quintal/ Hectare)")
fig.show()
```



now split the data in X and Y means in actual and targeted value

```
In [105…  X=data1.drop(columns='target',axis=1)
          Y=data1['target']

In [106…  print(X)
```


	Cost of Cultivation (`/Hectare) A2+FL	Cost of Cultivation (`/Hectare) C2 \
0	9794.05	23076.74
1	10593.15	16528.68
2	13468.82	19551.90
3	17051.66	24171.65
4	17130.55	25270.26
5	23711.44	33116.82
6	29047.10	50828.83
7	29140.77	44756.72
8	29616.09	42070.44
9	29918.97	44018.18
10	8552.69	12610.85
11	9803.89	16873.17
12	12833.04	21618.43
13	12985.95	18679.33
14	14421.98	26762.09
15	13647.10	17314.20
16	21229.01	30434.61
17	22507.86	30393.66
18	22951.28	30114.45
19	26078.66	32683.46
20	13513.92	19857.70
21	13792.85	20671.54
22	14421.46	19810.29
23	15635.43	21045.11
24	25687.09	37801.85
25	5483.54	8266.98
26	6204.23	9165.59
27	6440.64	7868.64

scale the feature from MinMaxScaler

```
In [108…  #scale the feature from MinMaxScaler
          from sklearn.preprocessing import MinMaxScaler
          scaler=MinMaxScaler()
          X=scaler.fit_transform(X)

In [109…  X
```


Out[109]:
```
array([[7.08365214e-02, 1.81971688e-01, 3.26047272e-01, 8.39142911e-03],
       [8.39684859e-02, 1.03621234e-01, 3.66616945e-01, 6.06431128e-03],
       [1.31225646e-01, 1.39795408e-01, 3.18448475e-01, 8.15477306e-03],
       [1.90104043e-01, 1.95072773e-01, 6.29821723e-01, 5.02894106e-03],
       [1.91400478e-01, 2.08218131e-01, 4.72620610e-01, 7.29689488e-03],
       [2.99547160e-01, 3.02105715e-01, 4.31098672e-01, 1.12115804e-02],
       [3.87230426e-01, 5.14037801e-01, 3.36977242e-01, 2.27485628e-02],
       [3.88769746e-01, 4.41382301e-01, 4.25919191e-01, 1.62799641e-02],
       [3.96580891e-01, 4.09239765e-01, 3.67811669e-01, 1.74829657e-02],
       [4.01558252e-01, 4.32545341e-01, 3.58691355e-01, 1.83211225e-02],
       [5.04367023e-02, 5.67426540e-02, 2.88214291e-01, 5.43322848e-03],
       [7.09982265e-02, 1.07743211e-01, 2.57594844e-01, 8.84501987e-03],
       [1.20777591e-01, 1.64522359e-01, 3.15704123e-01, 9.47610267e-03],
       [1.23290429e-01, 1.29354719e-01, 3.85103546e-01, 6.63623007e-03],
       [1.46889346e-01, 2.26068541e-01, 2.58842277e-01, 1.51558479e-02],
       [1.34155400e-01, 1.13020331e-01, 5.97049383e-01, 3.34276671e-03],
       [2.58752288e-01, 2.70011878e-01, 4.33811399e-01, 1.05016122e-02],
       [2.79768196e-01, 2.69521893e-01, 3.99215347e-01, 1.05114729e-02],
       [2.87055114e-01, 2.66181021e-01, 3.22071300e-01, 1.19609912e-02],
       [3.38448736e-01, 2.96920370e-01, 5.48441675e-01, 7.89839567e-03],
       [1.31966794e-01, 1.43454441e-01, 5.59833722e-02, 4.10499640e-02],
       [1.36550574e-01, 1.53192399e-01, 8.71270220e-02, 2.93650715e-02],
       [1.46880801e-01, 1.42887159e-01, 1.00669573e-01, 2.19301273e-02],
       [1.66830508e-01, 1.57662330e-01, 2.28679004e-01, 1.22075079e-02],
       [3.32013892e-01, 3.58164185e-01, 1.32612634e-01, 4.07837260e-02],
       [0.00000000e+00, 4.76631545e-03, 4.44217798e-01, 1.66645302e-03],
       [1.18434182e-02, 1.55185842e-02, 3.48381588e-01, 2.69196257e-03],
       [1.57284485e-02, 0.00000000e+00, 1.00000000e+00, 0.00000000e+00],
       [1.97306493e-02, 6.39036140e-02, 3.76545455e-01, 4.51618629e-03],
       [8.70515642e-02, 8.97744621e-02, 3.82215124e-01, 5.30503979e-03],
       [1.89616627e-01, 2.42609692e-01, 1.13644629e-01, 3.47983000e-02],
       [1.97111099e-01, 2.15861538e-01, 1.10555916e-01, 3.06666798e-02],
       [3.16303027e-01, 3.01259758e-01, 1.13403927e-01, 3.71944425e-02],
       [3.23265713e-01, 4.47778071e-01, 1.02618027e-01, 6.51691598e-02],
       [3.97382021e-01, 4.61645543e-01, 1.23708424e-01, 5.39181367e-02],
       [5.26345110e-02, 1.17707555e-01, 2.09746139e-01, 1.14580971e-02],
       [9.69928114e-02, 1.36300780e-01, 2.20584747e-01, 1.20497372e-02],
       [1.19814098e-01, 1.75792253e-01, 2.65258649e-01, 1.20793192e-02],
       [1.35692584e-01, 1.34191391e-01, 2.67865959e-01, 1.01466281e-02],
       [1.51709111e-01, 2.34988182e-01, 2.04742352e-01, 1.83605652e-02],
       [3.13135646e-01, 4.47159098e-01, 1.37920372e-03, 4.41333951e-01],
       [8.24497071e-01, 9.44039288e-01, 1.30014108e-04, 9.71167405e-01],
       [8.40367176e-01, 1.00000000e+00, 5.96132256e-03, 7.46058198e-01],
       [8.57662389e-01, 9.32506752e-01, 3.82487451e-03, 7.32342007e-01],
       [1.00000000e+00, 9.71075211e-01, 0.00000000e+00, 1.00000000e+00],
       [1.14719567e-01, 1.74948091e-01, 1.27283812e-01, 2.19597093e-02],
       [2.04794227e-01, 3.29705929e-01, 1.26326276e-01, 3.79734354e-02],
       [2.21783121e-01, 2.87578707e-01, 1.20183988e-01, 3.32008717e-02],
       [2.24078873e-01, 2.63332168e-01, 1.05028559e-01, 3.53702188e-02]])
```

Standardization of data

```
In [110…  #Standardization of data
          scaler=StandardScaler()

In [111…  X=scaler.fit_transform(X)

In [112…  print(X)
```

#NOW TIME FOR GIVING DATA TO TRAINING AND TESTING

TRAINING AND TESTING DATA

In [113…
```
#NOW TIME FOR GIVING DATA TO TRAINING AND TESTING
# TRAINING AND TEST DATA

#testsize=0.22 means give 22percent of data as test data and remaining 80percent as training data
X_train,X_test,Y_train,Y_test =train_test_split(X,Y,test_size=0.22,stratify=Y,random_state=72)
#stratify=Y means data is splitted on the basis of  ....and .. in equal manner
#random_state is used to splitted the data in particular order
```

In [114…
```
print(X.shape,X_train.shape,X_test.shape)
```

```
(49, 4) (38, 4) (11, 4)
```

In [115…
```
print(X_train)
print(Y_train)#ytrain is realoutput
```



```
[[ 0.39661882  0.32364459 -0.71316491 -0.22822047]
 [-0.24674366 -0.3616459   1.87444726 -0.3775755 ]
 [-0.5103147  -0.57853992 -1.11196456 -0.22710834]
 [ 2.70128925  3.02056147 -1.372293    2.71785482]
 [-1.01914848 -0.91280293  0.55632825 -0.37971738]
 [-0.49342345 -0.61746214 -0.00926959 -0.35619788]
 [ 0.06448001 -0.0467608   0.85435589 -0.35471504]
 [-0.56104207 -0.49001507  0.2396933  -0.3589988 ]
 [ 0.65392716  0.6733171   0.81328268 -0.33057769]
 [ 0.69297177  0.75846081 -0.75950486 -0.17335536]
 [-0.71394238 -0.80409674  0.58583476 -0.37642218]
 [-0.56541016 -0.44266033 -0.02283875 -0.34812464]
 [ 0.32539204  0.08453903 -0.81313228 -0.24321364]
 [-0.51367477 -0.59391474  0.25397566 -0.36451827]
 [-1.03729285 -1.18131823  3.80095632 -0.39858242]
 [-0.78672179 -0.72859422 -0.06272328 -0.36163496]
 [-0.54964986 -0.63778526  0.60086689 -0.37086153]
 [-1.10859935 -1.16129074  0.90851364 -0.3916213 ]
 [-0.24895341 -0.16190149 -0.8118796  -0.25322282]
 [ 0.24942765  0.08809364  0.84023812 -0.35174936]
 [-0.42081093 -0.19392615 -0.33778212 -0.32188659]
 [ 0.35695808  0.70019136 -0.86926505 -0.12635754]
 [-0.87993952 -0.94289242  0.06503159 -0.37588671]
 [-0.66887274 -0.60859875 -0.25533396 -0.34824821]
 [-0.1031234   0.0270529  -0.77784699 -0.2598956 ]
 [-0.18014425  0.20406642 -0.74588084 -0.23995963]
 [-0.35225656 -0.51884008 -0.21320923 -0.34758917]
 [ 3.42500061  2.89902298 -1.40331735  3.77862152]
 [ 0.6469485   0.97860677  0.35040444 -0.30355703]
 [-0.09271538 -0.07482814 -0.85671999 -0.2508338 ]
 [-0.50039243 -0.70642039  1.70389103 -0.384619  ]
 [-0.7874549  -0.41669509  0.29352188 -0.3635297 ]
 [ 2.77969882  2.73696296 -1.38341166  2.6605595 ]
 [ 0.42579183  0.06630545  1.45092322 -0.36558921]
 [-0.72791982 -0.74591427  0.50465756 -0.37325055]
 [ 0.68933976  0.53825803  0.51087524 -0.3255525 ]
 [-0.48953367 -0.5376222  -0.94988448 -0.27591852]
 [ 0.1927937  -0.0628576   0.27282984 -0.34861892]]
24    4
3     0
```

I USED RANDOMFOREST CLASSIFICATION BECAUSE THIS MODEL IS BEST FOR MY GIVEN DATASET VALUE and its training and testing accuracy is above 70 percent so I THINK IT WILL PREDICT THE TARGET RESULT BETTER THAN OTHER MODEL.HENCE I AM USING RANDOMFOREST CLASSIFICATION FOR TRAINING THE MODEL

In [117…
```
#I USED RANDOMFOREST CLASSIFICATION BECAUSE THIS MODEL IS BEST FOR MY GIVEN DATASET VALUE
#I THINK IT WILL PREDICT THE TARGET RESULT  BETTER THAN OTHER MODEL

model = RandomForestClassifier()
```

In [118…
```
sk=model.fit(X_train,Y_train)
```

In [119…
```
print(sk)
```

```
RandomForestClassifier()
```

accuracy on training data

In [120…
```
#accuracy on training data

X_train_prediction=sk.predict(X_train).round()
training_data_accuracy=accuracy_score(X_train_prediction,Y_train)
```

In [121…
```
print(X_train_prediction)
```

```
[4 0 4 8 5 7 3 2 1 6 5 7 6 0 5 2 2 5 6 1 7 6 2 7 9 9 4 8 1 9 3 0 8 3 0 1 4
 3]
```

In [122…
```
print(training_data_accuracy)
```

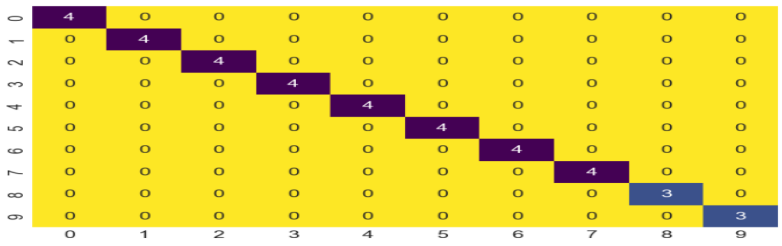
```
1.0
```

training_data_accuracy is 1

Draw confusion matrix for training data

In [123…
```
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(X_train_prediction,Y_train)
sns.heatmap(cm, annot=True, cbar=False, cmap="viridis_r",
            yticklabels=sk.classes_, xticklabels=sk.classes_)
```

Out[123]: <AxesSubplot:>



accuracy on test data

In [124…
```
#accuracy on test data
X_test_prediction=sk.predict(X_test).round()#that is Ypred also
test_data_accuracy=accuracy_score(X_test_prediction,Y_test)
```

In [125…
```
print('Accuracy on test data:',test_data_accuracy)
```

```
Accuracy on test data: 0.7272727272727273
```

Accuracy on test data is around 72 percent

6 Performance Test

This is very important part and defines why this work is meant of Real industries, instead of being just academic project.

Here we need to first find the constraints.

Constraints used in my models for evaluation are:

1.Accuracy:Accuracy simply measures how often the classifier correctly predicts. We can define accuracy as the ratio of the number of correct predictions and the total number of predictions

2.Recall:Recall explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negative is of higher concern than False Positive

3.Precision:Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives

4.Root Mean Squared Error (RMSE):The Root Mean Squared Error (RMSE) has been used as a standard statistical metric to measure model performance in meteorology, air quality, and climate research studies

5.MSE:Mean Squared Error (MSE) calculates the average sum of the squared difference between the actual and predicted value for the entire data points.

6.Confusion Matrix:A confusion matrix is defined as thetable that is often used to describe the performance of a classification model on a set of the test data for which the true values are known.

6.1 Test Plan/ Test Cases:

I had checked the performance of all 5 models using the suitable parameter for them means those model who used classification algorithms then I used the performance measuring parameter as accuracy,precision,recall,etc and those model who used regression algorithms then I used the R2 _score,RMSE,etc.so,in this way I measured the performance of the model.

General Test Plan/Test Cases for Machine Learning Model:

1. Data Preprocessing:

- Test if the data is correctly loaded into the model.
- Test the handling of missing values (if any) by the model.
- Test the correctness of data scaling/normalization.
- Test the handling of categorical variables (if any) by the model.

2. Model Training:

- Test if the model can be trained without any errors.
- Test the convergence of the training process.
- Test if the model is correctly learning from the training data.
- Test the impact of different hyperparameters on model performance.
- Test the ability of the model to handle large datasets (if applicable).
- Test the model's sensitivity to class imbalance (if applicable).

3. Model Evaluation:

- Test the model's performance on a separate validation dataset.
- Test the accuracy of the model's predictions.
- Test the model's precision, recall, and F1-score.
- Test the model's performance on different subsets of the data.
- Test the model's robustness to noise and outliers.
- Test the model's interpretability (if applicable).

These are general categories for test cases and test plan for a machine learning model.

Here I can attached the some screenshot of all the steps of train/test case of model for better understanding of all the steps.

MAKING SECOND MACHINE LEARNING MODEL FOR PREDICTION OF CROPS ..... NOW i will try to Recommend crop name by a attributes variety,duration and Recommended Zone

for the prediction of crop name i will used dataset data3

data preprocessing and data cleaning

In [132…
```
data3.head()
```

Out[132]:

	Crop	Variety	Season/ duration in days	Recommended Zone
0	Paddy	Chinsurah Rice (IET 19140)	Medium	Andhra Pradesh, Tamil Nadu, Gujarat, Orissa, a...
1	Paddy	(CNI 383-5-11)		
2	Paddy	IGKVR-1 (IET 19569)	Mid-early	Chhattisgarh, Madhya Pradesh and Orissa under ...
3	Paddy	IGKVR-2 (IET 19795)	Medium	Chhattisgarh, Bihar and Orissa under both irr...
4	Paddy	CR Dhan 401 (REETA)	145-150	Orissa, West Bengal, Tamil Nadu and Andhra Pra...

In [133…
```
data3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 78 entries, 0 to 77
Data columns (total 4 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   Crop                       78 non-null     object
 1   Variety                    78 non-null     object
 2   Season/ duration in days   78 non-null     object
 3   Recommended Zone           78 non-null     object
dtypes: object(4)
memory usage: 2.6+ KB
```

In [134…
```
data3.describe()
```

Out[134]:

	Crop	Variety	Season/ duration in days	Recommended Zone
count	78	78	78	78
unique	29	78	40	77
top	Wheat	Chinsurah Rice (IET 19140)		Punjab, Haryana, Rajasthan, Uttarakhand, Centr...
freq	9	1	28	2

In [135…
```
data3.shape
```

Out[135]: (78, 4)

In [136…
```
data3['Crop'].value_counts()
```

Out[136]:

```
Wheat                9
Paddy                8
Pearl Millet         7
Maize                6
Indian Mustard       4
Groundnut            3
Cotton               3
```

```
In [138…] data3.isnull().sum()
Out[138]: Crop                          0
          Variety                       0
          Season/ duration in days      0
          Recommended Zone              0
          dtype: int64
In [139…] data3.duplicated().sum()
Out[139]: 0
In [140…] data3.groupby('Crop')
Out[140]: <pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f7ab78eacd0>
In [141…] data3.groupby('Crop').count()
```

Out[141]:

Crop	Variety	Season/ duration in days	Recommended Zone
Barley	3	3	3
Bengal Gram	2	2	2
Chickpea	1	1	1
Cluster Bean	2	2	2
Cotton	3	3	3
Cowpea (Fodder)	1	1	1
Desi Cotton	1	1	1
Fieldpea	3	3	3
Finger Millet	1	1	1
French Bean	1	1	1
Groundnut	3	3	3
Horse Gram	1	1	1
Indian Mustard	4	4	4
Jute	2	2	2
Lentil	2	2	2
Linseed	3	3	3
Maize	6	6	6
Mesta	2	2	2
Mungbean	1	1	1
Napier Bajra Hybrid	1	1	1
Oat	3	3	3
Paddy	8	8	8
Pearl Millet	7	7	7
Sesame	1	1	1
Sugarcane	3	3	3

Out[142]:

	Crop	Variety	Season/ duration in days	Recommended Zone
count	78	78	78	78
unique	29	78	40	77
top	Wheat	Chinsurah Rice (IET 19140)		Punjab, Haryana, Rajasthan, Uttarakhand, Centr...
freq	9	1	28	2

merging the Variety,Season/ duration in days and Recommended Zone

```
In [143…] # merging the   Variety,Season/ duration in days and Recommended Zone
          data3['cp']=data3['Variety']+data3['Season/ duration in days']+data3['Recommended Zone']

In [144…] data3.head()
```

encoding of target i.e Crop Encoding

In [145…
```
#encoding of target i.e Crop
#Encoding
label_encode=LabelEncoder()
```

In [146…
```
labels=label_encode.fit_transform(data3['Crop'])
```

In [147…
```
print(labels)
```

```
[21 21 21 21 21 21 21 21 27 27 27 27 27 27 27 27 27  0  0  0 16 16 16 16
 16 16 22 22 22 22 22 22 22  8 12 12 12 12 28 15 15 15 10 10 10 23  2  1
  1 14 14  7  7  7  9 18 26 26  3  3 11 24 24 24 19 20 20 20 25  5 13 13
 17 17  4  4  4  6]
```

In [148…
```
labels
```

Out[148]:
```
array([21, 21, 21, 21, 21, 21, 21, 21, 27, 27, 27, 27, 27, 27, 27, 27, 27,
        0,  0,  0, 16, 16, 16, 16, 16, 16, 22, 22, 22, 22, 22, 22, 22,  8,
       12, 12, 12, 12, 28, 15, 15, 15, 10, 10, 10, 23,  2,  1,  1, 14, 14,
        7,  7,  7,  9, 18, 26, 26,  3,  3, 11, 24, 24, 24, 19, 20, 20, 20,
       25,  5, 13, 13, 17, 17,  4,  4,  4,  6])
```

In [149…
```
data3['Target']=labels
```

In [150…
```
data3.head(60)
```

78 rows × 5 columns

split the dataset in X and Y

In [153…
```
X=newdata3.drop(columns='Target',axis=1)
Y=newdata3['Target']
```

In [154…
```
print(X)
print(Y)
```

converting the textual data to numerical data

```
In [165…  # converting the textual data to numerical data
          vectorizer = TfidfVectorizer()
          vectorizer.fit(X)

          X = vectorizer.transform(X)

In [166…  print(X)
```

```
  (0, 225)      0.23151205300423863
  (0, 204)      0.23726571339825048
  (0, 176)      0.4168959172319725
  (0, 160)      0.12473586268921583
  (0, 144)      0.22610731984179702
  (0, 133)      0.23726571339825048
  (0, 121)      0.4168959172319725
  (0, 82)       0.17849344223400335
  (0, 76)       0.30521110407128804
  (0, 63)       0.19915874014541193
  (0, 25)       0.147622640835503
  (0, 18)       0.4168959172319725
  (0, 11)       0.24341648266204338
  (1, 22)       1.0
  (2, 166)      0.2318802055749639
  (2, 160)      0.1423963298515425
  (2, 144)      0.25812025350127604
  (2, 126)      0.4346558469787531
  (2, 114)      0.2523030689226532
  (2, 82)       0.2037650642623713
  (2, 77)       0.4346558469787531
  (2, 76)       0.3484237821641815
  (2, 45)       0.4759212568388139
  (2, 25)       0.1685234847843439
  (3, 166)      0.2562875765749
  :     :
  (74, 63)      0.35342969669200636
  (74, 21)      0.7398289297866207
  (75, 228)     0.43413133877154564
  (75, 218)     0.18913052039313774
  (75, 161)     0.2115191591767338
  (75, 160)     0.12989272579506075
  (75, 65)      0.19601044438049314
  (75, 55)      0.43413133877154564
  (75, 20)      0.43413133877154564
  (75, 19)      0.43413133877154564
  (75, 7)       0.3321400304419525
  (76, 221)     0.6344991013184095
  (76, 166)     0.3091431196582845
  (76, 160)     0.18984305076426447
  (76, 144)     0.3441264001664629
  (76, 115)     0.30311172745704035
  (76, 114)     0.3363709189092146
  (76, 63)      0.30311172745704035
```

scale the feature from MinMaxScaler

```
In [167…  #scale the feature from MinMaxScaler
          from sklearn.preprocessing import MaxAbsScaler
          scaler=MaxAbsScaler()
          X=scalers.fit_transform(X)

In [168…  X

Out[168]: <78x230 sparse matrix of type '<class 'numpy.float64'>'
              with 926 stored elements in Compressed Sparse Row format>
```

Standardization of data

```
In [169…  #Standardization of data
          scaler=StandardScaler(with_mean=False)
```



```
In [170…  X=scaler.fit_transform(X)

In [171…  X

Out[171]: <78x230 sparse matrix of type '<class 'numpy.float64'>'
              with 926 stored elements in Compressed Sparse Row format>
```

split the data into training and testing(CROSS VALIDATION)

```
In [172…  X_train, X_test, Y_train, Y_test = train_test_split(X, Y,test_size = 0.2178, random_state=42)









          """from sklearn.model_selection import RepeatedKFold

          kf = RepeatedKFold(n_splits=2,n_repeats=12,random_state=42)


          for train_index, test_index in kf.split(X):
              print('TRAIN:', train_index, 'TEST:', test_index)
              X_train, X_test = X[train_index], X[test_index]


              Y_train, Y_test = Y[train_index], Y[test_index]
          """

Out[172]: 'from sklearn.model_selection import RepeatedKFold\n\nkf = RepeatedKFold(n_splits=2,n_repeats=12,random_state=42)\n\n
          \nfor train_index, test_index in kf.split(X):\n    print('TRAIN:', train_index, 'TEST:', test_index)\n    X_train, X_
          test = X[train_index], X[test_index]\n    \n    \n    Y_train, Y_test = Y[train_index], Y[test_index]\n'

In [173…  X.shape,X_test.shape,X_train.shape

Out[173]: ((78, 230), (17, 230), (61, 230))
```

make a function of different different algorithms to check which one give the better accuracy and after that those have better accuracy will be used for further implementation

In [174…

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.tree import ExtraTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import BaggingRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import ExtraTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.linear_model import LinearRegression
results=[]
names=[]
# create instances of all models
models = {
    #'Linear Discriminant Analysis': LinearDiscriminantAnalysis(),
    'Logistic Regression': LogisticRegression(),
    #'Naive Bayes': GaussianNB(),
    'Support Vector Machine': SVC(),
    'K-Nearest Neighbors': KNeighborsClassifier(),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Bagging': BaggingClassifier(),
    'AdaBoost': AdaBoostClassifier(),
    'Gradient Boosting': GradientBoostingClassifier(),
    'Extra Trees': ExtraTreeClassifier(),

}

from sklearn.metrics import accuracy_score
for name, model in models.items():
    model.fit(X_train, Y_train)
    X_train_prediction = model.predict(X_train)
```



```
    trainacc = accuracy_score( X_train_prediction, Y_train)
    print(f'{name}:\ntrainAccuracy: {trainacc:.4f}')
#for training data
print("TESTING ACCURACY:")
for name, model in models.items():
    model.fit(X_train, Y_train)
    y_pred = model.predict(X_test)
    testacc = accuracy_score(Y_test, y_pred)
    print(f'{name}:\ntestingAccuracy: {testacc:.4f}')
```

```
Logistic Regression:
trainAccuracy: 1.0000
Support Vector Machine:
trainAccuracy: 0.8852
K-Nearest Neighbors:
```

from above result we can clearly see that Gradient boosting give better result .so,it will be use for the further implementation

In [175…
```
#Training the Model: Gradient boosting
```

In [176…
```
#Training the Model: Gradient boosting
model =GradientBoostingClassifier()
```

In [177…
```
model=model.fit(X_train, Y_train)
```

In [178…
```
# accuracy score on the training data
X_train_prediction = model.predict(X_train).round()
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

In [179…
```
print('Accuracy score of the training data : ', training_data_accuracy)
```

```
Accuracy score of the training data :  1.0
```

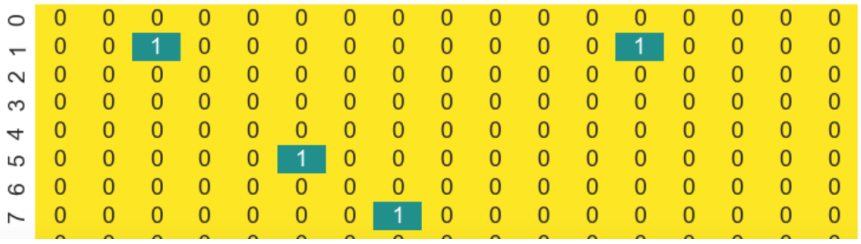
In [180…
```
# accuracy score on the test data
X_test_prediction = model.predict(X_test).round()



test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

In [181…
```
print('Accuracy score of the test data : ', test_data_accuracy)
```

```
Accuracy score of the test data :  0.5294117647058824
```

In [182…
```
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(X_test_prediction,Y_test)
sns.heatmap(cm, annot=True, cbar=False, cmap="viridis_r",
            yticklabels=sk.classes_, xticklabels=sk.classes_)
```

Out[182]: <AxesSubplot:>



Making predictive system

In [185…

```
X_new = X_test[16]

prediction = model.predict(X_new)
print(prediction)

if (prediction[0]==0):
  print('The suitable crop for recommended zone is:Barley')
elif(prediction[0]==1):
    print('The suitable crop for recommended zone is:Bengal Gram')
elif(prediction[0]==2):
    print('The suitable crop for recommended zone is:Chickpea')

elif(prediction[0]==3):
    print('The suitable crop for recommended zone is:Cluster Bean')
elif(prediction[0]==4):
    print('The suitable crop for recommended zone is:Cotton')
elif(prediction[0]==5):
    print('The suitable crop for recommended zone is:Cowpea (Fodder)')
elif(prediction[0]==6):
    print('The suitable crop for recommended zone is:Desi Cotton')
elif(prediction[0]==7):
    print('The suitable crop for recommended zone is:Fieldpea')
elif(prediction[0]==8):
    print('The suitable crop for recommended zone is:Finger Millet')
elif(prediction[0]==9):
    print('The suitable crop for recommended zone is:French Bean')
elif(prediction[0]==10):
    print('The suitable crop for recommended zone is:Groundnut')
elif(prediction[0]==11):
    print('The suitable crop for recommended zone is:Horse Gram')
elif(prediction[0]==12):
    print('The suitable crop for recommended zone is:Indian Mustard')
elif(prediction[0]==13):
    print('The suitable crop for recommended zone is:Jute')
elif(prediction[0]==14):
    print('The suitable crop for recommended zone is:Lentil')
elif(prediction[0]==15):
    print('The suitable crop for recommended zone is:Linseed')
elif(prediction[0]==16):
    print('The suitable crop for recommended zone is:Maize')
elif(prediction[0]==17):
    print('The suitable crop for recommended zone is:Mesta')
elif(prediction[0]==18):
    print('The suitable crop for recommended zone is:Mungbean')
elif(prediction[0]==19):
    print('The suitable crop for recommended zone is:Napier Bajra Hybrid')
elif(prediction[0]==20):
    print('The suitable crop for recommended zone is:Oat')
elif(prediction[0]==21):
    print('The suitable crop for recommended zone is:Paddy')
elif(prediction[0]==22):
    print('The suitable crop for recommended zone is:Pearl Millet')
```



```
elif(prediction[0]==23):
    print('The suitable crop for recommended zone is:Sesame')
elif(prediction[0]==24):
    print('The suitable crop for recommended zone is:Sugarcane')
elif(prediction[0]==25):
    print('The suitable crop for recommended zone is:Tall Fescue Grass')
elif(prediction[0]==26):
    print('The suitable crop for recommended zone is:Urdbean')
elif(prediction[0]==27):
```

```
elif(prediction[0]==21):
    print('The suitable crop for recommended zone is:Paddy')
elif(prediction[0]==22):
    print('The suitable crop for recommended zone is:Pearl Millet')
```



```
elif(prediction[0]==23):
    print('The suitable crop for recommended zone is:Sesame')
elif(prediction[0]==24):
    print('The suitable crop for recommended zone is:Sugarcane')
elif(prediction[0]==25):
    print('The suitable crop for recommended zone is:Tall Fescue Grass')
elif(prediction[0]==26):
    print('The suitable crop for recommended zone is:Urdbean')
elif(prediction[0]==27):
    print('The suitable crop for recommended zone is:Wheat')
elif(prediction[0]==28):
    print('The suitable crop for recommended zone is:Yellow Sarson')
else:
  print('The suitable crop   for recommended zone is:others')
```

```
[22]
The suitable crop for recommended zone is:Pearl Millet
```

Search documents and filenames for text

```
In [186…  print(Y_test[16])
```

```
22
```

6.2 Test Procedure

Test Procedure for Machine Learning Model:

1. Define Test Objectives:

- Clearly define the objectives and goals of the testing process.
- Determine what aspects of the machine learning model need to be tested.
- Identify the desired outcomes and performance metrics for the model.

--→in all five machine learnings models my objectives and goal of the testing process was to select the dataset attributes for actual and target value in such a way that my all the model predict target value very well with good performance score…means selection of attributes (data)for training and testing phase such a that it predict my desired outcome very well .

2. Test Data Preparation:

- Collect or generate the test dataset, representing a realistic scenario.
- Preprocess the test data to align with the model's input requirements.
- Split the dataset into training, validation, and test sets (if applicable).

----→in cross validation assigned 70-80 percent datavalue for training the model and 20-30 percent for testing phase means prediction of target value.

- **3.Test Execution:**

- Train the machine learning model using the training dataset.
- Monitor the training process for any errors or anomalies.
- Evaluate the model's performance using the validation dataset.
- Adjust the model's hyperparameters or architecture if necessary.
- Conduct any necessary retraining cycles based on the results.

4.Performance Testing:

- Measure the model's accuracy, precision, recall, and F1-score.
- Evaluate the model's performance on different subsets of the data.
- Test the model's ability to handle outliers and noisy data.
- Assess the model's robustness to variations in the input data.

We can easily understood all the steps for test procedure From above screenshot that I had attached .

6.3 Performance Outcome

The performance outcome of a machine learning model can be assessed using various evaluation metrics and techniques. The specific metrics used depend on the type of problem being solved (classification, regression, etc.) and the goals of the project. Here are some common performance outcomes for machine learning models:

1. Classification Problems:

- **Accuracy:** Measures the overall correctness of the model's predictions.
- **Precision:** Indicates the proportion of correctly predicted positive instances among all predicted positive instances.

- **Recall:** Measures the proportion of correctly predicted positive instances among all actual positive instances.
- **F1-Score:** Combines precision and recall into a single metric that balances their trade-off.
- **Area Under the Receiver Operating Characteristic Curve (AUC-ROC):** Evaluates the model's ability to distinguish between different classes.

2. Regression Problems:

- **Mean Absolute Error (MAE):** Measures the average absolute difference between the predicted and actual values.
- **Root Mean Squared Error (RMSE):** Similar to MAE but penalizes larger errors more heavily.
- **R-squared (Coefficient of Determination):** Represents the proportion of the variance in the dependent variable explained by the model.

3. Clustering Problems:

- **Silhouette Score:** Quantifies the compactness and separation of clusters.
- **Inertia:** Measures the within-cluster sum of squared distances, reflecting the compactness of the clusters.

4. Time Series Problems:

- **Mean Absolute Percentage Error (MAPE):** Calculates the average percentage difference between predicted and actual values.
- **Mean Squared Error (MSE):** Measures the average of the squared differences between predicted and actual values.

5. Anomaly Detection:

- **True Positive Rate (TPR):** Measures the proportion of true anomalies correctly identified.
- **False Positive Rate (FPR):** Indicates the proportion of non-anomalies incorrectly identified as anomalies.

6. Recommendation Systems:

- **Precision at K:** Measures the proportion of relevant items among the top K recommendations.

- **Mean Average Precision (MAP):** Calculates the average precision across different values of K.

It's important to note that the choice of performance metrics should align with the specific problem, the nature of the data, and the project's objectives. Furthermore, the performance outcome should be interpreted in conjunction with domain knowledge and the context of the problem to assess the overall effectiveness and utility of the machine learning model.

Here is my performance outcome for every model that I make in this project.

For first model:

accuracy on testing data is around:72 percent

I USED RANDOMFOREST CLASSIFICATION BECAUSE THIS MODEL IS BEST FOR MY GIVEN DATASET VALUE and its training and testing accuracy is above 70 percent so I THINK IT WILL PREDICT THE TARGET RESULT BETTER THAN OTHER MODEL.HENCE I AM USING RANDOMFOREST CLASSIFICATION FOR TRAINING THE MODEL

In [117…
```
#I USED RANDOMFOREST CLASSIFICATION BECAUSE THIS MODEL IS BEST FOR MY GIVEN DATASET VALUE
#I THINK IT WILL PREDICT THE TARGET RESULT  BETTER THAN OTHER MODEL

model = RandomForestClassifier()
```

In [118…
```
sk=model.fit(X_train,Y_train)
```

In [119…
```
print(sk)
```
```
RandomForestClassifier()
```

accuracy on training data

In [120…
```
#accuracy on training data

X_train_prediction=sk.predict(X_train).round()
training_data_accuracy=accuracy_score(X_train_prediction,Y_train)
```

In [121…
```
print(X_train_prediction)
```
```
[4 0 4 8 5 7 3 2 1 6 5 7 6 0 5 2 2 5 6 1 7 6 2 7 9 9 4 8 1 9 3 0 8 3 0 1 4
 3]
```

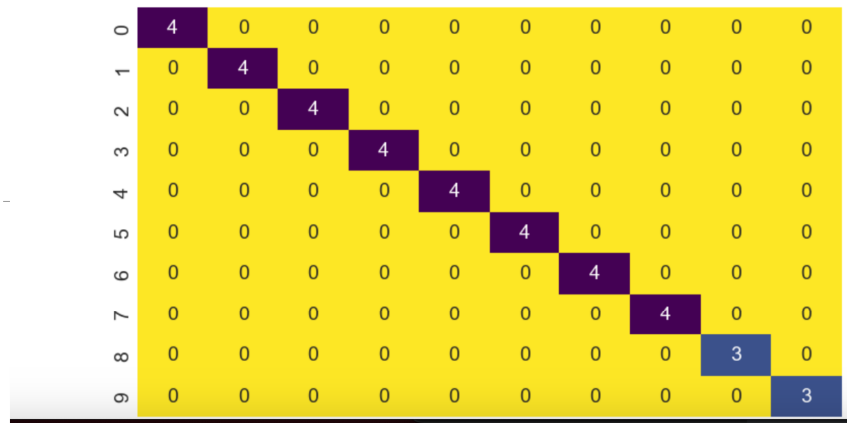
In [122…
```
print(training_data_accuracy)
```
```
1.0
```

training_data_accuracy is 1

Draw confusion matrix for training data

In [123…
```
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(X_train_prediction,Y_train)
sns.heatmap(cm, annot=True, cbar=False, cmap="viridis_r",
            yticklabels=sk.classes_, xticklabels=sk.classes_)
```

Out[123]: <AxesSubplot:>



accuracy on test data

In [124…
```
#accuracy on test data
X_test_prediction=sk.predict(X_test).round()#that is Ypred also
test_data_accuracy=accuracy_score(X_test_prediction,Y_test)
```

In [125…
```
print('Accuracy on test data:',test_data_accuracy)
```

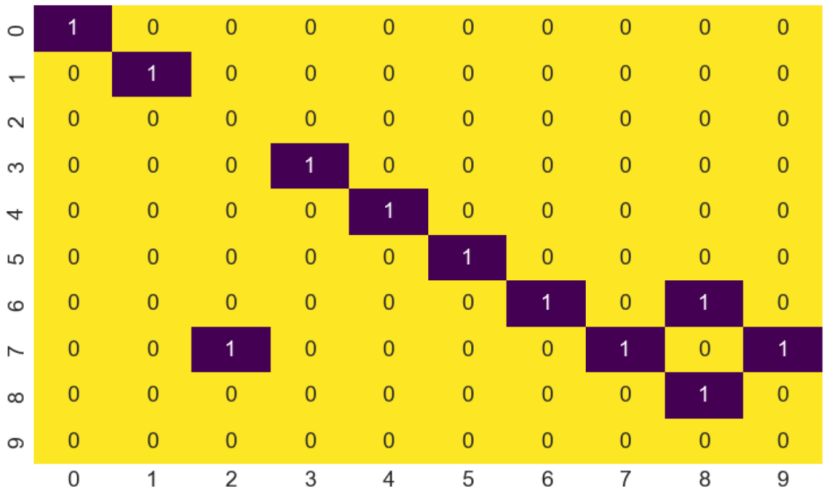
```
Accuracy on test data: 0.7272727272727273
```

Accuracy on test data is around 72 percent

Draw confusion matrix for testing data

In [126…
```
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(X_test_prediction,Y_test)
sns.heatmap(cm, annot=True, cbar=False, cmap="viridis_r",
            yticklabels=sk.classes_, xticklabels=sk.classes_)
```

Out[126]: <AxesSubplot:>



In [127…
```
from sklearn import metrics
# Print the confusion matrix

metrics.confusion_matrix(Y_test, X_test_prediction)
```

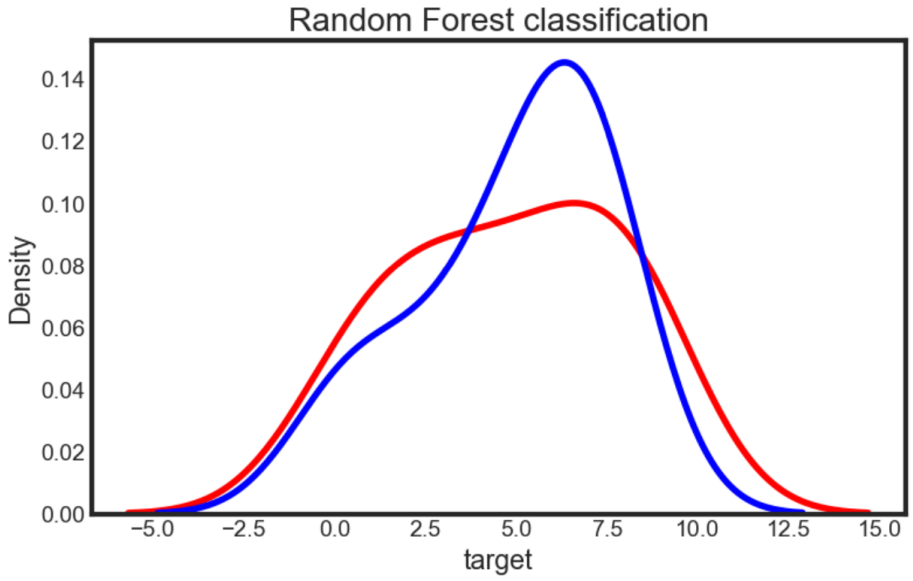
Out[127]:
```
array([[1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
```

plot graph to compare result of actual and target value

In [129…
```
ax=sns.distplot(Y_test,hist=False,color="r",label="Actual value")
sns.distplot(X_test_prediction,hist=False,color="b",label="Predicted value",ax=ax)
plt.title('Random Forest classification')
```

```
/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `kdeplot` (an axes-level function for kernel density plot
s).

/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
plot` (a figure-level function with similar flexibility) or `kdeplot` (an axes-level function for kernel density plot
s).
```

Out[129]: Text(0.5, 1.0, 'Random Forest classification')



0->arhar,1->cotton,2->gram,3->groundnut,4->maize ,5->moong,6->paddy,7->rapeseed and mustard,8->sugarcane,9->wheat

0->arhar,1->cotton,2->gram,3->groundnut,4->maize ,5->moong,6->paddy,7->rapeseed and mustard,8->sugarcane,9->wheat

making a predictive System for testing the data

In [130…
```
# making a predictive System
input_data=(8686.43,17705.93,1279.60,12.94)
#changing the input data to a numpy arrary
input_data_as_numpy_array=np.asarray(input_data)
#reshape the np array s we are predictive for one instance
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
#as i already standarize the data so here for input value we once standarize this
std_data=scalers.transform(input_data_reshaped)
print(std_data)
std_data=scaler.transform(std_data)
```



```
print(std_data)
prediction=sk.predict(std_data)#model=Randomforest classification
print("target:",prediction)
if(prediction==0):
    print("THE SUITABLE CROP IS: ARHAR")
elif(prediction==1):
    print("THE SUITABLE CROP IS: COTTON")
elif(prediction==2):
    print("THE SUITABLE CROP IS: GRAM")
elif(prediction==3):
    print("THE SUITABLE CROP IS: GROUNDNUT")
elif(prediction==4):
    print("THE SUITABLE CROP IS: MAIZE")
elif(prediction==5):
    print("THE SUITABLE CROP IS: MOONG")
elif(prediction==6):
    print("THE SUITABLE CROP IS: PADDY")
elif(prediction==7):
    print("THE SUITABLE CROP IS: RAPESEED AND MUSTARD")
elif(prediction==8):
    print("THE SUITABLE CROP IS: SUGARCANE")
elif(prediction==9):
    print("THE SUITABLE CROP IS: WHEAT")
else:
    print("THE SUITABLE CROP IS OTHER THAN THIS")
```

```
/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning:

X does not have valid feature names, but MinMaxScaler was fitted with feature names

[[0.05263451 0.11770755 0.20974614 0.0114581 ]]
[[-0.86997553 -0.68672524 -0.31174104 -0.35071961]]
target: [7]
THE SUITABLE CROP IS: RAPESEED AND MUSTARD
```

For second model:

Training data accuracy=98-99 percent

testing data accuracy=60-70 percent

In [175…
```
#Training the Model: Gradient boosting
```

In [176…
```
#Training the Model: Gradient boosting
model =GradientBoostingClassifier()
```

In [177…
```
model=model.fit(X_train, Y_train)
```

In [178…
```
# accuracy score on the training data
X_train_prediction = model.predict(X_train).round()
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

In [179…
```
print('Accuracy score of the training data : ', training_data_accuracy)
```

```
Accuracy score of the training data :  1.0
```

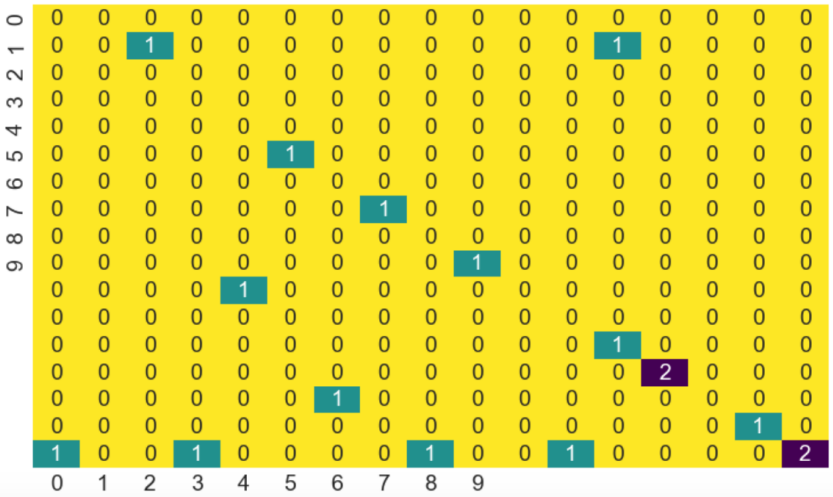
In [180…
```
# accuracy score on the test data
X_test_prediction = model.predict(X_test).round()



test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

In [181…
```
print('Accuracy score of the test data : ', test_data_accuracy)
```

```
Accuracy score of the test data :  0.5294117647058824
```

In [182…
```
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(X_test_prediction,Y_test)
sns.heatmap(cm, annot=True, cbar=False, cmap="viridis_r",
            yticklabels=sk.classes_, xticklabels=sk.classes_)
```

Out[182]: <AxesSubplot:>



Making predictive system

In [185…
```
X_new = X_test[16]

prediction = model.predict(X_new)
print(prediction)

if (prediction[0]==0):
  print('The suitable crop for recommended zone is:Barley')
elif(prediction[0]==1):
    print('The suitable crop for recommended zone is:Bengal Gram')
elif(prediction[0]==2):
    print('The suitable crop for recommended zone is:Chickpea')

elif(prediction[0]==3):
    print('The suitable crop for recommended zone is:Cluster Bean')
elif(prediction[0]==4):
    print('The suitable crop for recommended zone is:Cotton')
elif(prediction[0]==5):
    print('The suitable crop for recommended zone is:Cowpea (Fodder)')
elif(prediction[0]==6):
    print('The suitable crop for recommended zone is:Desi Cotton')
elif(prediction[0]==7):
    print('The suitable crop for recommended zone is:Fieldpea')
elif(prediction[0]==8):
    print('The suitable crop for recommended zone is:Finger Millet')
elif(prediction[0]==9):
    print('The suitable crop for recommended zone is:French Bean')
elif(prediction[0]==10):
    print('The suitable crop for recommended zone is:Groundnut')
elif(prediction[0]==11):
    print('The suitable crop for recommended zone is:Horse Gram')
elif(prediction[0]==12):
    print('The suitable crop for recommended zone is:Indian Mustard')
elif(prediction[0]==13):
    print('The suitable crop for recommended zone is:Jute')
elif(prediction[0]==14):
    print('The suitable crop for recommended zone is:Lentil')
elif(prediction[0]==15):
    print('The suitable crop for recommended zone is:Linseed')
elif(prediction[0]==16):
    print('The suitable crop for recommended zone is:Maize')
elif(prediction[0]==17):
    print('The suitable crop for recommended zone is:Mesta')
elif(prediction[0]==18):
    print('The suitable crop for recommended zone is:Mungbean')
elif(prediction[0]==19):
    print('The suitable crop for recommended zone is:Napier Bajra Hybrid')
elif(prediction[0]==20):
    print('The suitable crop for recommended zone is:Oat')
elif(prediction[0]==21):
    print('The suitable crop for recommended zone is:Paddy')
elif(prediction[0]==22):
    print('The suitable crop for recommended zone is:Pearl Millet')
```



```
elif(prediction[0]==23):
    print('The suitable crop for recommended zone is:Sesame')
elif(prediction[0]==24):
    print('The suitable crop for recommended zone is:Sugarcane')
elif(prediction[0]==25):
    print('The suitable crop for recommended zone is:Tall Fescue Grass')
elif(prediction[0]==26):
    print('The suitable crop for recommended zone is:Urdbean')
elif(prediction[0]==27):
    print('The suitable crop for recommended zone is:Wheat')
elif(prediction[0]==28):
    print('The suitable crop for recommended zone is:Yellow Sarson')
else:
  print('The suitable crop   for recommended zone is:others')
```

```
[22]
The suitable crop for recommended zone is:Pearl Millet
```

In [186…
```
print(Y_test[16])
```

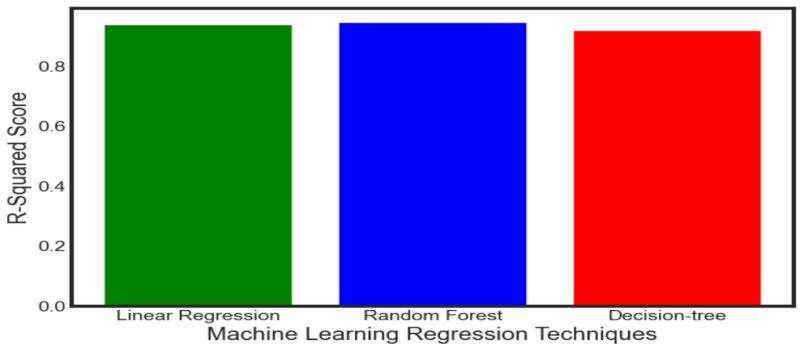
```
22
```

For third model:

Training data R2 score: 98-100 percent

Testing data R2 score: 92 percent

07/07/2023, 02:49 Prediction of agriculture crop in India




```
<Figure size 800x500 with 0 Axes>
```

all algorithms have a very good R squared score.so i have a option to use any one of them

Here I am using Decision tree for further implementation

making a predictive System using Decision tree

In [246…
```
# making a predictive System
input_data=(9794.05,23076.74,1941.55,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0)
#changing the input data to a numpy arrary
input_data_as_numpy_array=np.asarray(input_data)
#reshape the np array s we are predictive for one instance
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
#as i already standarize the data so here for input value we once standarize this
std_data=scalers.transform(input_data_reshaped)
print(std_data)
std_data=scaler.transform(std_data)
print(std_data)
prediction=Dt.predict(std_data)#model=Decision tree
print("Yield prediction:",prediction)
```

```
[[0.07083652 0.18197169 0.32604727 1.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         1.
  1.         0.        ]]
[[-0.7874549  -0.41669509  0.29352188  2.96647939 -0.33709993 -0.33709993
  -0.33709993 -0.33709993 -0.33709993 -0.33709993 -0.33709993 -0.33709993
  -0.2981424  -0.4417261  -0.14433757 -0.2981424  -0.20628425 -0.33709993
  -0.25537696 -0.37354368 -0.20628425 -0.25537696 -0.33709993 -0.20628425
   2.44948974 -0.14433757]]
Yield prediction: [9.83]
/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning:

X does not have valid feature names, but MinMaxScaler was fitted with feature names
```

In [247…
```
pd.set_option('display.max_columns',30)
```

In [248…
```
newdata1.head()
```

Out[248]:

	Cost of Cultivation (`/Hectare) A2+FL	Cost of Cultivation (`/Hectare) C2	Cost of Production (`/Quintal) C2	Yield(Quintal/Hectare)	Cropn_ARHAR	Cropn_COTTON	Cropn_GRAM	Cropn_GROUNDNUT	Cropn_MAI
0	9794.05	23076.74	1941.55	9.83	1	0	0	0	
1	10593.15	16528.68	2172.46	7.47	1	0	0	0	

Yield(Quintal/Hectare)

3.Decision tree

In [236…
```
#training model
Dt = DecisionTreeRegressor(random_state = 5)
Dt.fit(X_train,Y_train)
```

Out[236]:
```
DecisionTreeRegressor(random_state=5)
```

training score

In [237…
```
X_train_prediction = Dt.predict(X_train)
X_train_prediction
```

Out[237]:
```
array([  8.72,  56.  ,  19.05,   6.42,  24.39, 448.89, 986.21,  39.83,
         4.71,  19.9 ,  11.97,  42.68,  67.41,  36.61,   9.83, 744.01,
        39.04,  12.69,   6.7 ,  10.29,  13.54,   7.47,  31.1 ,   9.59,
        13.57,  12.94,  13.7 ,  19.94,   6.83,  23.56,  13.45,  37.19,
        42.95,  17.83, 757.92,  16.69,   5.9 ,  11.61])
```

In [238…
```
Dt.score(X_train,Y_train)
```

Out[238]:
```
1.0
```

In [239…
```
from sklearn.metrics import r2_score
r = r2_score(Y_train,X_train_prediction)
print("R2 score : ",r)
```

```
R2 score :  1.0
```

test score

In [240…
```
X_test_prediction = Dt.predict(X_test)
X_test_prediction
```

Out[240]:
```
array([  4.71,  36.61,  39.83, 744.01,  13.45,   5.9 ,   5.9 ,   5.9 ,
        36.61,  12.69,   4.71])
```

In [241…
```
Dt.score(X_test,Y_test)
```

Out[241]:
```
0.9189735872455934
```

In [242…
```
rs = r2_score(Y_test,X_test_prediction)
print("R2 score : ",rs)
```

```
R2 score :  0.9189735872455934
```

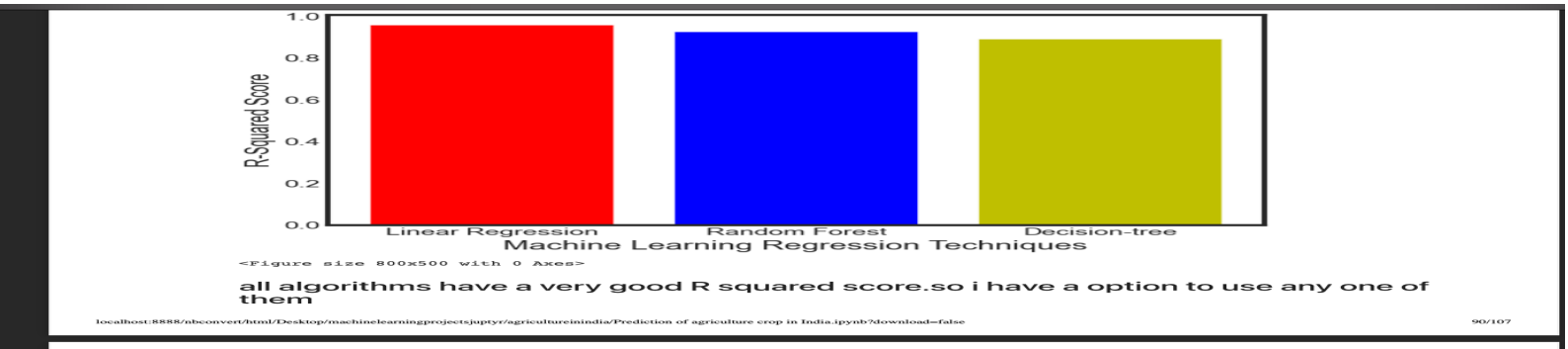
In [243…
```
#Calculating Adj. R2 score:

Adjr2_1 = 1 - (1-r)*(len(Y_test)-1)/(len(Y_test)-X_test.shape[1]-1)
print("Adj. R-Squared : {}".format(Adjr2_1))
```

```
Adj. R-Squared : 1.0
```

4th model

Training R2 score: 98-100 percent

Testing R2 score: 90 percent




```
<Figure size 800x500 with 0 Axes>
```

all algorithms have a very good R squared score.so i have a option to use any one of them

Here I am using Decision tree for further implementation

making a predictive System using Decision tree

In [306…

```
# making a predictive System
input_data=(23076.74,1941.55,9.83,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0)
#changing the input data to a numpy array
input_data_as_numpy_array=np.asarray(input_data)
#reshape the np array as we are predicting for one instance
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
#as i already standarize the data so here for input value we once standarize this
std_data=scalerx.transform(input_data_reshaped)
print(std_data)
std_data=scaler.transform(std_data)
print(std_data)
prediction=Dt.predict(std_data)#model=Decision tree
print("Cost of Cultivation (`/Hectare) A2+FL prediction:",prediction)
```

```
[[0.18197169 0.32604727 0.00839143 1.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         1.         0.        ]]
[[-0.41669509  0.29352188 -0.3635297   2.96647939 -0.33709993 -0.33709993
  -0.33709993 -0.33709993 -0.33709993 -0.33709993 -0.33709993 -0.33709993
  -0.29814224 -0.44172761 -0.14433757 -0.29814224 -0.20628425 -0.33709993
  -0.25537696 -0.37354368 -0.20628425 -0.25537696 -0.33709993 -0.20628425
   2.44948974 -0.14433757]]
Cost of Cultivation (`/Hectare) A2+FL prediction: [9794.05]
/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning:

X does not have valid feature names, but MinMaxScaler was fitted with feature names
```

In [307…

```
pd.set_option('display.max_columns',30)
```

In [308…

```
ccdata1.head()
```

Cost of tivation tectare) A2+FL	Cost of Cultivation (`/Hectare) C2	Cost of Production (`/Quintal) C2	Yield (Quintal/ Hectare)	Cropn_ARHAR	Cropn_COTTON	Cropn_GRAM	Cropn_GROUNDNUT	Cropn_MAIZE	Cropn_MOONG	Cropn_PADDY
9794.05	23076.74	1941.55	9.83	1	0	0	0	0	0	0
0593.15	16528.68	2172.46	7.47	1	0	0	0	0	0	0
3468.82	19551.90	1898.30	9.59	1	0	0	0	0	0	0

3.Decision tree

In [296…

```
#training model
Dt = DecisionTreeRegressor(random_state = 5)
Dt.fit(X_train,Y_train)
```

Out[296]:

```
DecisionTreeRegressor(random_state=5)
```

training score

In [297…

```
X_train_prediction = Dt.predict(X_train)
X_train_prediction
```

Out[297]:

```
array([17130.55, 29664.84, 29616.09, 17051.66, 29047.1 , 24538.32,
       55655.44, 17945.58, 13647.1 , 29918.97, 21229.01, 25687.09,
       25154.75, 17022.  , 9794.05, 57673.6 , 24731.06, 23711.44,
       10780.76,  9803.89, 11385.7 , 10593.15, 13792.85, 13468.82,
       12774.41,  8686.43, 15635.43, 14715.27,  8552.69, 14421.46,
       22951.28, 19119.08, 13513.92, 29140.77, 56621.16, 14421.98,
        6684.18, 13740.64])
```

In [298…

```
Dt.score(X_train,Y_train)
```

Out[298]:

```
1.0
```

In [299…

```
from sklearn.metrics import r2_score
r = r2_score(Y_train,X_train_prediction)
print("R2 score : ",r)
```

```
R2 score :  1.0
```

test score

In [300…

```
X_test_prediction = Dt.predict(X_test)
X_test_prediction
```

Out[300]:

```
array([13468.82, 14421.46, 17945.58, 55655.44, 21229.01, 13647.1 ,
       13647.1 , 13647.1 , 17022.  , 21229.01, 13468.82])
```

In [301…

```
Dt.score(X_test,Y_test)
```

Out[301]:

```
0.8910132122449651
```

In [302…

```
rs = r2_score(Y_test,X_test_prediction)
print("R2 score : ",rs)
```

```
R2 score :  0.8910132122449651
```

In [303…

```
#Calculating Adj. R2 score:

Adjr2_1 = 1 - (1-r)*(len(Y_test)-1)/(len(Y_test)-X_test.shape[1]-1)
print("Adj. R-Squared : {}".format(Adjr2_1))
```

```
Adj. R-Squared : 1.0
```

plot

In [304…

```
ax = sns.distplot(Y_test, hist = False, color = "r", label = "Actual value ")
sns.distplot(X_test_prediction, hist = False, color = "b", label = "Predicted Values", ax = ax)
plt.title('Decision Tree Regression')
```

```
/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
tplot` (a figure-level function with similar flexibility) or `kdeplot` (an axes-level function for kernel density plot
s).

/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `dis
tplot` (a figure-level function with similar flexibility) or `kdeplot` (an axes-level function for kernel density plot
s).
```

5th model

Training R2_score: 96 percent

Testing R2_score: 65-70 percent

In [337…

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.tree import ExtraTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import BaggingRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import ExtraTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.linear_model import LinearRegression
results=[]
names=[]
# create instances of all models
models = {
    #'Linear Discriminant Analysis': LinearDiscriminantAnalysis(),
    #'Logistic Regression': LogisticRegression(),
    #'Naive Bayes': GaussianNB(),
    #'Support Vector Machine': SVC(),
    'linear regression':LinearRegression(),
    'K-Nearest Neighbors': KNeighborsRegressor(n_neighbors=8),
    'Decision Tree': DecisionTreeRegressor(random_state = 8),
    'Random Forest': RandomForestRegressor(n_estimators =8),
```



```
    'Bagging': BaggingRegressor(),
    'AdaBoost': AdaBoostRegressor(),
    'Gradient Boosting': GradientBoostingRegressor(),
    'Extra Trees': ExtraTreeRegressor(),

}

from sklearn.metrics import accuracy_score
print('TRAINING accuracy ')
for name, model in models.items():
    model.fit(X_train, Y_train)
    X_train_prediction = model.predict(X_train)
    rs = r2_score(Y_train,X_train_prediction)
    print(f'{name}:\nR2 score : {rs:4f}')

print("TESTING ACCURACY:")
for name, model in models.items():
    model.fit(X_train, Y_train)
    y_pred = model.predict(X_test)
    rs = r2_score(Y_test,y_pred)
    print(f'{name}:\nR2 score : {rs:4f}')
```

```
TRAINING accuracy
linear regression:
R2 score : 0.965161
K-Nearest Neighbors:
R2 score : 0.488778
Decision Tree:
R2 score : 1.000000
Random Forest:
R2 score : 0.959914
Bagging:
R2 score : 0.962875
AdaBoost:
R2 score : 0.972357
Gradient Boosting:
R2 score : 0.999963
Extra Trees:
R2 score : 1.000000
TESTING ACCURACY:
linear regression:
R2 score : 0.517859
K-Nearest Neighbors:
R2 score : 0.185126
Decision Tree:
R2 score : 0.587763
Random Forest:
R2 score : 0.514346
Bagging:
R2 score : 0.510193
AdaBoost:
R2 score : 0.648337
Gradient Boosting:
R2 score : 0.623162
```

Best on above result , Gradient boosting give best R2 score(Nearly to 65-70 percent) among all.so, for further implementation i am using this

```
In [338…]: ads=GradientBoostingRegressor()

In [339…]: ads.fit(X_train,Y_train)

Out[339]: GradientBoostingRegressor()
```

making a predictive System using Gradient boosting

```
In [340…]: # making a predictive System
input_data=(9794.05,23076.74,9.83,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0)
#changing the input data to a numpy array
input_data_as_numpy_array=np.asarray(input_data)
#reshape the np array s we are predictive for one instance
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
#as i already standarize the data so here for input value we once standarize this
std_data=scalers.transform(input_data_reshaped)
print(std_data)
std_data=scaler.transform(std_data)
print(std_data)
prediction=ads.predict(std_data)#model=Gradient boosting
print("Cost of Production (`/Quintal) C2 prediction:",prediction)
```



```
[[0.07083652 0.18197169 0.00839143 1.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  1.         0.        ]]
[[-0.7874549  -0.41669509 -0.3635297   2.96647939 -0.33709993 -0.33709993
  -0.33709993 -0.33709993 -0.33709993 -0.33709993 -0.33709993 -0.33709993
  -0.2981424  -0.4417261  -0.14433757 -0.2981424  -0.20628425 -0.33709993
  -0.25537696 -0.37354368 -0.20628425 -0.25537696 -0.33709993 -0.20628425
   2.44948974 -0.14433757]]
Cost of Production (`/Quintal) C2 prediction: [1945.17287086]
/Users/pritikumari/opt/anaconda3/lib/python3.9/site-packages/sklearn/base.py:450: UserWarning:

X does not have valid feature names, but MinMaxScaler was fitted with feature names
```

```
In [341…]: pdata1.head()

Out[341]:
```

	Cost of Cultivation (`/Hectare) A2+FL	Cost of Cultivation (`/Hectare) C2	Cost of Production (`/Quintal) C2	Yield (Quintal/ Hectare)	Cropn_ARHAR	Cropn_COTTON	Cropn_GRAM	Cropn_GROUNDNUT	Cropn_MAIZE	Cropn_M
0	9794.05	23076.74	1941.55	9.83	1	0	0	0	0	
1	10593.15	16528.68	2172.46	7.47	1	0	0	0	0	
2	13468.82	19551.90	1898.30	9.59	1	0	0	0	0	
3	17051.66	24171.65	3670.54	6.42	1	0	0	0	0	
4	17130.55	25270.26	2775.80	8.72	1	0	0	0	0	

this model also give proper accurate result on all the value of given dataset.so,this model also work very well as other models.

7 My learnings

During the internship of this machine learning project focused on agricultural crop production in India, several key learnings and insights were gained. The project aimed to leverage machine learning techniques to predict crop yields and identify influential factors for crop production in different regions of India. Here are the key learning points summarized that I learned during the whole project

1. Domain Knowledge and Expertise:

- **Acquiring domain knowledge**: Through this project, i have deepened my understanding of agricultural practices, crop production, and the various factors that influence yields in India. This knowledge strengthens my expertise in the intersection of data science and agriculture, positioning you as a valuable resource in this domain.
- **Collaborating with domain experts**: Engaging with agricultural experts during the project has expanded my network and exposed me to real-world challenges and insights. Collaborative efforts contribute to my professional growth and enable me to bridge the gap between data science and domain-specific knowledge.

2. Data Science Skills:

- **Data preprocessing and feature engineering:** Cleaning and preprocessing agricultural datasets, handling missing values, and deriving meaningful features from the data have enhanced my data manipulation skills. These skills are fundamental in data science projects and can be applied to diverse domains beyond agriculture.
- **Model selection and evaluation:** Exploring and evaluating various machine learning algorithms for yield prediction has improved my understanding of algorithm selection, model training, and performance evaluation. This knowledge can be transferred to other predictive modeling projects across industries.
- **Interpretability and explainability:** Gaining insights into the inner workings of the model and interpreting feature importance have enhanced my ability to communicate complex machine learning concepts to stakeholders. Explainability is a

sought-after skill in the industry, ensuring transparency and trust in machine learning models.

3. Project Management and Problem-Solving:

- **End-to-end project execution:** Successfully completing a machine learning project from data acquisition to model deployment demonstrates my ability to manage and execute projects independently. This experience showcases my project management skills and my capability to navigate challenges throughout the project lifecycle.
- **Problem-solving and critical thinking:** Tackling the complexities and uncertainties inherent in agriculture yield prediction has honed my problem-solving and critical thinking abilities. These skills are invaluable in data-driven decision-making and can be applied to various business problems.

4. Impact and Future Opportunities:

- **Real-world impact:** Demonstrating the practical application of machine learning in agriculture, specifically in crop production, highlights my potential to drive positive change in the industry. The insights and recommendations from my project can contribute to informed decision-making, resource optimization, and sustainable agricultural practices.
- **Career opportunities:** The expertise gained through this project opens up diverse career opportunities in fields such as agricultural technology, precision farming, agronomy, or data science roles within agricultural organizations. my demonstrated ability to apply data science in a domain-specific context makes you a valuable asset in these industries.

i learned all these things that I mentioned above which help me in my personal growth as well as in my career growth

CONCLUSION OF MY PROJECTS: in this project i have design 5 different different models on different different parameter which helps farmers to predict suitable crops for their area,crops cultivation cost ,crop production cost,yield,etc.My project helps the farmers to boost thier productivity of crops.this project helps to bring modernisation in our agriculture system by creating awareness among the farmers

related to various aspects because when i was designing the projects i thinks abouts various aspects and try to cover Maximum of them in all the machine learning model.

8 Future work scope

Overall Future work scope :

1.i will add additional dataset with additional parameter which help me to predict crop name by the soil type,by geographical area,ph value of soil,rainfall,nitrogen level,etc...so,that the scope of prediction of crops increase.

2.Regional variations: Conduct an in-depth regional analysis to understand how crop production is influenced by specific factors, such as climate, soil types, or farming practices, in different states or districts of India.
3.Crop-specific models: Build separate models for different crops, as each crop has unique growth patterns and requirements. Tailor the models to capture crop-specific relationships and optimize predictions accordingly.

4.Crop yield forecasting: Develop models for yield forecasting at different stages of crop growth to provide timely information for farmers, policymakers, and market stakeholders.
5.Early warning systems: Explore the possibility of creating early warning systems to alert farmers about potential risks, such as pest outbreaks, extreme weather events, or diseases, enabling timely interventions and mitigating crop losses.
6.Interdisciplinary research: Explore interdisciplinary approaches by collaborating with experts in fields like agronomy, climatology, economics, or social sciences to gain deeper insights into the multifaceted factors influencing crop production in India.

By outlining the potential future work and research scope, i demonstrated a forward-looking approach and highlight opportunities for continued exploration and innovation in the field of machine learning for agricultural crop production in India. These suggestions pave the way for further advancements, real-world impact, and potential collaborations in the domain of data-driven agriculture.

Github link for my project (that includes both code and report):

https://github.com/Krishna24072003/prediction-of-agriculture-crops-in-india/tree/5e94f23b945d3e4635f99fb229f92440fd724667/uploadagricultureinindia

Thank You