PROJECT DETAILS

1.INTRODUCTION

- Introduction
- Problem statement

2.LITERATURE SURVEY

3.PROPOSED METHODOLOGY

- Proposed methodology includes
- Data Acquisition
- Data Processing

4.REQIREMENTS

- Software Requirements

  4.1.1 Introduction to Python

  4.1.2.1 NumPy

  4.1.2.2 Pandas

  4.1.2.3 Sk-Learn

  4.1.2.4 Matplotlib

  4.1.2.5.Sea born

5.SAMPLE CODE

6.CONCLUSION AND FUTURE SCOPE

INTRODUCTION

- Agriculture has previously been the cornerstone of India's economy, providing sustenance and support for a sizable portion of the population. The constantly growing population presents a unique mix of challenges and events for the agriculture sector. This intricate relationship between agriculture and people demand a careful analysis and a calculated

approach in order to assure sustainable food production, rural development, and economic advancement.

- In recent years, India's farming sector has undergone a shift due to changes in customer tastes, land use patterns, and technological advancements. However, there have been some challenges with this shift. Variables like growing urbanization, fragmented landholdings, water scarcity, climate change, and the need for sustainable practices will have a significant impact on how Indian agriculture develops in the future
- The first section of this analysis will focus on the state of Indian agriculture at the present time, emphasizing the main crops, farming methods, and regional variations. In the sections that follow, the difficulties caused by the growing population will be covered in more detail with an emphasis on the need to increase agricultural productivity, distribute resources fairly, and adopt sustainable practices. This essay will also look at successful initiatives that have made advantage of agriculture's capacity to feed a growing population while minimizing harm to the environment.
- The intricate relationship between agriculture and population in India is used to analyze the current situation and suggest strategic options that could ensure food security, economic prosperity, and social well-being. By considering the shifting population dynamics along with agricultural practices, we can gain insights that guide policy development, technological advancement, and investment strategies.

PROBLEM STATEMENT

The agriculture sector in India plays a crucial role in the country's economy, providing employment and sustenance to a significant portion of the population.Here with the the growing population, there is a need to analyze the current state of agriculture and make strategic decisions to ensure sustainable agricultural practices, food security, and livelihood opportunities for the population.Analyzing Agriculture in INDIA and making strategic decisions based on it according to population

LITERATURE SURVEY

- The relationship between agriculture and population growth has been the subject of extensive inquiry in the literature. According to research, agricultural productivity must be expanded in order to meet the demands of a growing population. They place a strong focus on the value of agricultural diversity, the adoption of innovative technology, and efficient resource management. But argue that unchecked population growth can lead to

ecological deterioration and species extinction, demanding strategies that strike a balance between agricultural expansion and environmental protection.

- A common subject in the literature concerns the effects of urbanization and shifting consumer tastes on Indian agriculture. Researchers like talk about how urbanization is changing how people use the land and making less land available for agriculture. The consumer desire for processed and expensive agricultural products is changing in tandem with this situation. Establish a tactical plan that encourages urban-rural connections and the production of crops that are in line with consumer needs.

- Technological breakthroughs may revolutionize the Indian agriculture industry. Stress the need of biotechnology, remote sensing, and precision agriculture in order to boost productivity and resource efficiency. Technology solutions must be inclusive and specifically crafted to meet the demands of smallholder farmers if they are to be successful. emphasize the importance of sustainable practices, such as organic farming and water-efficient irrigation techniques, for tackling resource scarcity and environmental deterioration.

- To harmonize population dynamics with agricultural dynamics, it is crucial to execute effective policy adjustments. Stress the significance of programs that prioritize land tenure reforms and lending to small farmers Examine how government subsidies impact crop choices and argue in favor of a more sensible approach that encourages the cultivation of nutrient-dense crops. examine the impacts of trade policies on agriculture in further depth and make the case for policies that protect domestic farmers while enhancing global competitiveness.

- A thorough grasp of the intricate interactions between India's population dynamics, agricultural methods, and strategic decision-making is shown by the literature review. It emphasizes the necessity of using a combination of technology innovation, legislative reforms, and sustainable practices to address the issues caused by population expansion, urbanization, and environmental deterioration. Informed decision-making can lead to a

resilient, successful, and sustainable agriculture industry in India thanks to the synthesis of these findings.

PROPOSED METHODOLOGY

Methodology comprises of sequential steps that are followed in the process:
Proposed methodology includes:

- Data Acquisition
- Data Processing
- Output
- Storage

1. Data Acquisition

The process of gathering data from all sources relevant to our issue statement is known as data acquisition.This information can be gathered from a number of sources, including files and databases.The information gathered during this phase is what determines the final outcome.

2.Data Processing:

- Data Preparation

Data preparation is the first step in data process.In this process,data is sorted and filtered to remove useless and invalid data.The data collected is prior process is checked to find missing data and invalid data.This is an essential process since the data collected is used for deeper insights for a larger set of data.This process is essential for having high quality data.This process removes all bad contents in the collected data.Thus as a result,high quality data can be provided to our processing unit.Since the high quality data provides useful insights,this process has to be carefully processed for producing our required result.

This step includes process like removing null values,removing invalid data,removing invalid data.Removal of null values takes place by correcting or modifying the values with the mean or median of that specific column data.Thus after the data is modified with the values,a method called data wrangling is used for selecting the features to use and converting the collected data to a format that could be helpful in further process.
Next step in this process uses data processing methods to generate a desirable output.

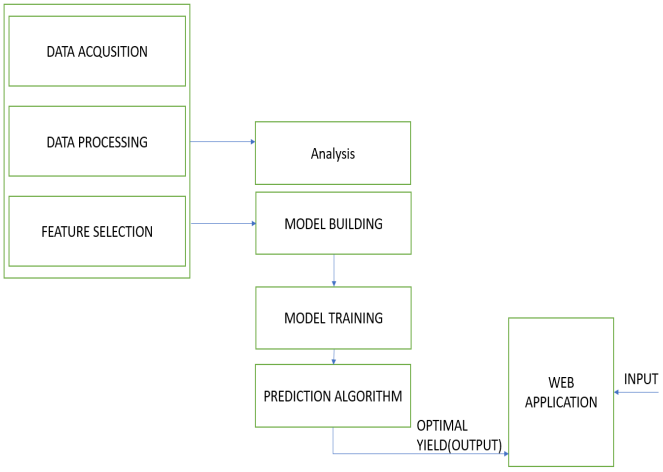
- Output

The data is finally transmitted and displayed to the user in a readable form like graphs, tables, vector files, audio, video, documents, etc. This output can be stored and further processed in the next data processing cycle.

- Storage

The last step of the data processing cycle is storage, where data and metadata are stored for further use. This allows for quick access and retrieval of information whenever needed, and also allows it to be used as input in the next data processing cycle directly.

After the whole data is properly organized, an analysis is done by constructing a graph between states and the production of the crop. This tells which state has higher production rate and which state has lower. This lets us know which state production has to be increased. The other analysis is done between year of the crop production and the production value. This depicts whether the population of the consecutive years are getting increased or not.



Requirements:

Software Requirements:

1. Software:

- Python Version 3.0 or above

2. Operating System: Windows 11

3Tools: Google Collab, Web Browser(Google Chrome/Firefox)

4. Python Libraries: NumPy, pandas, sklearn, matplotlib, sea born

Introduction to Python

Python is a well-liked programming language. It was created by Guido van Rossum and made available in 1991. The use of it as a "scripting language" for web-based applications is quite common. This suggests that it has the ability to automate a specific set of tasks, improving its efficiency.

The advantages of Python are as follows: It is compatible with a number of operating systems, including as Windows, Mac, and Linux.
Python's syntax, which is simple like the English language's, may help programmers write fewer lines of code.
Python operates on an interpreter system, which makes prototyping incredibly efficient.

NumPy

The essential Python library for scientific computing is called NumPy. The library known as NumPy, or "Numerical Python," contains multidimensional array objects and a selection of procedures for handling those arrays. A free and open-source library for Python is called NumPy. Rows and columns make up the robust N-dimensional array object known as NumPy array. We may access its elements and initialize NumPy arrays from nested Python lists. This bundle includes:

A strong N-dimensional array object, fundamental linear algebraic operations, and sophisticated random number abilities.

The scientific Python packages Numeric and Numarray were replaced by NumPy.

Pandas

Python's Pandas package is used to manipulate data sets. It offers tools for data exploration, cleaning, analysis, and manipulation. Both "Panel Data" and "Python Data Analysis" are

mentioned in the term "Pandas." With the aid of Pandas, we can examine large data sets and draw conclusions based on statistical principles. Pandas can organize disorganized data sets, making them readable and useful. In data science, relevant data is crucial..

The Data Frame can, first and foremost, carry information that is: A Pandas Data Frame; A Pandas Series, a one-dimensional labeled array that can hold any type of data with labels or index. One column from a Data Frame serves as an illustration of a Series object.

• A two-dimensional NumPy and array, which can be a record or structured.

Sk-Learn

The most effective and reliable Python machine learning library is called Skearn (Skit-Learn). Through a consistent Python interface, it offers a variety of effective methods for statistical modelling and machine learning, including classification, regression, clustering, and dimensional reduction. A Python module called Scikit-learn offers a variety of supervised and unsupervised learning techniques. It is based on a number of technologies that you may already be acquainted with, including NumPy, pandas, and Matplotlib. The features offered by scikit-learn include:

● Regression, including Linear and Logistic Regression

● Classification, including K-Nearest Neighbors

● Clustering, including K-Means and K-Means++

● Model selection

● Preprocessing, including Min-Max Normalization

Matplotlib

There are numerous plots in Matplotlib, including line, bar, scatter, histogram, etc. The functions in Matplotlib py-plot enable Matplotlib to function similarly to MATLAB. To integrate plots in GUI programs, each pyplot function modifies a figure using matplotlib's APIs (Application Programming Interfaces).

Seaborn

Python's Seaborn package is mostly used for statistical graphing. It is based on Matplotlib and

offers lovely default styles and color palettes to enhance the aesthetic appeal of statistical displays.

Seaborn assists in resolving Matplotlib's two main issues, which are

- Default Matplotlib parameters
- Working with data frames

As Seaborn compliments and extends Matplotlib, the learning curve is quite gradual. If you know Matplotlib, you are already halfway through Seaborn. Seaborn comes with some very important features. The features help in –

- Built in themes for styling matplotlib graphics
- Visualizing univariate and b ivariate data
- Fitting in and visualizing linear regression models
- Plotting statistical time series data
- Seaborn works well with NumPy and Pandas data structures
- It comes with built in themes for styling Matplotlib graphics

Sample Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
df = pd.read_csv("/content/India Agriculture Crop Production.csv")
n_by_state = df.groupby(["Year","State"])["Production"].sum()
```

```
n_by_state.head(10)
n_by_state.tail(10)
df.info()
df.describe()
df.isnull().sum()
df.dropna(inplace = True)
df.isnull().sum()
df.shape
temp = df.groupby(by='State')['Production'].sum().reset_index().sort_values(by='Production')
px.bar(temp, 'State', 'Production')
temp = df.groupby(by='Year')['Production'].sum().reset_index().sort_values(by='Production')
px.bar(temp, 'Year', 'Production')
df['Year'] = df['Year'].apply(lambda x: int(x.split('-')[0]) - 1998)
categorical_features = ["State","District","Crop","Season","Area Units","Production Units"]
numerical_features = ["Year","Area","Production","Yield"]

X = pd.get_dummies(df[categorical_features], columns=categorical_features)
X[numerical_features] = df[numerical_features]
y = df['Production']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the Random Forest regressor model
model = RandomForestRegressor(n_estimators=100, random_state=42)

# Fit the model on the training data
model.fit(X_train, y_train)

# Make predictions on the testing data
predictions = model.predict(X_test)

# Evaluate the model
```

```
mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
print(f"Mean Squared Error: {mse}")
print(f"R-squared Score: {r2}")
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

```
[42] df = pd.read_csv("/content/India Agriculture Crop Production.csv")
```

```
[43] n_by_state = df.groupby(["Year","State"])["Production"].sum()
```

```
[44] n_by_state.head(10)
```

```
Year     State
1997-98  Andhra Pradesh       2.855260e+07
         Arunachal Pradesh    2.671480e+05
         Assam                1.325564e+08
         Bihar                2.164166e+07
         Goa                  7.253800e+04
         Gujarat              2.723810e+07
         Haryana              2.000940e+07
         Jammu and Kashmir    8.428580e+05
         Karnataka            3.778696e+07
         Kerala               5.213065e+09
Name: Production, dtype: float64
```

```
[45] n_by_state.tail(10)
```

```
Year     State
2019-20  Punjab           3.933710e+07
         Rajasthan        4.003626e+07
         Sikkim           1.007820e+05
         Tamil Nadu       4.981512e+09
         Telangana        2.696972e+07
         Tripura          9.148741e+05
         Uttar Pradesh    2.599873e+08
         Uttarakhand      9.010708e+06
         West Bengal      4.330582e+08
2020-21  Uttarakhand      1.017723e+07
Name: Production, dtype: float64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 345407 entries, 0 to 345406
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   State             345407 non-null  object
 1   District          345407 non-null  object
 2   Crop              345375 non-null  object
 3   Year              345407 non-null  object
 4   Season            345406 non-null  object
 5   Area              345374 non-null  float64
 6   Area Units        345407 non-null  object
 7   Production        340414 non-null  float64
 8   Production Units  345407 non-null  object
 9   Yield             345374 non-null  float64
dtypes: float64(3), object(7)
memory usage: 26.4+ MB
```

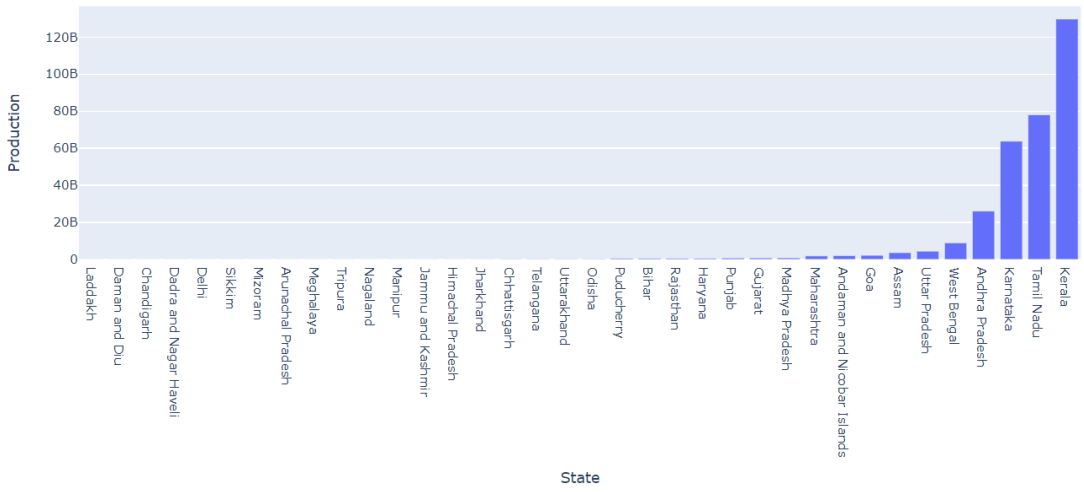
```
df.describe()
```

	Area	Production	Yield
count	3.453740e+05	3.404140e+05	345374.000000
mean	1.167019e+04	9.583711e+05	79.407569
std	4.583843e+04	2.152986e+07	916.628744
min	4.000000e-03	0.000000e+00	0.000000
25%	7.400000e+01	8.700000e+01	0.546742
50%	5.320000e+02	7.170000e+02	1.000000
75%	4.110000e+03	7.176000e+03	2.467080
max	8.580100e+06	1.597800e+09	43958.333330

```
[49] df.isnull().sum()
```

```
State                0
District             0
Crop                32
Year                 0
Season               1
Area                33
Area Units           0
Production        4993
Production Units     0
Yield               33
dtype: int64
```

```
temp = df.groupby(by='State')['Production'].sum().reset_index().sort_values(by='Production')
px.bar(temp, 'State', 'Production')
```



```
df.isnull().sum()
```

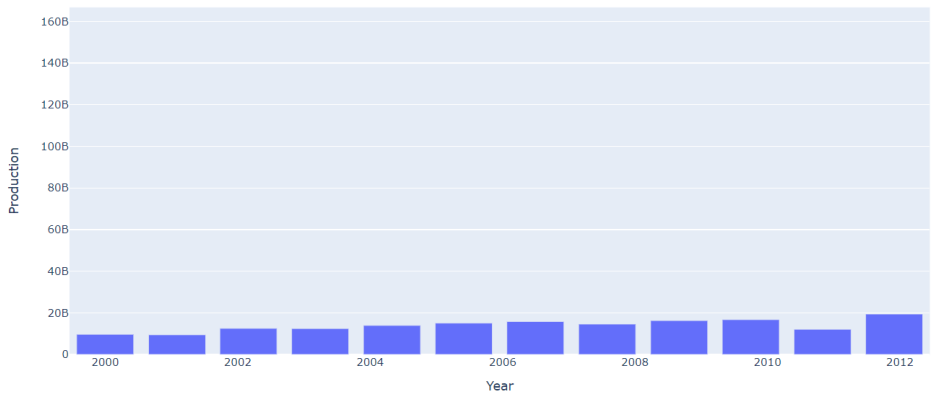
```
State                0
District             0
Crop                32
Year                 0
Season               1
Area                33
Area Units           0
Production        4993
Production Units     0
Yield               33
dtype: int64
```

```
[50] df.dropna(inplace = True)
```

```
[51] df.isnull().sum()
```

```
State               0
District            0
Crop                0
Year                0
Season              0
Area                0
Area Units          0
Production          0
Production Units    0
Yield               0
dtype: int64
```

```
temp = df.groupby(by='Year')['Production'].sum().reset_index().sort_values(by='Production')
px.bar(temp, 'Year', 'Production')
```



```
df['Year'] = df['Year'].apply(lambda x: int(x.split('-')[0]) - 1998)
categorical_features = ["State","District","Crop","Season","Area Units","Production Units"]
numerical_features = ["Year","Area","Production","Yield"]

X = pd.get_dummies(df[categorical_features], columns=categorical_features)
X[numerical_features] = df[numerical_features]
y = df['Production']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the Random Forest regressor model
model = RandomForestRegressor(n_estimators=100, random_state=42)

# Fit the model on the training data
model.fit(X_train, y_train)

# Make predictions on the testing data
predictions = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
print(f"Mean Squared Error: {mse}")
print(f"R-squared Score: {r2}")
```

```
Mean Squared Error: 10326061402.08208
R-squared Score: 0.9999685053244389
```

Conclusion And Future Scope:

Here,In our project we mainly did analysis on agriculture production in India .And how it get impact with growing population .We used Dashboard to predict various graph visuals like State and Production ,Year and Production ,Yield graphs we have showed graphs in different ways.Next we have used python programming language to predict production for upcoming years .We have used Random Forest Classifier to predict the result so, it can sustain its production for growing population.

THANKYOU