

Cardiff Metropolitan University	
Cardiff School of Technologies	
Academic Year : 2023/2024	
Term 2	
Module Name	: Programming for Data Analysis
Module Code	: CIS7031
Module Leader	: Dr Amrita Prasad
Workshop Tutor	: Dr Sandeep Singh Sengar
MSc Program	: MSc Data Science
Assignment Title	: Critical Reflection Report on Crop Yield Dataset
Student Name	: Aaditya Vengatachalapathy
Student Id	: st20290358

Contents

List Of Figures	4
Chapter – 1 - Introduction.....	5
1.1 Crop Yield Dataset.....	5
1.2 Analytical Question	5
1.3 Purpose of the Dataset	6
1.4 Workflow of the Data Analysis.....	7
Chapter – 2 – Working Progress On the Dataset	8
2.1 Dataset Description	8
2.2.1 Python Libraries	9
2.2.2 Import Libraries and Mount the Dataset	10
2.3 Gathering and Cleaning the Data	11
2.3.1 Yield Dataset.....	12
2.3.2 Rainfall Dataset – Climate Factor.....	13
2.3.3 Pesticide Dataset – Another Influential Factor	14
2.3.4 Temperature Dataset	15
2.4 Final Dataframe – yield_df	16
2.5 Exploratory Data Analysis.....	17
2.5.1 Data Visualisation for ‘Top 10 areas with Largest Crop Yield’	19
2.5.2 Top 10 Combinations of Crop Items,Areas by Crop Yield’	20
2.5.3 Data Visualization for ‘Crop Yield Over The Years’	21
2.5.4 Data Visualization for ‘Crop Yield Vs Average Rainfall’	22
2.5.6 Correlation Analysis	23
2.5.7 Data Pre-Processing – One-Hot-Encoding.....	24
2.5.8 Normalization	26
Chapter – 3 – Model Selection	28
3.1 Train and Split Data	28
3.2 Model Selection	29

Chapter – 4 – Key Visualizations based on Model Evaluation	31
4.1 Scatterplot for R^2 Score	31
4.2 Boxplot Visualisation on Crop Yield	34
4.3 Factors Influenced in Decision Making	35
Conclusion	36
References	37
Appendices	40
A1. Rainfall Dataset	40
A2. Pesticide Dataset	40
A3. Yield Dataset	41
A4. Temperature Dataset	41
A5. Final Data-frame	42
A6. Google Colaboratory	42

List Of Figures

Figure 1 - Import Libraries and Mount the Dataset	10
Figure 2 Rename column Value as hg/ha_yield	12
Figure 3 Dropping the unnecessary columns	12
Figure 4 Convert NaN values to empty or other string	13
Figure 5 Merged data-frame.....	13
Figure 6 Rename and Drop the column	14
Figure 7 Rename and Merge the column	15
Figure 8 yield_df dataframe.....	16
Figure 9 yield_df dataframe.....	18
Figure 10 Top 10 Areas with Largest Crop Yield.....	19
Figure 11 Top 10 Combinations of Crops, Areas by Crop Yield	20
Figure 12 Crop Yield Over The Years	21
Figure 13 Crop Yield Vs Average Rainfall.....	22
Figure 14 Correlation between the columns of the dataset	23
Figure 15 Encoding Categorical Variables	24
Figure 16 Dropping Year Field from X Dataframe	25
Figure 17 Normalization	26
Figure 18 Train and Split Data	28
Figure 19 Model Evaluation and Accuracy rate	29
Figure 20 R2 Score for Crop Items	32
Figure 21 Scatterplot for Actual Vs Predicted.....	32
Figure 22 Boxplot Visualization on Crop Yield	34
Figure 23 7-Factors Influenced In crop Yield Dataset	35

Chapter – 1 - Introduction

1.1 Crop Yield Dataset

A new technique called machine learning is used to comprehend the structure of data and fit it into models that may be used to make predictions in the future. The parameters in the data set selected for training the learning models may have an impact on the machine learning models' ability to forecast crop production in addition to the models themselves. The core of the Indian economy is agriculture. Predicting crop yields is a significant agricultural issue. Our proposed model uses supervised machine learning techniques to analyse multiple aspects such as crop type, state, season (assuming same weather and soil conditions in a certain region) in order to forecast the crop yield in advance. This aids farmers in anticipating crop yields (Harika et al., 2877)

Although there are significant regional variations in cuisine, most basic human needs are met by a comparable set of components. We consume a lot of simple crops like corn, wheat, and rice. In this study, machine learning techniques are applied to calculate the yield of top 10 most consumed yields globally ([kaggle.com](https://www.kaggle.com), n.d.).

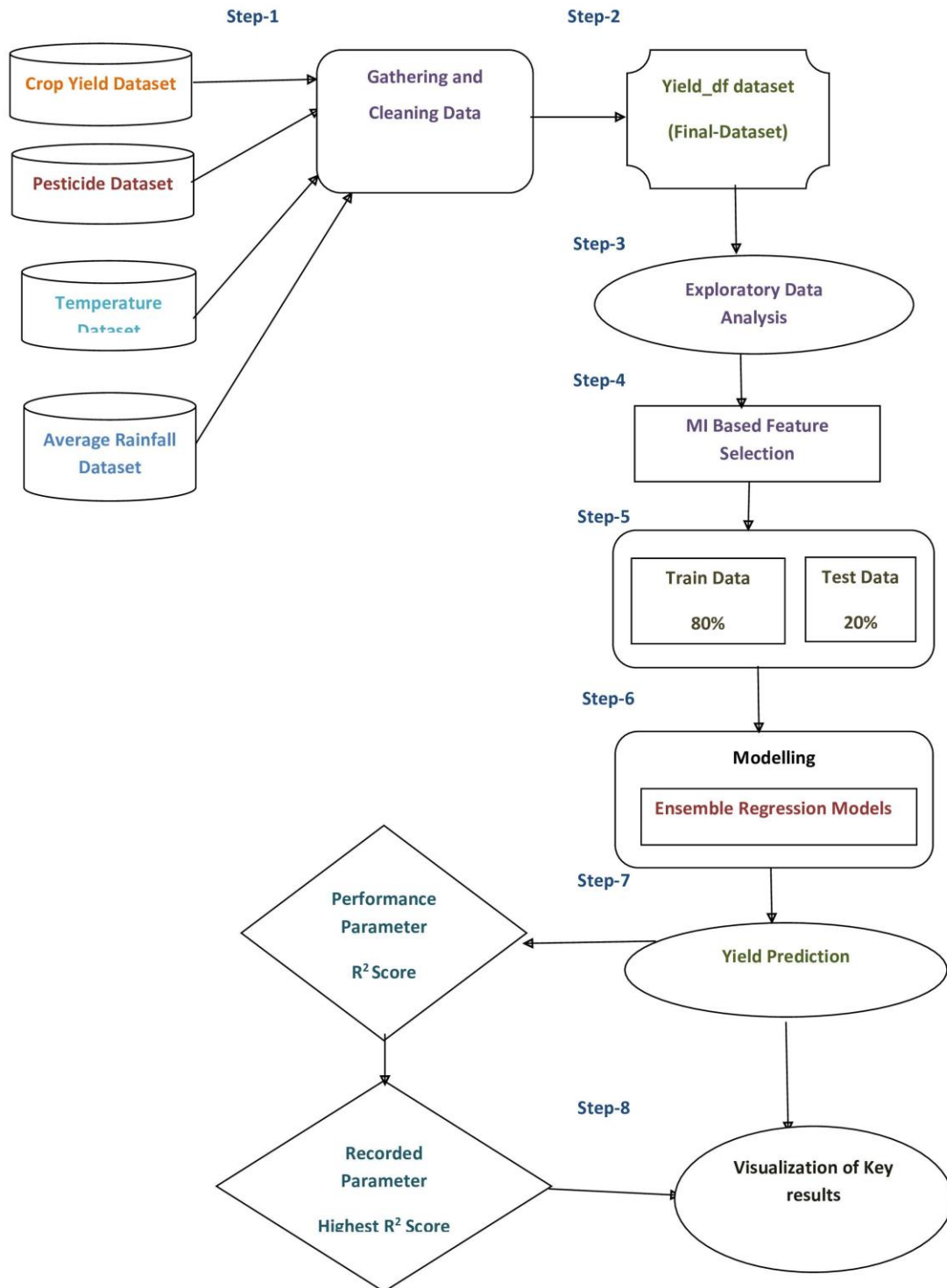
1.2 Analytical Question

- What patterns or trends can be seen in crop yield data over time, and how can past yield data be used to predict yields in the future and help farmers choose crops and cultivate their land wisely? (Harika et al., 2877)
- What are the main climatic parameters influencing successful agricultural growth, and how do varying climatic conditions across world regions affect the production of different crops? (Harika et al., 2877)
- How can machine learning techniques be used to properly estimate crop yields under different environmental conditions? What are the primary traits or features that significantly affect crop yield prediction models? (Harika et al., 2877)

1.3 Purpose of the Dataset

Knowing how weather affects crop yield can help consumers, policymakers, and farmers all gain important insights. With the use of this data, farmers will be able to make well-informed choices on crop selection, planting dates, irrigation plans, and pest control techniques. Using this information, policymakers can create agricultural policies and support initiatives that effectively reduce the negative effects of unfavourable weather occurrences on crop productivity and food security. As agricultural productivity is maximized, consumers may benefit from more consistent food supply, higher-quality produce, and maybe lower pricing. In general, the examination of the dataset's weather-crop yield connections helps both agricultural stakeholders and the larger objective of guaranteeing food security and sustainability for the general public (Malhi, Kaur and Kaushik, 2021).

1.4 Workflow of the Data Analysis



Chapter – 2 – Working Progress On the Dataset

2.1 Dataset Description

The final dataset which is obtained is derived from series of different dataset such as rainfall dataset, pesticide dataset, temperature dataset and yield dataset. This is performed as gathering and cleaning data to obtain a final dataset which is being used for Exploratory Data Analysis. The dataset is being taken from Kaggle website.

- **Pesticide Dataset** : <https://www.kaggle.com/code/tipatle/crop-yield-prediction-data-integration/input?select=pesticides.csv>
- **Rainfall Dataset** : <https://www.kaggle.com/code/tipatle/crop-yield-prediction-data-integration/input?select=rainfall.csv>
- **Temperature Dataset** : <https://www.kaggle.com/code/tipatle/crop-yield-prediction-data-integration/input?select=temp.csv>
- **Yield Dataset** : <https://www.kaggle.com/code/tipatle/crop-yield-prediction-data-integration/input?select=yield.csv>

From all this datasets the important column is identified and those columns from all the dataset is being merged to obtain the final dataset that is named as yield_df.

- ❖ **Final Dataset** : https://www.kaggle.com/code/tipatle/crop-yield-prediction-data-integration/input?select=yield_df.csv

The final dataset contains 28242 rows and 7 columns in total which meets the criteria of this Report.

2.2.1 Python Libraries

- **NumPy** – This library is being imported for supporting large mathematical calculations and it is a scientific computing in Python (AlmaBetter, n.d.).
- **Pandas** – This library is being imported for the purpose of merging, reshaping and aggregating data in short (AlmaBetter, n.d.).
- **Seaborn** – This library is being imported for the purpose of Data Visualization providing high-level interface of graphs (AlmaBetter, n.d.).
- **Matplotlib.pyplot** – This library is being imported for creating static, interactive and animated visualization (AlmaBetter, n.d.).
- **Sklearn** – This library is being imported for performing the tasks such as classification, regression, clustering and model selection (AlmaBetter, n.d.).
- **One-Hot-Encoder** – This library is being imported to convert the encoding variables to one-hot encoded format which can be used as input in Machine Learning Models (AlmaBetter, n.d.).
- **MinMaxScaler** – This library is being imported to measure the scale of numerical features from a specified range typically from 0 and 1 (AlmaBetter, n.d.).
- **train_test_split** – This library is being imported to divide the given dataset into two half which is one for training and the other one is for testing (AlmaBetter, n.d.).
- **r2score** – This library is being imported to evaluate the performance of regression models (AlmaBetter, n.d.).
- **DecisionTreeRegressor** – This library is being imported to create decision tree models specifically for regression based tasks (AlmaBetter, n.d.).

2.2.2 Import Libraries and Mount the Dataset

The initial stage of the program is that importing necessary libraries and mounting the dataset which is being to be used further. The libraries used are being explained in previous section.

```
Section One : Import Libraries and mounting the dataset.

1.Importing all necessary libraries needed for the rest part of the code. 2.Mounting the datasets to the google colab from google drive.

Double-click (or enter) to edit

import numpy as np # linear algebra
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

[ ] # Load the Drive helper and mount
from google.colab import drive

# This will prompt for authorization.
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[ ] %cd "/content/drive/My Drive/Colab"

/content/drive/My Drive/Colab
```

Figure 1 - Import Libraries and Mount the Dataset

2.3 Gathering and Cleaning the Data

The first section is to gather all the dataset to get a final data-frame in which we perform all visualisations and other analysis for the rest of the part. In this section, all the four factors are involved such as pesticide dataset, rainfall dataset, temperature dataset and yield dataset are all coordinated and all these datasets are cleaned to derive a final data-frame or dataset called `yield_df`.

2.3.1 Yield Dataset

In this dataset, the keyword used is `df_crop`. There are two main steps being involved that is renaming and dropping column for better understanding.

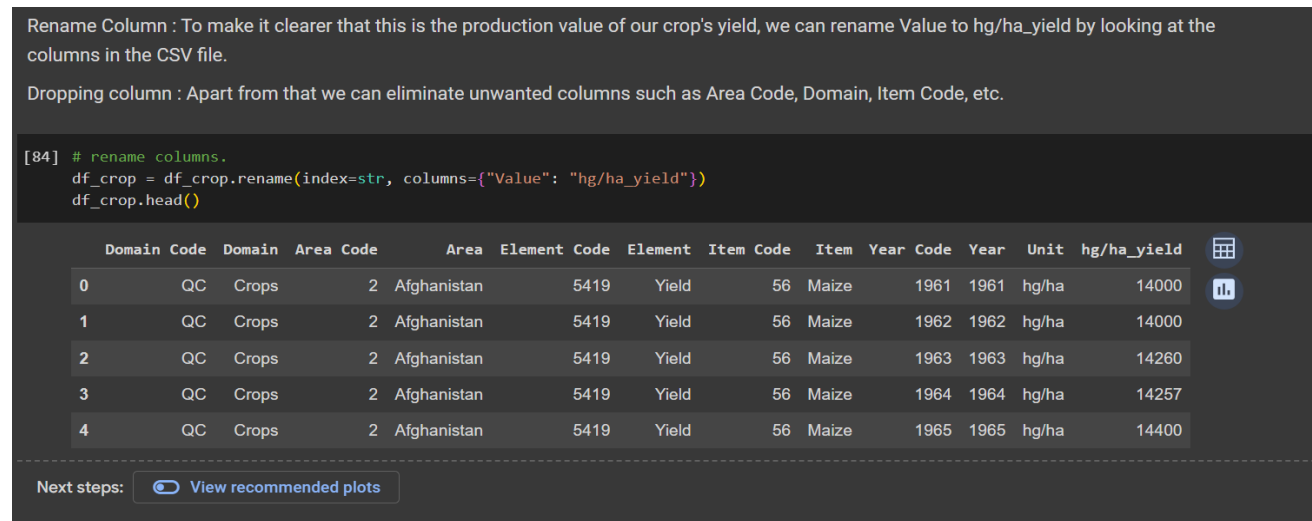


Figure 2 Rename column Value as `hg/ha_yield`

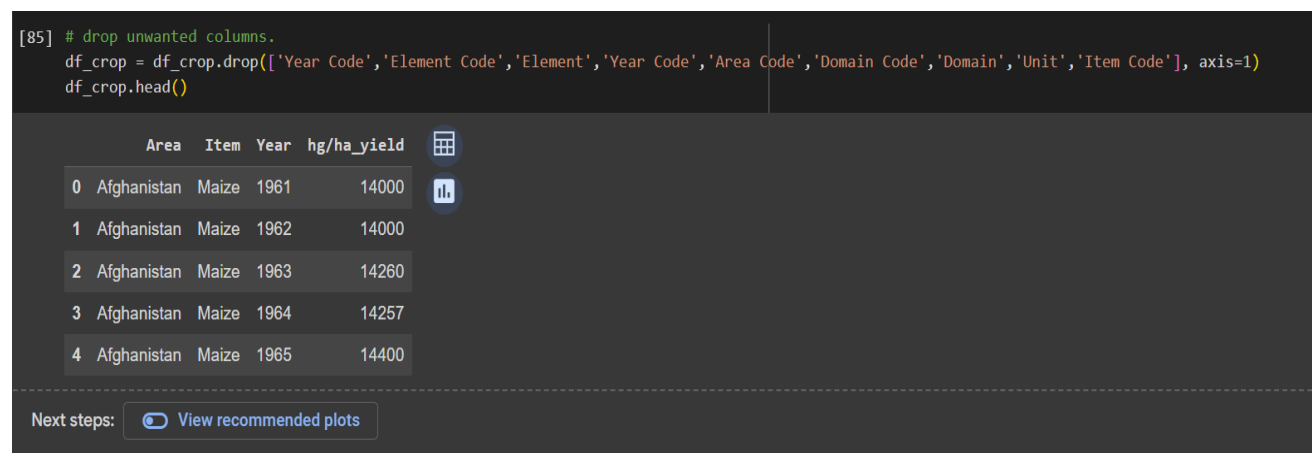


Figure 3 Dropping the unnecessary columns

2.3.2 Rainfall Dataset – Climate Factor

In this dataset, the keyword used is `df_climate`. Here, in `avg_rain_fall_mm_per_year` contains NaN values which can interrupt the progress. So we can either empty it or take mean of the column or drop it. In this case we are dropping it.

```
[90] # Convert NaN values to empty or other string

nan_value = float("NaN")
df_climate.replace("..", nan_value, inplace=True)

df_climate.dropna(subset = ["average_rain_fall_mm_per_year"], inplace=True)

# Convert the object dataframe to float value
df_climate["average_rain_fall_mm_per_year"] = pd.to_numeric(df_climate["average_rain_fall_mm_per_year"], downcast='float')

[92] df_climate.describe()
```

	Year	average_rain_fall_mm_per_year
count	5947.000000	5947.000000
mean	2001.365899	1124.743286
std	9.526335	786.257324
min	1985.000000	51.000000
25%	1993.000000	534.000000
50%	2001.000000	1010.000000

Figure 4 Convert NaN values to empty or other string

The data-frame now is being merged with previous cleaned dataset which is assigned to keyword as `yield_df`.

```
# merge yield dataframe with rain dataframe by year and area columns
yield_df = pd.merge(df_crop, df_climate, on=['Year','Area'])
yield_df.head()
```

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year
0	Afghanistan	Maize	1985	16652	327.0
1	Afghanistan	Potatoes	1985	140909	327.0
2	Afghanistan	Rice, paddy	1985	22482	327.0
3	Afghanistan	Wheat	1985	12277	327.0
4	Afghanistan	Maize	1986	16875	327.0

Figure 5 Merged data-frame

2.3.3 Pesticide Dataset – Another Influential Factor

In this dataset, the keyword used is `df_pest` . Rename and dropping the column to obtain final data-frame. Then merge the data-frame to `yield_df`.

Rename Column : To make it clearer that this is the production value of our pesticide , we can rename Value to pesticides_tonnes by looking at the columns in the CSV file.

Dropping column : Apart from that we can eliminate unwanted columns such as Element,Domain,Unit,Item etc.

```
[96] df_pest = df_pest.rename(index=str, columns={"Value": "pesticides_tonnes"})
      #rename
      df_pest = df_pest.drop(['Element','Domain','Unit','Item'], axis=1)
      #drop column
      df_pest.head()
```

	Area	Year	pesticides_tonnes
0	Albania	1990	121.0
1	Albania	1991	121.0
2	Albania	1992	121.0
3	Albania	1993	121.0
4	Albania	1994	201.0

Figure 6 Rename and Drop the column

2.3.4 Temperature Dataset

In this dataset, the keyword used is avg_temp. Here we are renaming and merging the dataframe again to obtain a complete data-frame.

```
[164] #rename the column name
avg_temp = avg_temp.rename(index=str, columns={"country": 'Area', "year": "Year"})
avg_temp.head()
```

	Year	Area	avg_temp
0	1849	Côte D'Ivoire	25.58
1	1850	Côte D'Ivoire	25.52
2	1851	Côte D'Ivoire	25.67
3	1852	Côte D'Ivoire	NaN
4	1853	Côte D'Ivoire	NaN

Next steps: [View recommended plots](#)

```
[165] yield_df = pd.merge(yield_df, avg_temp, on=['Area', 'Year'])
```

Figure 7 Rename and Merge the column

2.4 Final Dataframe – yield_df

The final dataset obtained is yield_df. It is constructed from merging the all the above mentioned datasets.

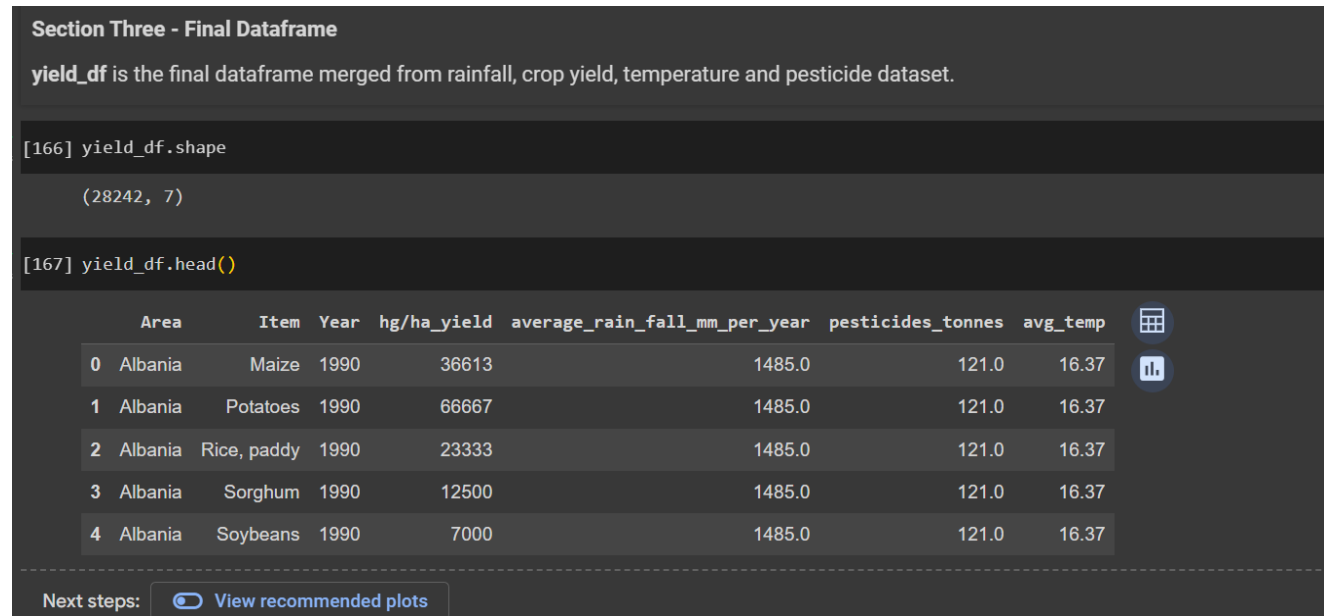


Figure 8 yield_df dataframe

2.5 Exploratory Data Analysis

- **Exploratory Data Analysis (EDA)** on the `yield_df` dataset is the process of studying and summarising the data's major properties in order to better understand its structure, patterns, and relationships. Here are some typical EDA steps for this dataset. (Mahadevan, 2022)
- **Data Overview:** Begin by loading the dataset and inspecting its fundamental attributes, such as the number of rows and columns, data types, and missing values. (Mahadevan, 2022)
- **Normalization** : Calculate descriptive statistics for numerical variables such as mean, median, standard deviation, minimum, and maximum values to better understand the data's central tendency and distribution (Mahadevan, 2022).
- **Data Visualisation:** Develop various forms of plots and charts to visualise the distribution of numerical variables (e.g., histograms, box plots), investigate correlations between variables (e.g., scatter plots, pair plots) (Mahadevan, 2022).
- **Correlation Analysis:** Use correlation coefficients between numerical variables to determine the strength and direction of linear relationships. Use a correlation matrix or heat-map to visually represent the correlations (Mahadevan, 2022).
- **Feature Engineering:** Create new features or modify existing ones to better capture patterns in data and improve model performance (Mahadevan, 2022).
- **Data pre-processing:** It involves handling missing values, encoding categorical variables, and scaling or normalising numerical features as needed to prepare the data for modelling (Mahadevan, 2022).

Overall, the purpose of EDA is to gather insights into the dataset that will help with later processes in the data analysis process, such as model selection, feature selection, and model evaluation.

```
[169] yield_df.groupby('Item').count()
```

	Area	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
Item						
Cassava	2045	2045	2045	2045	2045	2045
Maize	4121	4121	4121	4121	4121	4121
Plantains and others	556	556	556	556	556	556
Potatoes	4276	4276	4276	4276	4276	4276
Rice, paddy	3388	3388	3388	3388	3388	3388
Sorghum	3039	3039	3039	3039	3039	3039
Soybeans	3223	3223	3223	3223	3223	3223
Sweet potatoes	2890	2890	2890	2890	2890	2890
Wheat	3857	3857	3857	3857	3857	3857
Yams	847	847	847	847	847	847

Figure 9 yield_df dataframe

2.5.1 Data Visualisation for 'Top 10 areas with Largest Crop Yield'

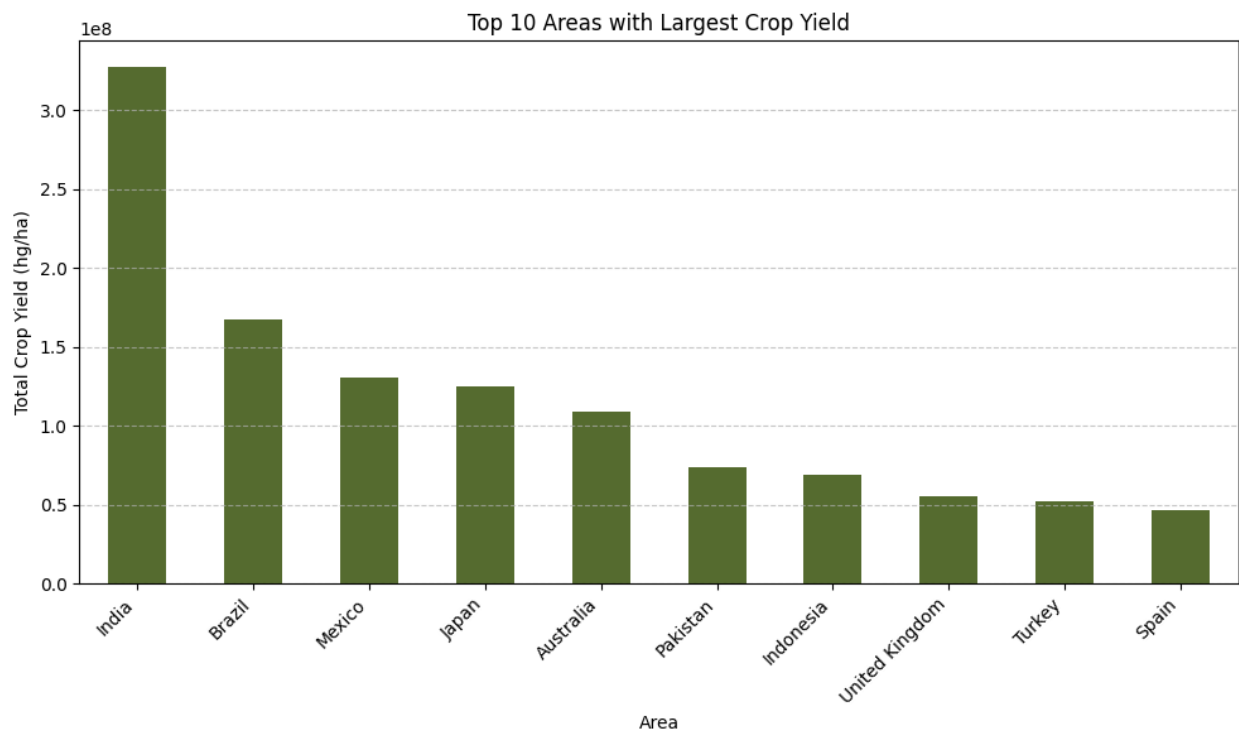


Figure 10 Top 10 Areas with Largest Crop Yield

This Visualization depicts that the range of crop yield all over the countries mentioned in final dataframe . This visualization helps in researching which has suitable land for crops to be implanted. The diagram itself specifies that India has much yield when compared to other countries which also explains that it has wide range of all types of crops that is being mentioned in the dataframe.

2.5.2 Top 10 Combinations of Crop Items, Areas by Crop Yield'

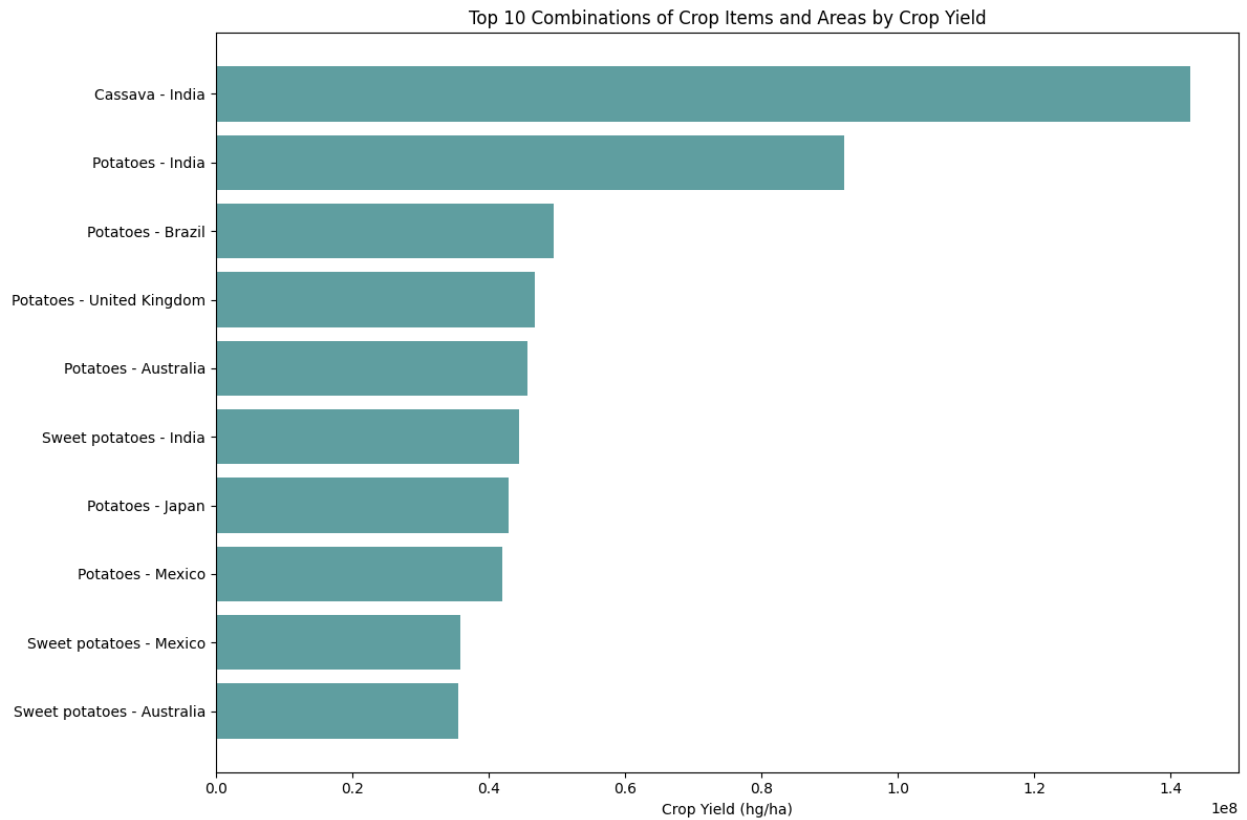


Figure 11 Top 10 Combinations of Crops, Areas by Crop Yield

The visualization is being visualized based on the crop yield of crops across the countries. This is to explain that it helps to detect which land is being suitable for type of crop. From this we can understand that which country has high yield of what crop.

2.5.3 Data Visualization for 'Crop Yield Over The Years'

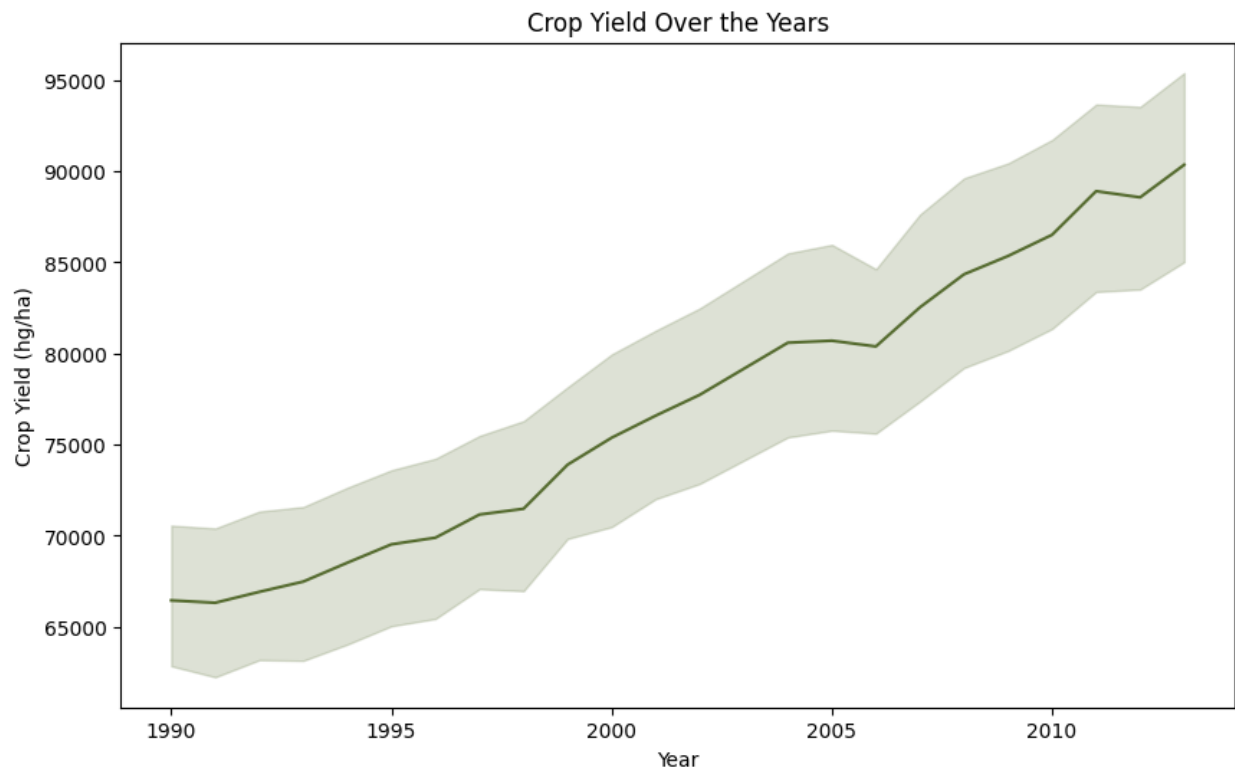


Figure 12 Crop Yield Over The Years

This visualization depicts that the crop yield has been increased or not all over these upcoming years. This is constructed because to prove that agriculture is in stable positioning of the crop standards.

2.5.4 Data Visualization for 'Crop Yield Vs Average Rainfall'

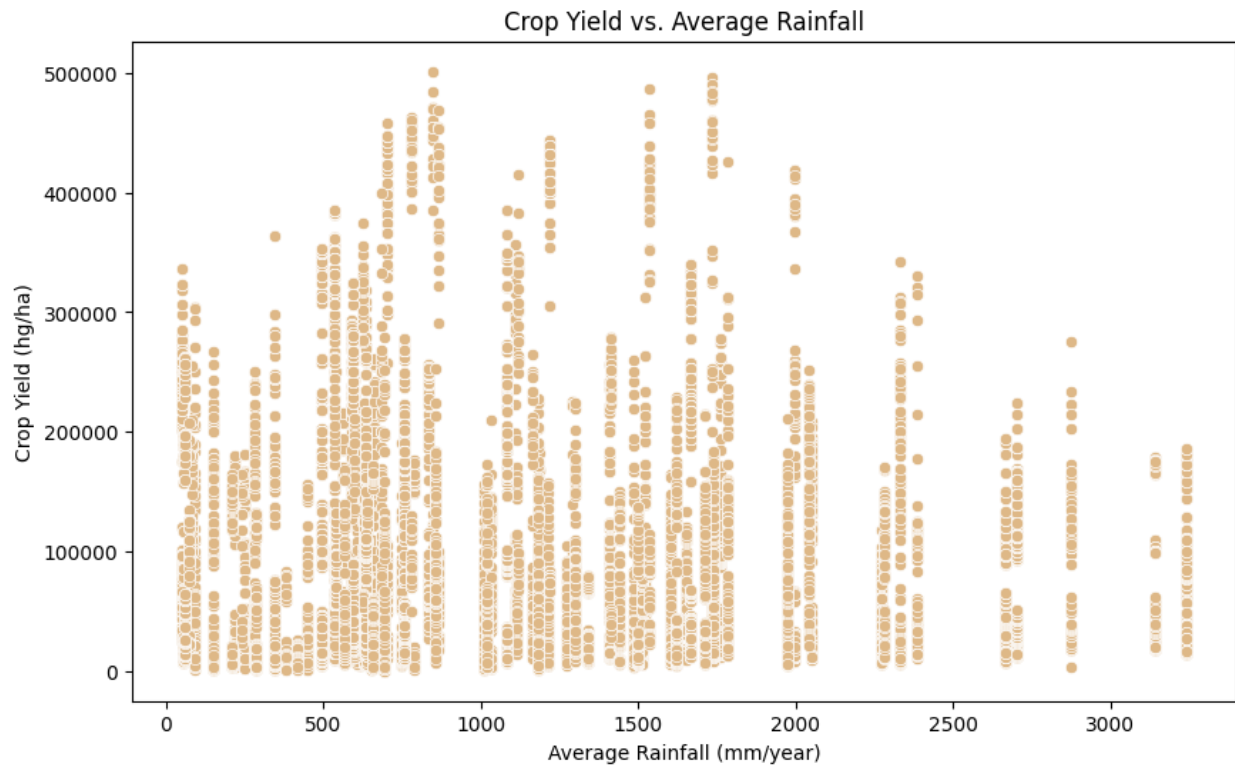


Figure 13 Crop Yield Vs Average Rainfall

This scatterplot depicts that there is huge crop yield when there is more rainfall. So, the average rainfall is being taken into account because it is one of the influential and basic factor for a crop yield. So when there is average rainfall there is more or average yield when compare to less rainfall.

2.5.6 Correlation Analysis

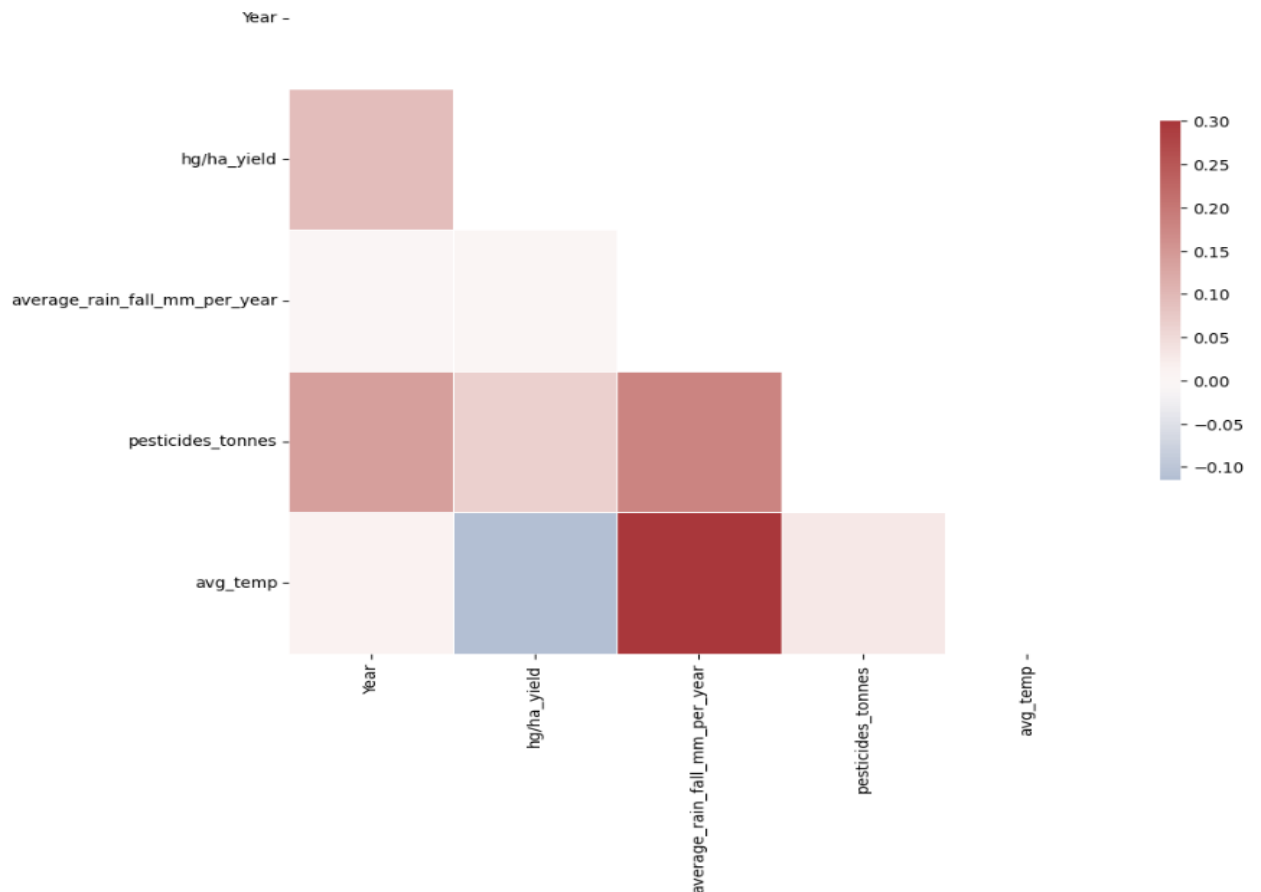


Figure 14 Correlation between the columns of the dataset

Note : From above correlation heat-map it is found that there is no correlation between any of the columns of the dataframe. It is so because we have limited number of columns so each column is associated with output column but no desired relationship with each of other defined columns. The variables in the dataset may not change enough to reveal meaningful associations. Identifying correlations can be difficult if the data is excessively uniform or lacking variation (Szabo, 2020).

2.5.7 Data Pre-Processing – One-Hot-Encoding

The data-frame contains two categorical columns, which are variables with label values rather than numeric values. In many cases, the number of possible values is limited to a specific set, such as the values for objects and datatypes. (Ali, 2023).

This means that category data must be transformed into numerical form. One hot encoding is the process of converting categorical information into a form that may be used by ML algorithms to improve prediction accuracy. One-Hot Encoding will be used to turn these two columns into a single numeric array (Ali, 2023).

The category value is the numerical value of an entry in the dataset. This encoding generates a binary column for each category and produces a matrix with the results (Ali, 2023).

```
[93] from sklearn.preprocessing import OneHotEncoder

[94] yield_df_onehot = pd.get_dummies(yield_df, columns=['Area','Item'], prefix = ['Country','Item'])
      X=yield_df_onehot.loc[:, yield_df_onehot.columns != 'hg/ha_yield']
      Y=yield_df['hg/ha_yield']
      X.head()
```

	Year	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argentina	Country_Armenia	Country_Australia	...	Item_C
0	1990	1485.0	121.0	16.37	True	False	False	False	False	False
1	1990	1485.0	121.0	16.37	True	False	False	False	False	False
2	1990	1485.0	121.0	16.37	True	False	False	False	False	False
3	1990	1485.0	121.0	16.37	True	False	False	False	False	False
4	1990	1485.0	121.0	16.37	True	False	False	False	False	False

5 rows x 115 columns

Figure 15 Encoding Categorical Variables

This phase of code creates new binary columns by performing one-hot encoding on the categorical columns "Area" and "Item" in Data-Frame yield_df. This is done for regression algorithm, separating predictor variables (X) and the desired variable (Y) and removing the 'hg/ha_yield' column from predictors, since we already know that hg/ha_yield is the output column. The encoded data is shown in the generated X Data-Frame (Ali, 2023).

```
[95] X = X.drop(['Year'], axis=1)

[96] X.head()
```

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argentina	Country_Armenia	Country_Australia	Country_Austria
0	1485.0	121.0	16.37	True	False	False	False	False	False	False
1	1485.0	121.0	16.37	True	False	False	False	False	False	False
2	1485.0	121.0	16.37	True	False	False	False	False	False	False
3	1485.0	121.0	16.37	True	False	False	False	False	False	False
4	1485.0	121.0	16.37	True	False	False	False	False	False	False

5 rows × 114 columns

Figure 16 Dropping Year Field from X Dataframe

In this phase, we are dropping the Year column we are dealing with the country and item that has to be yielded which accordingly nothing to do with the year. And the further analysis is also based on Country and Item and not Year so to keeping it simple we are dropping the respective column 'Year'.

2.5.8 Normalization

Scaling the numerical features to a similar range is commonly referred to as normalization in the context of the yield_df dataset. This keeps features with higher magnitudes from controlling the model training process and guarantees that every feature contributes equally to the analysis. When working with machine learning methods that are sensitive to feature sizes, like support vector machines (SVM) normalization is extremely crucial (Bhandari, 2020).

Scaling variables like agricultural yield, average rainfall, pesticide use, and average temperature to a common range, like [0, 1] or [-1, 1], may be necessary for normalization in the context of the yield_df dataset. Normalization ensures that algorithms can learn from the data efficiently and are not influenced by feature size. It also makes the dataset more suited for analysis and modelling (Bhandari, 2020).

The dataset shows significant variation in magnitudes, units, and ranges. Features with high magnitudes will be weighted more heavily in distance estimates than those with low magnitudes. To suppress this impact, we must bring all characteristics to the same magnitude level. This can be accomplished by scaling (Bhandari, 2020).

```
[97] from sklearn.preprocessing import MinMaxScaler
     scaler=MinMaxScaler()
     X=scaler.fit_transform(X)
```

X

```
array([[4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       ...,
       [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
        1.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
        0.00000000e+00, 1.00000000e+00, 0.00000000e+00]])
```

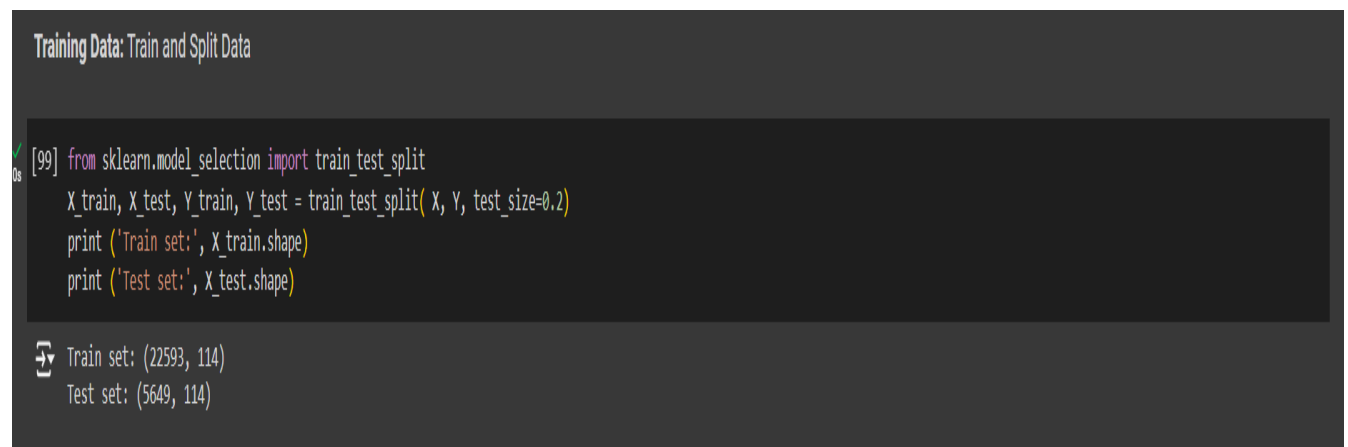
Figure 17 Normalization

This code scales the features in the 'X' variable using the scikit-learn Min-Max-Scaler. By transforming the features to fall inside a given range—typically between 0 and 1—MinMaxScaler ensures that all features have the same scale. We are doing this because we can compare with accuracy by using our SVM approach. It is utilized for several other distant calculative procedures in addition to the SVM technique. Each feature's minimum and maximum values are determined, and each feature is then scaled appropriately by deducting the minimum value and dividing by the range. For many machine learning algorithms, this normalization stage is crucial because it improves the algorithms' numerical stability, convergence speed, and performance(scikit-learn, 2019).

Chapter – 3 – Model Selection

3.1 Train and Split Data

The performance of machine learning algorithms when they are applied to prediction on non-training data is estimated using the train-test split approach. Here we are splitting in range of 80 percent training and 20 percent split. Even though the technique is easy to use and understand, there are scenarios in which it should not be used, such as when you have a tiny dataset, and others in which further configuration is necessary, like when the dataset is not balanced and the procedure is being used for classification (Brownlee, 2020).

A screenshot of a Jupyter Notebook interface. The top section is titled "Training Data: Train and Split Data". Below the title, there is a code cell with the following Python code:

```
[99] from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
print('Train set:', X_train.shape)
print('Test set:', X_test.shape)
```

 The code cell is followed by an output cell showing the results of the print statements:

```
Train set: (22593, 114)
Test set: (5649, 114)
```

```
Training Data: Train and Split Data

[99] from sklearn.model_selection import train_test_split
     X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
     print('Train set:', X_train.shape)
     print('Test set:', X_test.shape)

Train set: (22593, 114)
Test set: (5649, 114)
```

Figure 18 Train and Split Data

3.2 Model Selection

The system in this research uses machine learning approaches to forecast crop yield rate. Python is the programming language of choice since it is well-liked for implementing novel concepts in the field of machine learning. The gathered data set will be submitted for this project, and a crop yield estimate will be produced by using machine learning techniques such as ****decision tree regression, gradient boosting regression, and random forest regression.**** (Harika et al., 2877)

The information in the gathered data set determines the outcome. More accurate the parameter information in the gathered datasets, the better the outcomes will be(Harika et al., 2877)

```
[492] from sklearn.metrics import r2_score
def compare_models(model):
    model_value = model.__class__.__name__
    fit=model.fit(X_train,Y_train)
    pred=fit.predict(X_test)
    r2=r2_score(Y_test,pred)
    return([model_value,r2])

from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor

models = [
    GradientBoostingRegressor(n_estimators=200, max_depth=3, random_state=0),
    RandomForestRegressor(n_estimators=200, max_depth=3, random_state=0),
    DecisionTreeRegressor()
]

model_train=list(map(compare_models,models))
print("model_train, sep = "\n")

['GradientBoostingRegressor', 0.8996773945430375]
['RandomForestRegressor', 0.6964963480549694]
['DecisionTreeRegressor', 0.9644252268229165]
```

Figure 19 Model Evaluation and Accuracy rate

Note : Why Decision Tree Regression Tree has more accuracy?

The Decision Tree Regressor was chosen for the Crop yield_df dataset because of its capacity to handle the complicated, non-linear correlations found in agricultural data. Unlike linear regression models, decision trees can efficiently capture the interplay of different factors influencing crop output, such as weather, soil qualities, and agricultural techniques. This is critical in agriculture, where the link between input variables and crop production is frequently non-linear and affected by a variety of factors (Gupta et al., n.d.).

Furthermore, decision trees are highly interpretable, allowing stakeholders to better grasp the decision-making process and which factors are most relevant in crop yield prediction. Transparency is critical in agricultural decision-making, as farmers and policymakers require actionable facts to optimize crop output and resource allocation (Gupta et al., n.d.).

Furthermore, decision trees are resistant to outliers and can handle both numerical and categorical data, making them ideal for the diverse nature of agricultural information. Using the capabilities of decision tree regression, we can effectively model and estimate crop production based on a variety of agricultural parameters, resulting in more informed decision-making and resource management in agriculture (Gupta et al., n.d.).

Chapter – 4 – Key Visualizations based on Model Evaluation

4.1 Scatterplot for R² Score

Using agricultural data, this code segment trains a Decision Tree Regressor model (clf). The dataset's features are probably contained in the X_train variable, while crop yield (hg/ha_yield), which appears to be the target variable in this instance, is stored in Y_train (Amir, 2023).

Based on the features in the training set, the model predicts the crop yield (yield_predicted) after training. The actual yield values, which are added to the plot_df dataframe as yield_actual, are then compared to these predictions and saved in Y_train (Amir, 2023).

The data is then grouped by item type (Item) in the code, which probably corresponds to various crops. The R-squared score for each group is then determined. The percentage of variance in the dependent variable (crop yield) that can be predicted from the independent variables (features) is measured by the R-squared score (Amir, 2023).

For each crop type, the scores that are obtained show how well the model fits the data. Strong fits are indicated by high R-squared values near 1, which imply that most crop kinds' actual yield values and the model's forecasts closely match each other. Through an examination of the model's performance on diverse crops, this analysis sheds light on how well the model predicts crop yields in a range of agricultural settings (Amir, 2023).

```
[288] yield_df_onehot = yield_df_onehot.drop(['Year'], axis=1)

why we are dropping the year is because the R2 score is purely depend on the Items of crops so to get the score the numeric column is being
dropped.And only the categorical variables are taken into account.

[289] plot_df['Country']=countries
plot_df['Item']=items

[290] clf=DecisionTreeRegressor()
model=clf.fit(X_train,Y_train)

plot_df["yield_predicted"]= model.predict(X_train)
plot_df["yield_actual"]=pd.DataFrame(Y_train)["hg/ha_yield"].tolist()
test_group=plot_df.groupby("Item")
test_group.apply(lambda x: r2_score(x.yield_actual,x.yield_predicted))
```

Item	
Cassava	0.999333
Maize	0.999759
Plantains and others	0.999938
Potatoes	0.999626
Rice, paddy	0.999645
Sorghum	0.999865
Soybeans	0.999614
Sweet potatoes	0.999433
Wheat	0.999893
Yams	0.999734

0s completed at 2:57 AM

Figure 20 R2 Score for Crop Items

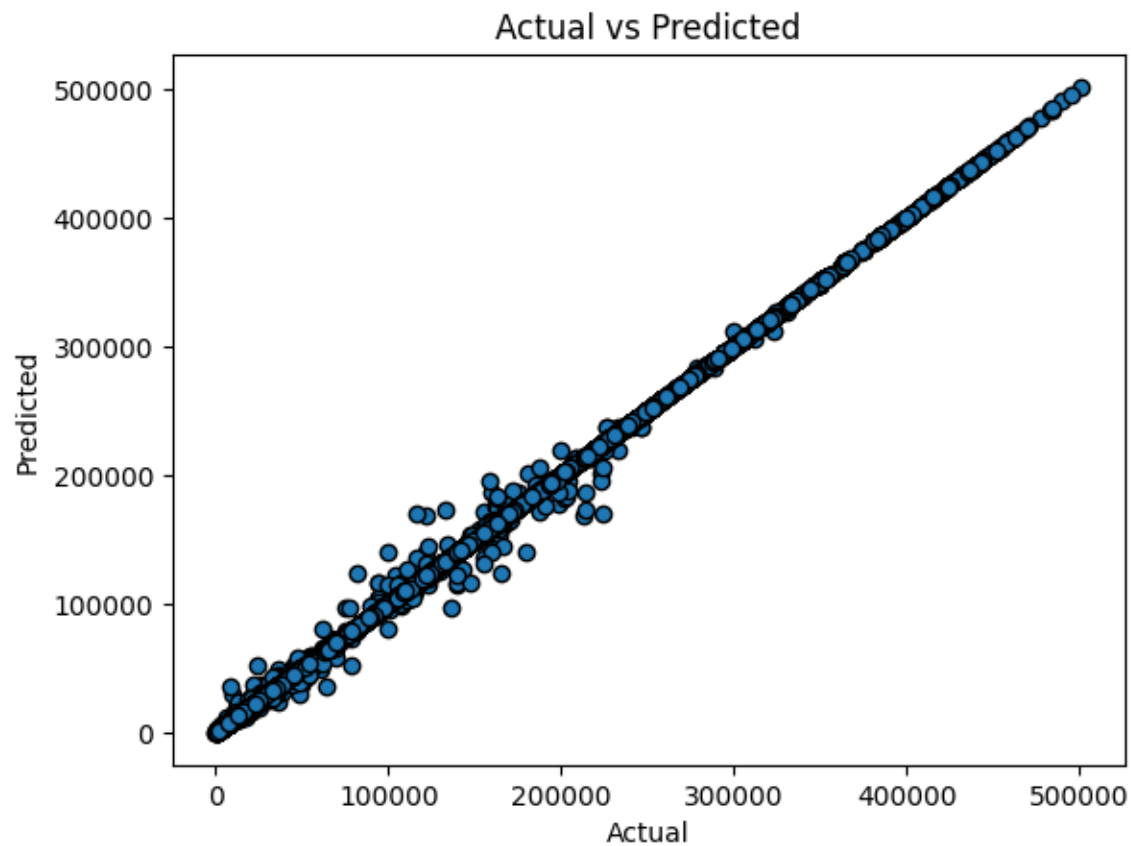


Figure 21 Scatterplot for Actual Vs Predicted

Note: The above figure displays the goodness of fit as a line based on the predictions. The excellent R Square score is evident. This indicates that we have discovered a well-fitting model to forecast the value of a given country's crop produce. The model's forecasts will likely be improved by including more factors, such as information on the climate, wind, pollution, the state of a particular nation's economy, and so forth (Frost, 2018).

4.2 Boxplot Visualisation on Crop Yield

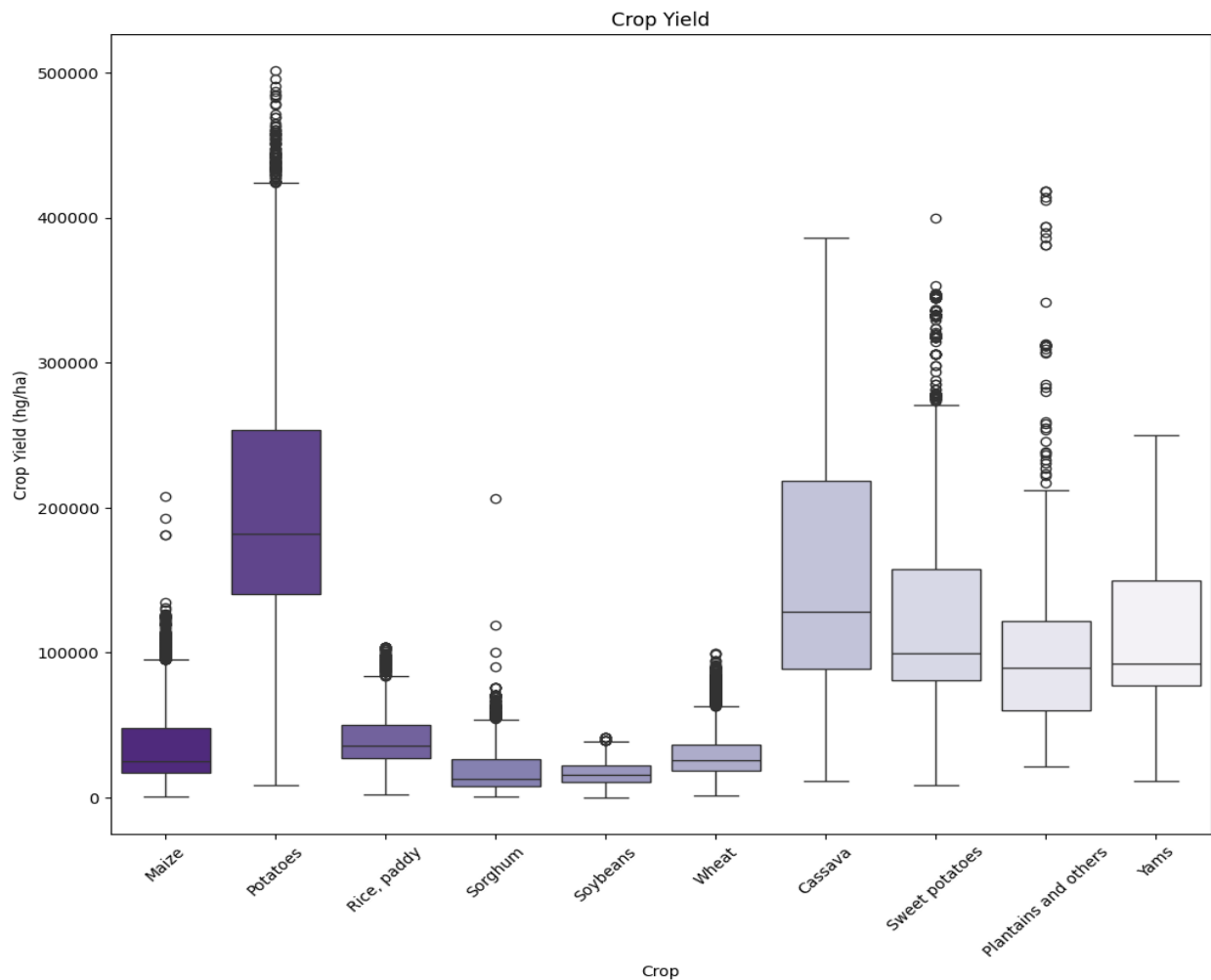


Figure 22 Boxplot Visualization on Crop Yield

The distribution of crop yield data is probably visualized by the `yield_df` boxplot function. The dataset's median, quartiles, and any outliers are shown in a boxplot. The range is represented by each box, with a line inside each box depicting the median. Usually 1.5 times the range extend to the lowest and maximum values within a certain range. Plotting of outliers outside of this range is done separately. The yield distribution's central tendency and spread are briefly summarized in this graphic, which makes it easier to spot any notable deviations or outliers in the data (www.ncl.ac.uk, n.d.).

4.3 Factors Influenced in Decision Making

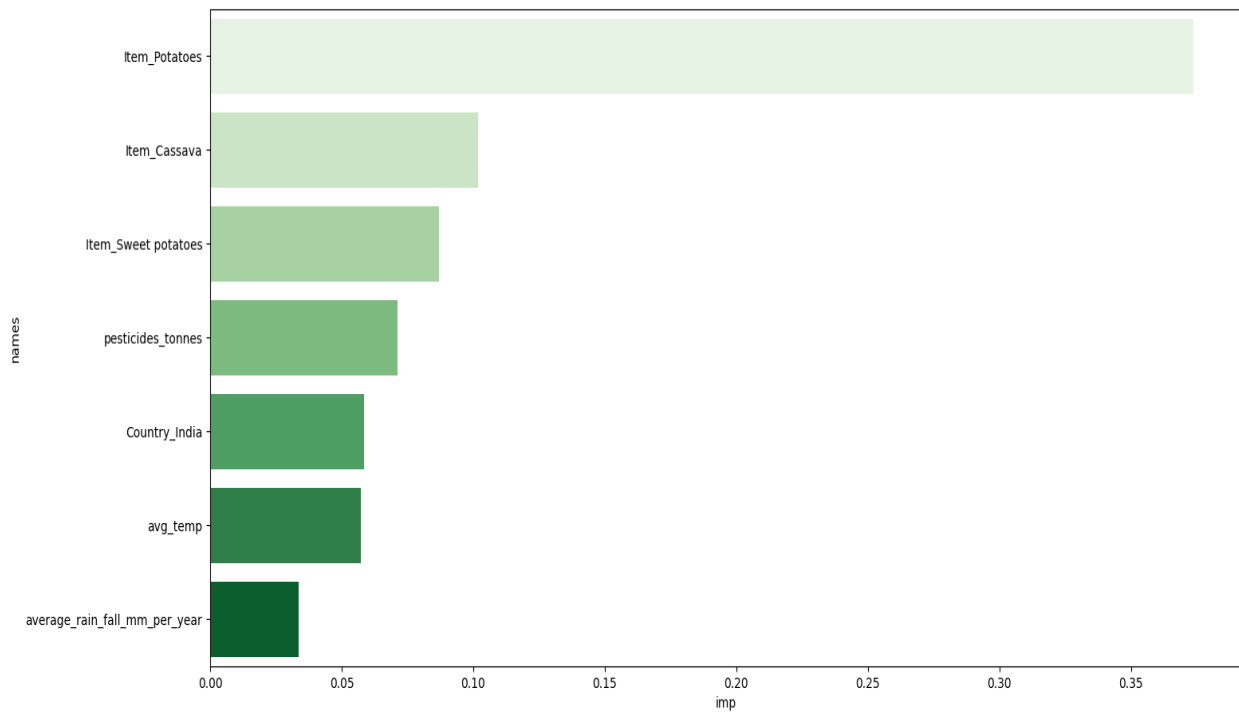


Figure 23 7-Factors Influenced In crop Yield Dataset

The most important variables influencing crop yields are graphically represented by this code, which facilitates the identification and prioritization of these variables in agricultural decision-making (Liliane and Charles, 2020).

Conclusion

The goal of this project is to help farmers become more financially stable and to address the rising number of farmer suicides. Farmers can use this project to choose the highest producing and most suited crops to plant on their land and to achieve the desired output. Machine learning tools were utilized to gather, analyze, and train relevant datasets. This project advances the agricultural industry. A list of crops and their productions according to climate conditions is one of the project's most significant and original contributions. This innovative, clever method of crop yield prediction works well for computing precise results. If this innovative method is applied to the cultivation of crops, choosing which of crops for farmers (Harika et al., 2877).

References

1. Malhi, G.S., Kaur, M. and Kaushik, P. (2021). Impact of Climate Change on Agriculture and Its Mitigation Strategies: A Review. *Sustainability*, 13(3), p.1318. doi: <https://doi.org/10.3390/su13031318>.
2. Talaviya, T., Shah, D., Patel, N., Yagnik, H. and Shah, M. (2020). Implementation of artificial intelligence in agriculture for optimisation of irrigation and application of pesticides and herbicides. *Artificial Intelligence in Agriculture*, [online] 4(2589-7217). doi: <https://doi.org/10.1016/j.aiia.2020.04.002>.
3. Ali, M. (2023). *Handling Categorical Data in Python Tutorial*. [online] [www.datacamp.com](https://www.datacamp.com/tutorial/categorical-data). Available at: <https://www.datacamp.com/tutorial/categorical-data>.
4. AlmaBetter. (n.d.). *Popular Python Libraries - NumPy, Pandas, Seaborn, Sklearn*. [online] Available at: <https://www.almabetter.com/bytes/tutorials/python/popular-python-libraries>.
5. Amir (2023). *Decision Trees with python code*. [online] Medium. Available at: <https://medium.com/@ab.jannatpour/decision-trees-with-python-code-380c020b088f>[Accessed 12 May 2024].
6. Bhandari, A. (2020). *Feature Scaling | Standardization Vs Normalization*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/>.
7. Brownlee, J. (2020). *Train-Test Split for Evaluating Machine Learning Algorithms*. [online] Machine Learning Mastery. Available at:

<https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/>.

8. Frost, J. (2018). *How To Interpret R-squared in Regression Analysis*. [online] Statistics By Jim. Available at: <https://statisticsbyjim.com/regression/interpret-r-squared-regression/>.
9. GitHub. (n.d.). *Crop-Yield-Production/crop_yield_prediction.ipynb at master · TheSaintIndiano/Crop-Yield-Production*. [online] Available at: https://github.com/TheSaintIndiano/Crop-Yield-Production/blob/master/crop_yield_prediction.ipynb [Accessed 12 May 2024].
10. Gupta, I., Ayalasomayajula, S., Shashidhara, Y., Kataria, A., Shashidhara, S., Kataria, K. and Undurti, A. (n.d.). *Innovations in Agricultural Forecasting: A Multivariate Regression Study on Global Crop Yield Prediction*. [online] Available at: <https://arxiv.org/pdf/2312.02254> [Accessed 12 May 2024].
11. Harika, B., Gayathri, N., Prasanthi, S., Sai Pramod, G. and Purna Teja, K. (2877). *CROP YIELD PREDICTION USING DECISION TREE, RANDOM FOREST AND GRADIENT BOOSTING REGRESSION TECHNIQUES*. [online] *International Research Journal of Modernization in Engineering Technology and Science @International Research Journal of Modernization in Engineering*, pp.2582–5208. Available at: https://www.irjmets.com/uploadedfiles/paper/issue_3_march_2023/34866/final/fin_irjmets1679985936.pdf
12. jovian.com. (n.d.). *Crop Yield Production Data Analysis - Notebook by Chalachew Muluken Liyew (chalachewsweet) | Jovian*. [online] Available at: <https://jovian.com/chalachewsweet/crop-yield-production-data-analysis> [Accessed 12 May 2024].

13. kaggle.com. (n.d.). *Crop Yield Prediction*. [online] Available at:
<https://www.kaggle.com/code/kushagranull/crop-yield-prediction>
14. Liliane, T.N. and Charles, M.S. (2020). *Factors Affecting Yield of Crops*.
[online] *www.intechopen.com*. IntechOpen. Available at:
<https://www.intechopen.com/chapters/70658>.
15. Mahadevan, M. (2022). *Step-by-Step Exploratory Data Analysis (EDA) using Python -*. [online] Analytics Vidhya. Available at:
<https://www.analyticsvidhya.com/blog/2022/07/step-by-step-exploratory-data-analysis-eda-using-python/>.
16. scikit-learn (2019). *sklearn.preprocessing.MinMaxScaler — scikit-learn 0.22.1 documentation*. [online] Scikit-learn.org. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html>.
17. Szabo, B. (2020). *How to Create a Seaborn Correlation Heatmap in Python?* [online] Medium. Available at:
<https://medium.com/@szabo.bibor/how-to-create-a-seaborn-correlation-heatmap-in-python-834c0686b88e>.
18. *www.ncl.ac.uk*. (n.d.). *Numeracy, Maths and Statistics - Academic Skills Kit*.
[online] Available at: <https://www.ncl.ac.uk/webtemplate/ask-assets/external/maths-resources/statistics/data-presentation/box-and-whisker-plots.html>.

Appendices

A1. Rainfall Dataset

The screenshot shows an Excel spreadsheet with the following data:

Area	Year	average_rain_fall_mm_per_year
Afghanistan	1985	327
Afghanistan	1986	327
Afghanistan	1987	327
Afghanistan	1989	327
Afghanistan	1990	327
Afghanistan	1991	327
Afghanistan	1992	327
Afghanistan	1993	327
Afghanistan	1994	327
Afghanistan	1995	327
Afghanistan	1996	327
Afghanistan	1997	327
Afghanistan	1998	327
Afghanistan	1999	327
Afghanistan	2000	327
Afghanistan	2001	327
Afghanistan	2002	327
Afghanistan	2004	327
Afghanistan	2005	327
Afghanistan	2006	327
Afghanistan	2007	327
Afghanistan	2008	327
Afghanistan	2009	327
Afghanistan	2010	327
Afghanistan	2011	327
Afghanistan	2012	327
Afghanistan	2013	327
Afghanistan	2014	327
Afghanistan	2015	327

A2. Pesticide Dataset

The screenshot shows an Excel spreadsheet with the following data:

Domain	Area	Element	Item	Year	Unit	Value
Pesticides	Albania	Use	Pesticides	1990	tonnes of	121
Pesticides	Albania	Use	Pesticides	1991	tonnes of	121
Pesticides	Albania	Use	Pesticides	1992	tonnes of	121
Pesticides	Albania	Use	Pesticides	1993	tonnes of	121
Pesticides	Albania	Use	Pesticides	1994	tonnes of	201
Pesticides	Albania	Use	Pesticides	1995	tonnes of	251
Pesticides	Albania	Use	Pesticides	1996	tonnes of	313.96
Pesticides	Albania	Use	Pesticides	1997	tonnes of	376.93
Pesticides	Albania	Use	Pesticides	1998	tonnes of	439.89
Pesticides	Albania	Use	Pesticides	1999	tonnes of	502.86
Pesticides	Albania	Use	Pesticides	2000	tonnes of	565.82
Pesticides	Albania	Use	Pesticides	2001	tonnes of	628.79
Pesticides	Albania	Use	Pesticides	2002	tonnes of	691.75
Pesticides	Albania	Use	Pesticides	2003	tonnes of	754.71
Pesticides	Albania	Use	Pesticides	2004	tonnes of	817.68
Pesticides	Albania	Use	Pesticides	2005	tonnes of	880.64
Pesticides	Albania	Use	Pesticides	2006	tonnes of	943.61
Pesticides	Albania	Use	Pesticides	2007	tonnes of	1006.57
Pesticides	Albania	Use	Pesticides	2008	tonnes of	1069.54
Pesticides	Albania	Use	Pesticides	2009	tonnes of	1132.5
Pesticides	Albania	Use	Pesticides	2010	tonnes of	1311.17
Pesticides	Albania	Use	Pesticides	2011	tonnes of	1302.63
Pesticides	Albania	Use	Pesticides	2012	tonnes of	766.25
Pesticides	Albania	Use	Pesticides	2013	tonnes of	982.32
Pesticides	Albania	Use	Pesticides	2014	tonnes of	1365.34
Pesticides	Albania	Use	Pesticides	2015	tonnes of	1300.84
Pesticides	Albania	Use	Pesticides	2016	tonnes of	1419.44
Pesticides	Algeria	Use	Pesticides	1990	tonnes of	1828.92
Pesticides	Algeria	Use	Pesticides	1991	tonnes of	2461.8

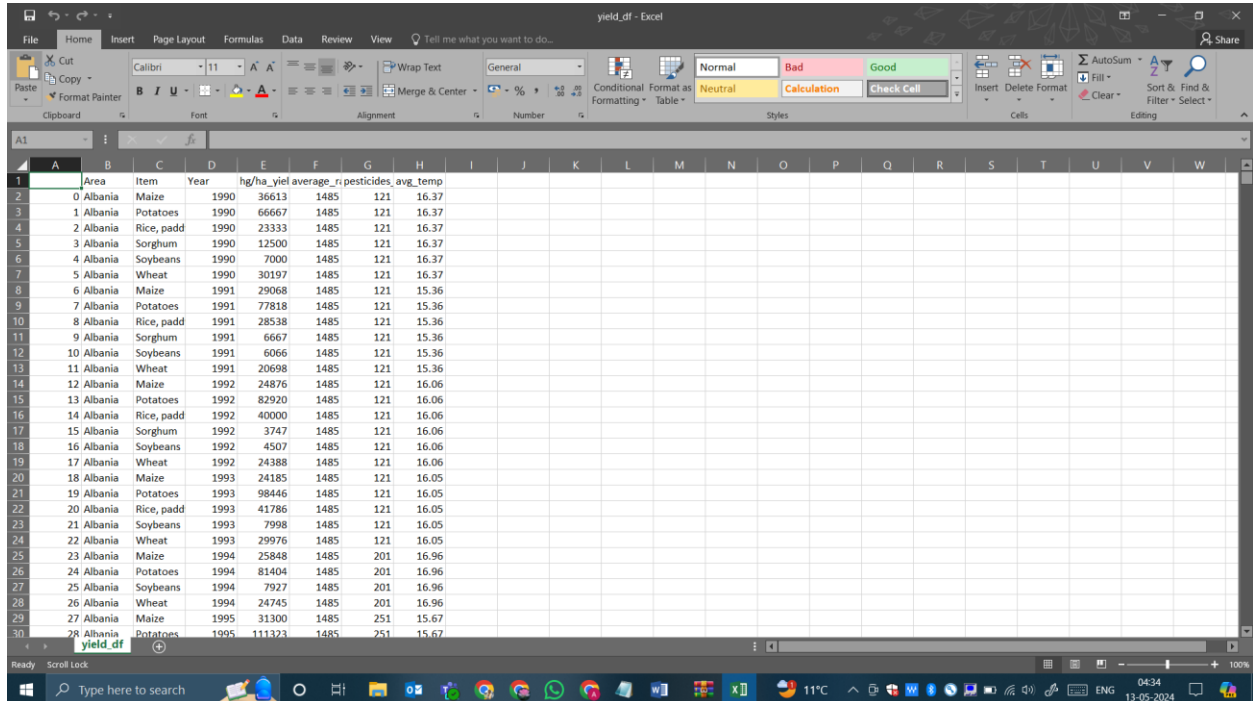
A3. Yield Dataset

Domain	Area Code	Area	Element	Item Code	Item	Year Code	Year	Unit	Value
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1961	14000
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1962	14000
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1963	14260
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1964	14257
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1965	14400
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1966	14400
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1967	14144
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1968	17064
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1969	17177
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1970	14757
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1971	13400
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1972	15652
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1973	16170
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1974	16170
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1975	16116
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1976	16598
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1977	15833
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1978	16183
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1979	16102
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1980	16711
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1981	16690
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1982	16658
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1983	16641
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1984	16612
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1985	16652
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1986	16875
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1987	17020
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1988	17034
QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1989	16963

A4. Temperature Dataset

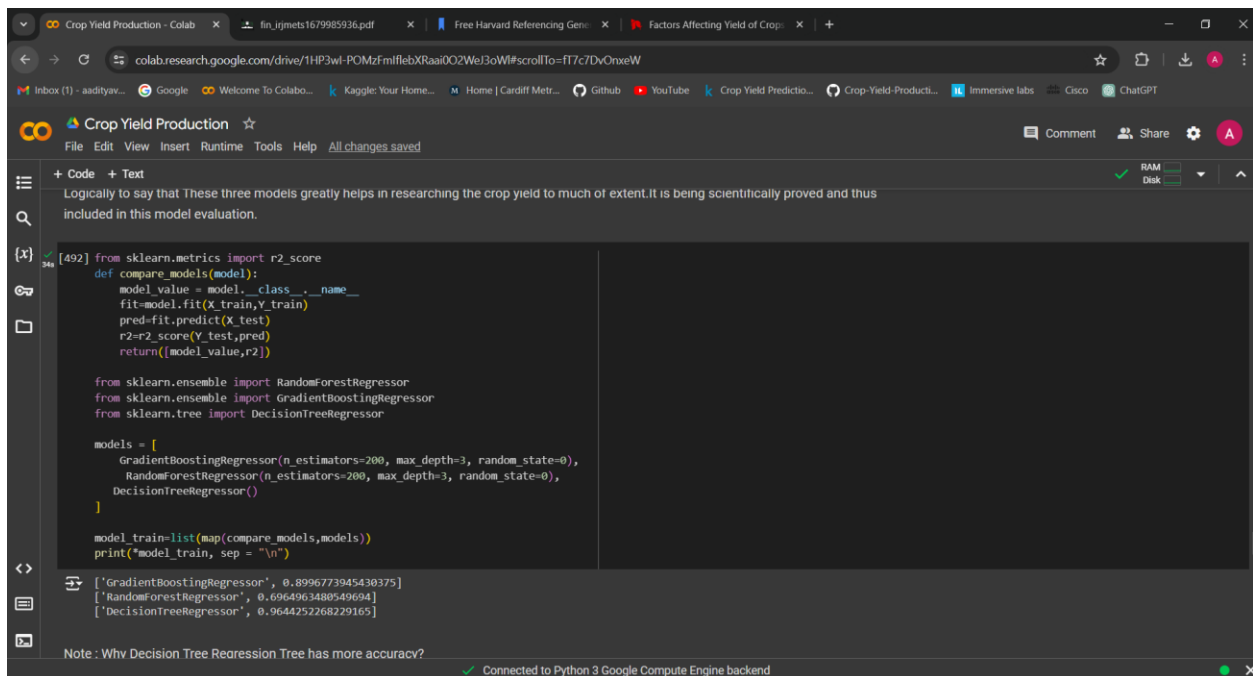
year	country	avg_temp
1849	Côte d'Ivoire	25.58
1850	Côte d'Ivoire	25.52
1851	Côte d'Ivoire	25.67
1852	Côte d'Ivoire	
1853	Côte d'Ivoire	
1854	Côte d'Ivoire	
1855	Côte d'Ivoire	
1856	Côte d'Ivoire	26.28
1857	Côte d'Ivoire	25.17
1858	Côte d'Ivoire	25.49
1859	Côte d'Ivoire	25.92
1860	Côte d'Ivoire	25.46
1861	Côte d'Ivoire	25.67
1862	Côte d'Ivoire	25.17
1863	Côte d'Ivoire	
1864	Côte d'Ivoire	
1865	Côte d'Ivoire	
1866	Côte d'Ivoire	
1867	Côte d'Ivoire	
1868	Côte d'Ivoire	
1869	Côte d'Ivoire	
1870	Côte d'Ivoire	
1871	Côte d'Ivoire	
1872	Côte d'Ivoire	
1873	Côte d'Ivoire	25.62
1874	Côte d'Ivoire	25.68
1875	Côte d'Ivoire	25.25
1876	Côte d'Ivoire	25.14
1877	Côte d'Ivoire	25.85

A5. Final Data-frame



Area	Item	Year	hg/ha_yiel	average_r	pesticides	avg_temp
0 Albania	Maize	1990	36613	1485	121	16.37
1 Albania	Potatoes	1990	66667	1485	121	16.37
2 Albania	Rice, padd	1990	23333	1485	121	16.37
3 Albania	Sorghum	1990	12500	1485	121	16.37
4 Albania	Soybeans	1990	7000	1485	121	16.37
5 Albania	Wheat	1990	30197	1485	121	16.37
6 Albania	Maize	1991	29068	1485	121	15.36
7 Albania	Potatoes	1991	77818	1485	121	15.36
8 Albania	Rice, padd	1991	28538	1485	121	15.36
9 Albania	Sorghum	1991	6667	1485	121	15.36
10 Albania	Soybeans	1991	6066	1485	121	15.36
11 Albania	Wheat	1991	20698	1485	121	15.36
12 Albania	Maize	1992	24876	1485	121	16.06
13 Albania	Potatoes	1992	82920	1485	121	16.06
14 Albania	Rice, padd	1992	40000	1485	121	16.06
15 Albania	Sorghum	1992	3747	1485	121	16.06
16 Albania	Soybeans	1992	4507	1485	121	16.06
17 Albania	Wheat	1992	24388	1485	121	16.06
18 Albania	Maize	1993	24185	1485	121	16.05
19 Albania	Potatoes	1993	98446	1485	121	16.05
20 Albania	Rice, padd	1993	41786	1485	121	16.05
21 Albania	Soybeans	1993	7998	1485	121	16.05
22 Albania	Wheat	1993	29976	1485	121	16.05
23 Albania	Maize	1994	25848	1485	201	16.96
24 Albania	Potatoes	1994	81404	1485	201	16.96
25 Albania	Soybeans	1994	7927	1485	201	16.96
26 Albania	Wheat	1994	24745	1485	201	16.96
27 Albania	Maize	1995	31300	1485	251	15.67
28 Albania	Potatoes	1995	111323	1485	251	15.67

A6. Google Colaboratory



```
[492] from sklearn.metrics import r2_score
def compare_models(model):
    model_value = model.__class__.__name__
    fit=model.fit(X_train,Y_train)
    pred=fit.predict(X_test)
    r2=r2_score(Y_test,pred)
    return([model_value,r2])

from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor

models = [
    GradientBoostingRegressor(n_estimators=200, max_depth=3, random_state=0),
    RandomForestRegressor(n_estimators=200, max_depth=3, random_state=0),
    DecisionTreeRegressor()
]

model_train=list(map(compare_models,models))
print("model_train, sep = '\n'")

['GradientBoostingRegressor', 0.8996773945430375]
['RandomforestRegressor', 0.6964963480549694]
['DecisionTreeRegressor', 0.9644252268229165]
```

Note : Why Decision Tree Regression Tree has more accuracy?