Crop Yield Prediction

A project report submitted in partial fulfillment of the requirements for the award of the degree of

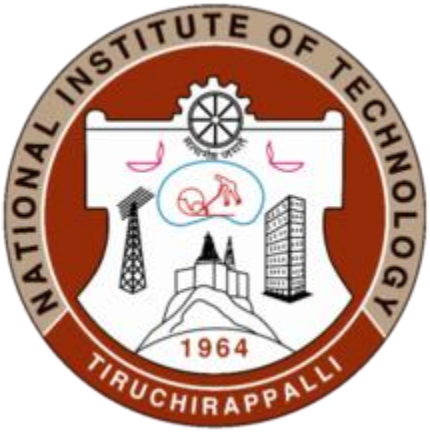
Master of Computer Applications

in

Computer Applications

By

Aman Singh (205121015)



DEPARTMENT OF COMPUTER APPLICATIONS

NATIONAL INSTITUTE OF TECHNOLOGY,

TIRUCHIRAPPALLI 620015

DECEMBER 2023

BONAFIDE CERTIFICATE

This is to certify that the project **"Crop Yield Prediction"** is a project work successfully done by

Aman Singh (205121015)

in partial fulfillment of the requirements for the award of the degree of Master of Computer Applications from the National Institute of Technology, Tiruchirappalli, during the academic year 2022-2023 (5th Semester – CA749 Mini Project Work).

Dr. S. Nickolas
Project Guide

Prof. Dr. Michael Arock
Head of the Department

Project viva-voce held on …….

Acknowledgment

Every project, big or small, is successful largely due to the effort of a number of wonderful people who have always given their valuable advice or lent a helping hand. I sincerely appreciate the inspiration, support, and guidance of all those people who have been instrumental in making this project successful.

We express our deep sense of gratitude to **Dr. G. Aghila**, Director, National Institute of Technology, Tiruchirappalli for giving us an opportunity to do this project.

I am grateful to **Dr. Michael Arock**, Professor, and Head of the Department of Computer Applications, National Institute of Technology, Tiruchirappalli for providing the infrastructure and facilities to carry out the project.

I express my gratitude to my Project Guide **Dr. S. Nickolas**, Professor, Department of Computer Applications, National Institute of Technology, Tiruchirappalli for his support and for arranging the project in a good schedule, and who assisted me in completing the project. I would like to thank him for duly evaluating my progress and evaluating me.

I express my sincere and heartfelt gratitude to **Project Evaluation Committee**, Department of Computer Applications, National Institute of Technology, Tiruchirappalli. I am sincerely thankful for its constant support, care, guidance, and regular interaction throughout my project.

I express my sincere thanks to all the faculty members, and scholars of NIT Trichy for their critical advice and guidance to develop this project directly or indirectly.

Abstract

Crop yield prediction is vital in modern agriculture, aiding farmers in decision-making, resource optimization, and overall farm management. This study explores the application of machine learning models for accurate and timely crop yield prediction. The project leverages diverse datasets, including soil characteristics, temperature, and rainfall, to develop robust predictive models.

The dataset is preprocessed to handle missing values, normalize data, and address outliers. Feature engineering is applied to extract relevant information from soil profiles, temperature records, and rainfall measurements. The dataset is split into training and testing sets for developing and evaluating machine-learning models.

The proposed solution will be based on the dataset,

https://www.kaggle.com/datasets/noorsaeed/crop-yield-prediction-dataset

Several machine-learning algorithms, including Linear Regression, Decision Tree Regressor, Random Forest Regressor, KNN Regressor, Gradient Boosting Regressor, and Bagging Regressor, are implemented to analyze the relationships between soil properties, temperature variations, rainfall patterns, and crop yields. The study investigates the impact of each feature on prediction accuracy and explores potential interactions among them.

The accuracy measure of a crop yield prediction project serves as a critical gauge of the model's performance in estimating crop outcomes based on soil, temperature, and rainfall data. Two key metrics, Root Mean Squared Error (RMSE) and R-squared (R2) score are commonly employed to evaluate the models' accuracy and goodness of fit.

Table of Contents

1. Introduction

Agriculture, a cornerstone of global food security, faces the continuous challenge of maximizing crop yields sustainably and efficiently. In this context, leveraging advanced technologies, such as machine learning, has become increasingly vital for enhancing precision agriculture. **"Crop Yield Prediction"** focuses on predicting crop yields by harnessing the power of machine learning algorithms, specifically emphasizing the influence of three pivotal factors: pesticides, temperature variations, and rainfall patterns.

Integrating artificial intelligence and machine learning into crop yield prediction models has significantly enhanced our ability to capture the complexity of agricultural systems. By leveraging historical and real-time data, these models can discern patterns, identify correlations, and make predictions that are precise and adaptable to changing environmental conditions.

Using advanced crop prediction models is essential for ensuring we can grow enough food sustainably, reduce farming risks, and ensure everyone has enough to eat worldwide. This report will explore these models in detail, showing how they could change how we farm as our world keeps changing.

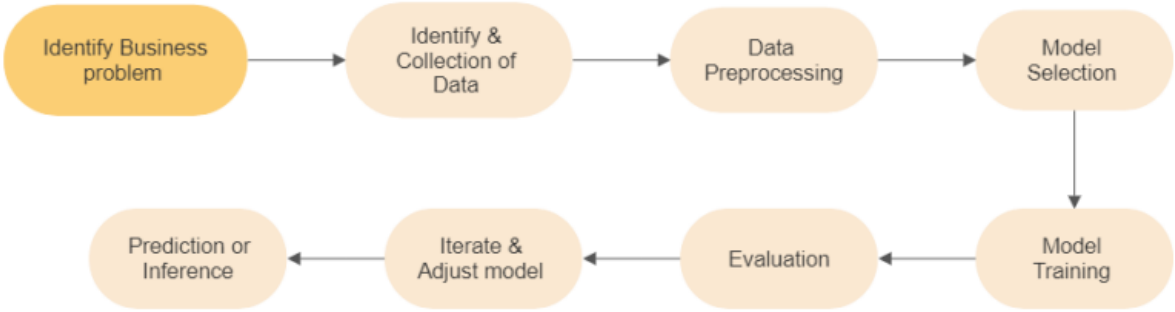
2. Problem Statement

Despite significant advancements in agricultural technology, the accurate prediction of crop yields remains a complex challenge. Traditional methods often fail to provide precise forecasts due to the intricate interplay of various factors influencing crop growth. Reliable crop yield predictions are crucial for sustainable agriculture, risk management, and global food security.

The problem is to develop a predictive model for crop yield estimation using historical data on temperature, rainfall, and pesticide usage aggregated at the country and year level. This model aims to provide accurate forecasts of crop yields, which are critical for food security, agricultural planning, and economic stability.

The following stages were involved in finding solution:

- Collecting the dataset from the online resource
- Preprocessing the dataset.
- Build models using the preprocessed dataset and algorithms identified
- Testing the models.
- Evaluating the performance of the models by considering performance metrics.



Stages in Machine Learning

3. TECHNOLOGY USED

Software

To implement experimentation, I have used the following technologies. A brief description of used technologies, along with versions is represented below.

- Python - Python is an interpreted, high-level, and object-oriented programming language. It is an open source.
- Jupyter Notebook - It is a web-based interactive computing platform. It helps in developing, documenting, and executing code.
- Pandas - It is a free, open-source Python library. It is mainly used for data analysis. It helps to perform various data manipulation operations.
- Numpy - It is a Python library that is used for scientific computing in Python. It is used to perform wide mathematical operations on data.
- Matplotlib - It is a Python package used to create static, animated, and interactive visualizations.
- Seaborn - It is a Python library that is used for data visualization. It is based on the matplotlib library. It helps make statistical graphics using Python.
- Plotly - Plotly's Python graphing library makes interactive, publication-quality graphs.
- Scikit-learn - It is a Python library that is used for machine learning. It consists of many machine learning algorithms, best suited for predictive analysis.

Hardware

- Hard Disk 1 TB
- Processor Intel i3 11th gen
- RAM 4 GB

4. METHODOLOGY

The complete life cycle of this project is divided into five challenges. Each of them will be explained in detail in the following step:

1. Data Collection

It contains more than 28000 records of different kinds of crops grown in different countries worldwide. It contains essential features which will to used to predict the yield of different crops in different countries.

It contains the following seven fields:

- area: Country.
- items: Name of crops.
- year: Year in which the crop is produced.
- hg/ha_yield: Crop produced per unit area(hectogram/hectare).
- average_rainfall_mm_per_year: Average rainfall of that country.
- pesticides_tonnes: Amount of pesticides used.
- avg_temp: Average Temperature of that country.

Import Libraries

In [1]:
```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import plotly.express as px
```

Reading dataset

In [2]:
```
df=pd.read_csv("Crop/yield_df.csv")
df
```

Out[2]:

	Unnamed: 0	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	0	Albania	Maize	1990	36613	1485.0	121.00	16.37
1	1	Albania	Potatoes	1990	66667	1485.0	121.00	16.37
2	2	Albania	Rice, paddy	1990	23333	1485.0	121.00	16.37
3	3	Albania	Sorghum	1990	12500	1485.0	121.00	16.37
4	4	Albania	Soybeans	1990	7000	1485.0	121.00	16.37
...	...	...	...	...	...	...	...	...
28237	28237	Zimbabwe	Rice, paddy	2013	22581	657.0	2550.07	19.76
28238	28238	Zimbabwe	Sorghum	2013	3066	657.0	2550.07	19.76
28239	28239	Zimbabwe	Soybeans	2013	13142	657.0	2550.07	19.76
28240	28240	Zimbabwe	Sweet potatoes	2013	22222	657.0	2550.07	19.76
28241	28241	Zimbabwe	Wheat	2013	22888	657.0	2550.07	19.76

28242 rows × 8 columns

Dataset Information

```
In [4]: 1 df.describe()
```

Out[4]:

	Unnamed: 0	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
count	28242.000000	28242.000000	28242.000000	28242.00000	28242.000000	28242.000000
mean	14120.500000	2001.544296	77053.332094	1149.05598	37076.909344	20.542627
std	8152.907488	7.051905	84956.612897	709.81215	59958.784665	6.312051
min	0.000000	1990.000000	50.000000	51.00000	0.040000	1.300000
25%	7060.250000	1995.000000	19919.250000	593.00000	1702.000000	16.702500
50%	14120.500000	2001.000000	38295.000000	1083.00000	17529.440000	21.510000
75%	21180.750000	2008.000000	104676.750000	1668.00000	48687.880000	26.000000
max	28241.000000	2013.000000	501412.000000	3240.00000	367778.000000	30.650000

```
In [5]: 1 df.describe(include='object')
```

Out[5]:

	Area	Item
count	28242	28242
unique	101	10
top	India	Potatoes
freq	4048	4276

2. Data Preprocessing

Any processing done on raw data to get it ready for another data processing step is referred to as data preprocessing, which is a part of data preparation. It has always been a crucial first stage in the data mining procedure. Data preparation techniques have been modified more recently to train AI and machine learning models and to make inferences against them.

Data preprocessing transforms the data into a format that may be used for data mining, machine learning, and other data science operations more quickly and efficiently. To guarantee reliable findings, the techniques are typically applied early in the machine learning and artificial intelligence development pipeline.

There are several different tools and methods used for preprocessing data, including sampling, transformation, denoising, imputation, normalization, feature extraction.

These tools and methods can be used on a variety of data sources, including data stored in files or databases and streaming data.

In [6]:
```
df.drop("Unnamed: 0", axis=1,inplace=True)
df
```

Out[6]:

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	Albania	Maize	1990	36613	1485.0	121.00	16.37
1	Albania	Potatoes	1990	66667	1485.0	121.00	16.37
2	Albania	Rice, paddy	1990	23333	1485.0	121.00	16.37
3	Albania	Sorghum	1990	12500	1485.0	121.00	16.37
4	Albania	Soybeans	1990	7000	1485.0	121.00	16.37
...	...	...	...	...	...	...	...
28237	Zimbabwe	Rice, paddy	2013	22581	657.0	2550.07	19.76
28238	Zimbabwe	Sorghum	2013	3066	657.0	2550.07	19.76
28239	Zimbabwe	Soybeans	2013	13142	657.0	2550.07	19.76
28240	Zimbabwe	Sweet potatoes	2013	22222	657.0	2550.07	19.76
28241	Zimbabwe	Wheat	2013	22888	657.0	2550.07	19.76

28242 rows × 7 columns

In [7]:
```
# remove countries with less than 100 record
country_counts =df['Area'].value_counts()
countries_to_drop = country_counts[country_counts < 100].index.tolist()
df_filtered = df[~df['Area'].isin(countries_to_drop)]
df = df_filtered.reset_index(drop=True)
df
```

Out[7]:

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	Algeria	Maize	1990	16500	89.0	1828.92	17.48
1	Algeria	Potatoes	1990	78936	89.0	1828.92	17.48
2	Algeria	Rice, paddy	1990	28000	89.0	1828.92	17.48
3	Algeria	Sorghum	1990	16571	89.0	1828.92	17.48
4	Algeria	Wheat	1990	6315	89.0	1828.92	17.48
...	...	...	...	...	...	...	...
26292	Zimbabwe	Rice, paddy	2013	22581	657.0	2550.07	19.76
26293	Zimbabwe	Sorghum	2013	3066	657.0	2550.07	19.76
26294	Zimbabwe	Soybeans	2013	13142	657.0	2550.07	19.76
26295	Zimbabwe	Sweet potatoes	2013	22222	657.0	2550.07	19.76
26296	Zimbabwe	Wheat	2013	22888	657.0	2550.07	19.76

26297 rows × 7 columns

In [8]:
```
## showing the count of Nans
print(df.isnull().sum())
```

```
Area                            0
Item                            0
Year                            0
hg/ha_yield                     0
average_rain_fall_mm_per_year   0
pesticides_tonnes               0
avg_temp                        0
dtype: int64
```

In [9]:
```
df.describe()
```

Out[9]:

	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
count	26297.000000	26297.000000	26297.000000	26297.000000	26297.000000
mean	2001.501464	75442.329277	1165.491577	39492.486263	20.990297
std	7.052085	82727.001265	712.540421	61396.180191	5.989020
min	1990.000000	50.000000	51.000000	0.040000	1.610000
25%	1995.000000	19730.000000	593.000000	2513.000000	17.280000
50%	2001.000000	37322.000000	1083.000000	24304.000000	21.960000
75%	2008.000000	101741.000000	1668.000000	51741.990000	26.090000
max	2013.000000	468991.000000	3240.000000	367778.000000	30.420000

In [10]:
```
df.describe(include='object')
```

Out[10]:

	Area	Item
count	26297	26297
unique	71	10
top	India	Potatoes
freq	4048	3724

3. Feature Extraction

Feature extraction is a process in which relevant information or features are selected, extracted, or transformed from raw data to be used in machine learning models. The goal is to reduce the dimensionality of the data, capture the most important information, and improve the model's performance. The techniques for feature selection in machine learning can be broadly classified into the following categories:

Supervised Techniques: These techniques can be used for labeled data and to identify the relevant features for increasing the efficiency of supervised models like regression and classification. For Example- linear regression, decision tree, random forest, etc.

Unsupervised Techniques: These techniques can be used for unlabeled data. For Example- K-Means Clustering, Hierarchical Clustering, Principal Component Analysis, etc.

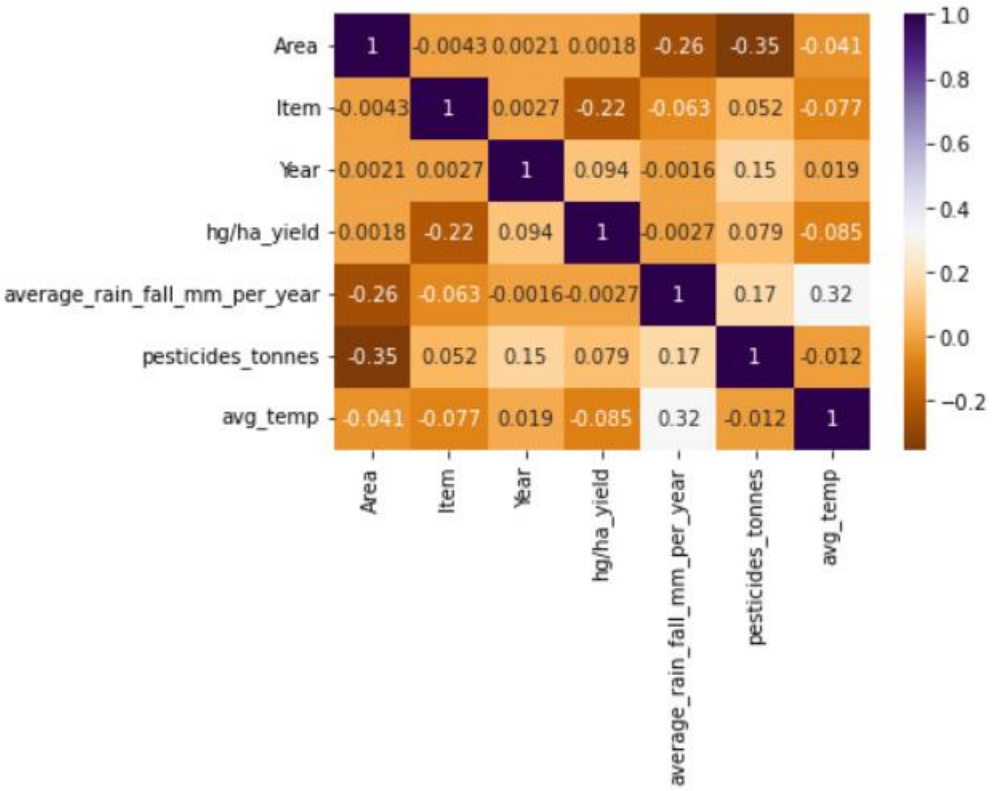
Correlation Coefficient

The correlation coefficient is a statistical measure that quantifies the degree to which two variables are linearly related. It assesses the strength and direction of a linear relationship between two continuous variables. The correlation coefficient is denoted by the symbol "r," and its values range between -1 and 1. A positive correlation indicates that as one variable increases, the other variable tends to increase as well. A negative correlation indicates that the other variable tends to decrease as one variable increases.

```
In [11]:
datacorr=df.copy()
```

```
In [12]:
from sklearn.preprocessing import LabelEncoder
categorical_columns = datacorr.select_dtypes(include=['object']).columns.tolist()
label_encoder = LabelEncoder()
for column in categorical_columns:
    datacorr[column] = label_encoder.fit_transform(datacorr[column])

sns.heatmap(datacorr.corr() , annot= True , cmap='PuOr')
```

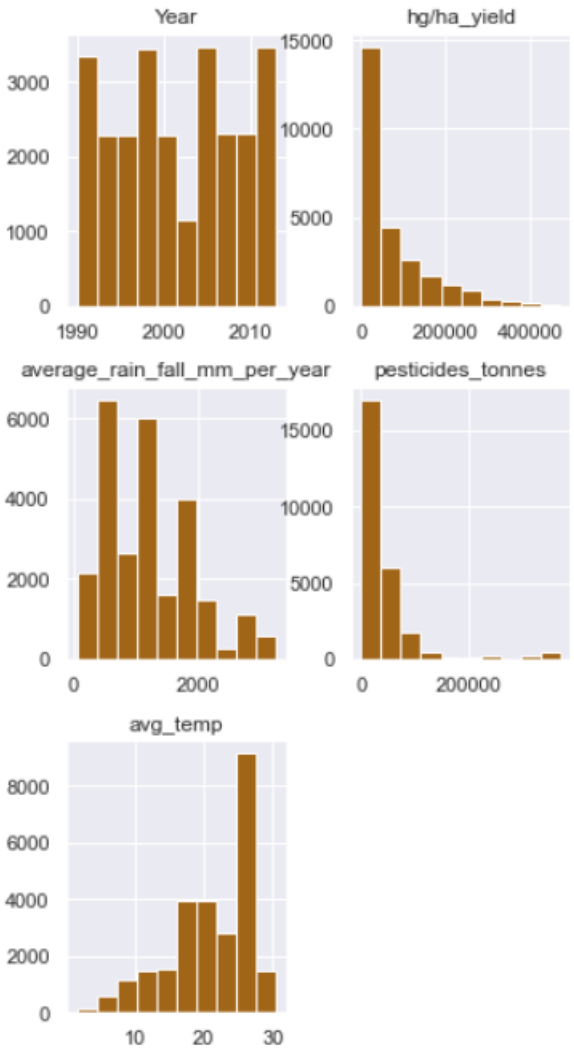
```
Out[12]: <AxesSubplot:>
```



Histogram

A histogram is a graphical representation of the distribution of a dataset. It is a way to visualize the underlying frequency distribution of a set of continuous or discrete data. Histograms provide insights into the shape, center, and spread of the data. They are commonly used in data analysis and statistics. Histograms are useful for understanding the central tendency and variability of a dataset, identifying outliers, and making decisions about data preprocessing or modeling approaches.

In [14]:
```
sns.set(palette='BrBG')
df.hist(figsize=(5,10));
```

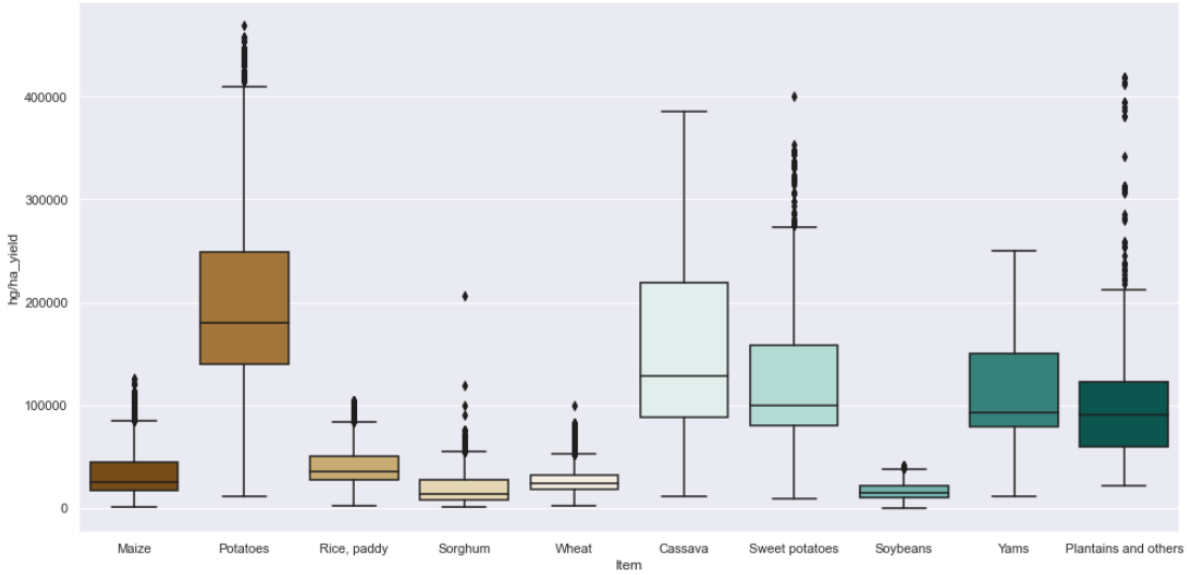


Boxplot

A boxplot (box-and-whisker plot) is a graphical representation of the distribution of a dataset. It provides a summary of the central tendency, dispersion, and skewness of the data. Boxplots are especially useful for comparing the distribution of multiple groups or variables. Boxplots are effective for visually comparing the central tendency and spread of different groups or variables, especially when dealing with skewed or non-normal distributions. They are commonly used in exploratory data analysis and statistical analysis.

In [16]:
```
a4_dims = (16.7, 8.27)

fig, ax = plt.subplots(figsize=a4_dims)
sns.boxplot(x="Item",y="hg/ha_yield",palette="BrBG",data=df,ax=ax)
```

Out[16]: <AxesSubplot:xlabel='Item', ylabel='hg/ha_yield'>



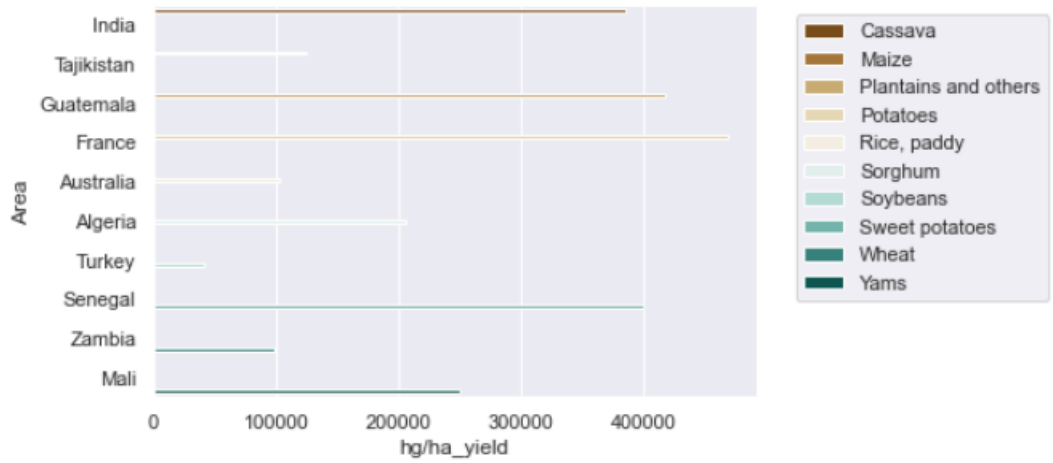
In [18]:
```
grouped = df.groupby('Item')

best_areas = []

for item, group in grouped:
    max_production_row = group[group['hg/ha_yield'] == group['hg/ha_yield'].max()]

    area = max_production_row['Area'].values[0]
    production = max_production_row['hg/ha_yield'].values[0]

    best_areas.append({'Item': item, 'Area': area, 'hg/ha_yield': production})

best_areas_df = pd.DataFrame(best_areas)

best_areas_df
```

Out[18]:

	Item	Area	hg/ha_yield
0	Cassava	India	385818
1	Maize	Tajikistan	125670
2	Plantains and others	Guatemala	418505
3	Potatoes	France	468991
4	Rice, paddy	Australia	103895
5	Sorghum	Algeria	206000
6	Soybeans	Turkey	41609
7	Sweet potatoes	Senegal	400000
8	Wheat	Zambia	99387
9	Yams	Mali	250000

Bar graph

A bar graph can be defined as a graphical representation of data, quantities, or numbers using bars or strips. They are used to compare and contrast different types of data, frequencies, or other measures of distinct categories of data.

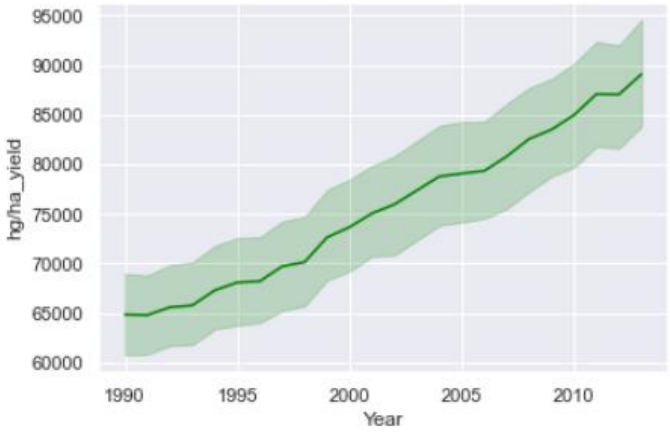


Lineplot

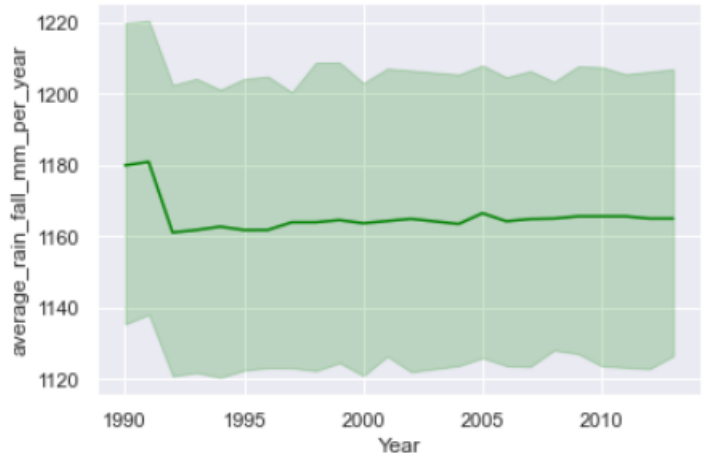
A line plot is a chart displaying data points along a number line. It is particularly useful for showing trends or patterns over a continuous interval or time. It is commonly used to visualize trends and patterns over time or a continuous variable. Line plots are effective for displaying how a variable changes about another variable or over a specific time period. It is possible to show up to three dimensions independently by using all three semantic types, but this style of plot can be hard to interpret and is often ineffective.

In [20]:
```
import matplotlib.pyplot as plt
import seaborn as sns

def change_of_years(data, template='seaborn'):
    col = data.columns[3:].tolist()
    for i in col:
        sns.lineplot(data=data, x='Year', y=i, color='green')
#         plt.title('Effect of Years on {i}')
        plt.show()
        yield plt

```

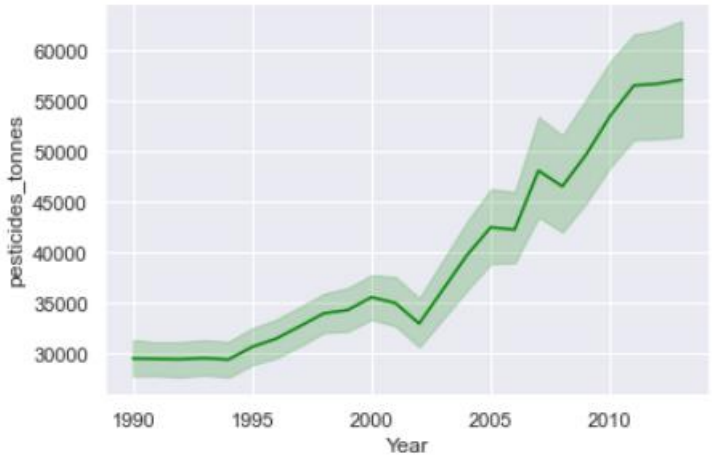
In [21]:
```
yplot = change_of_years(df)
next(yplot);
```



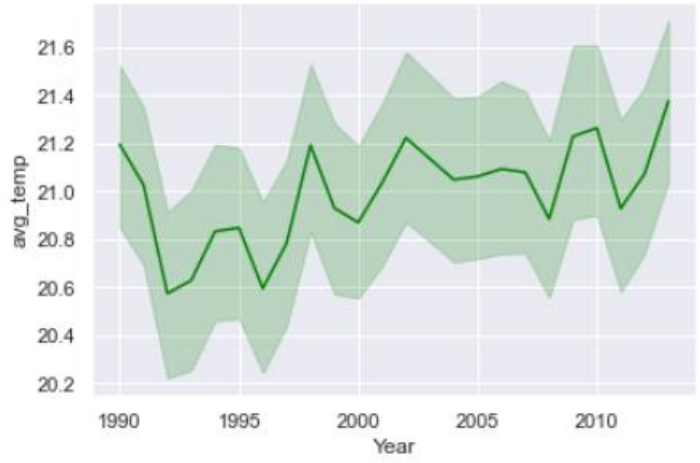
In [22]:
```
1 next(yplot);
```



In [23]:
```
1 next(yplot);
```



In [24]:
```
1 next(yplot);
```



4. Data Split

Basically, the Machine learning algorithm works in two different stages so we usually split the data to apply the machine learning algorithm. The whole data is divided into two different parts, which are generally 70 and 30 percent.

Splitting the dataset into training and test data

The same approach we are using here by diving out data in the Test and Train dataset.

By using the train test split in we have split our dataset now our task is to do training our dataset so that we can achieve our desired output but to train our dataset we need to apply a machine learning algorithm on our datasets, so this is our final and most important task to perform.

```
In [27]: 1 X, y = datacorr.drop(labels='hg/ha_yield', axis=1), datacorr['hg/ha_yield']

In [28]: 1 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

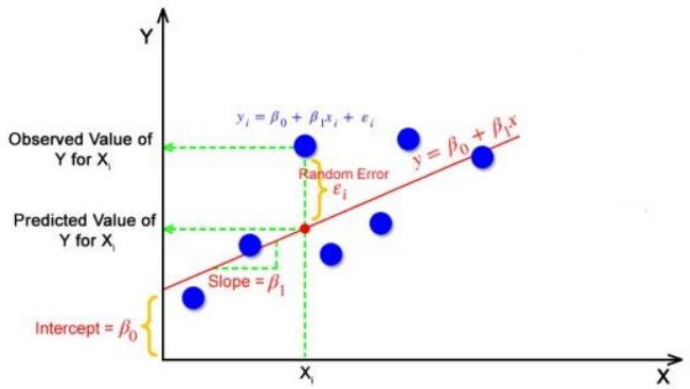
5. Machine Learning Algorithms

In [25]:
```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error, mean_absolute_percentage_error
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import BaggingRegressor
```

In [41]:
```
results = []
import math
models = [
    ('Linear Regression', LinearRegression()),
    ('Decision Tree',DecisionTreeRegressor(random_state=42)),
    ('Random Forest', RandomForestRegressor(n_estimators=1000,random_state=42)),
    ('Gradient Boost', GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, max_depth=3,random_state=42)),
    ('XGBoost', XGBRegressor(random_state=42)),
    ('KNN',KNeighborsRegressor(n_neighbors=5)),
    ('Bagging Regressor',BaggingRegressor(n_estimators=150, random_state=42))
          ]

for name, model in models:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = model.score(X_test, y_test)
    MSE = math.sqrt(mean_squared_error(y_test, y_pred))
    R2_score = r2_score(y_test, y_pred)
    results.append((name, accuracy, MSE, R2_score))
    acc = (model.score(X_train , y_train)*100)
    print(f'The accuracy of the {name} Model Train is {acc:.2f}')
    acc =(model.score(X_test , y_test)*100)
    print(f'The accuracy of the  {name} Model Test is {acc:.2f}')
    plt.scatter(y_test, y_pred,s=10,color='#9B673C')
    plt.xlabel('Actual Values')
    plt.ylabel('Predicted Values')
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='green', linewidth = 4)
    plt.show()
```

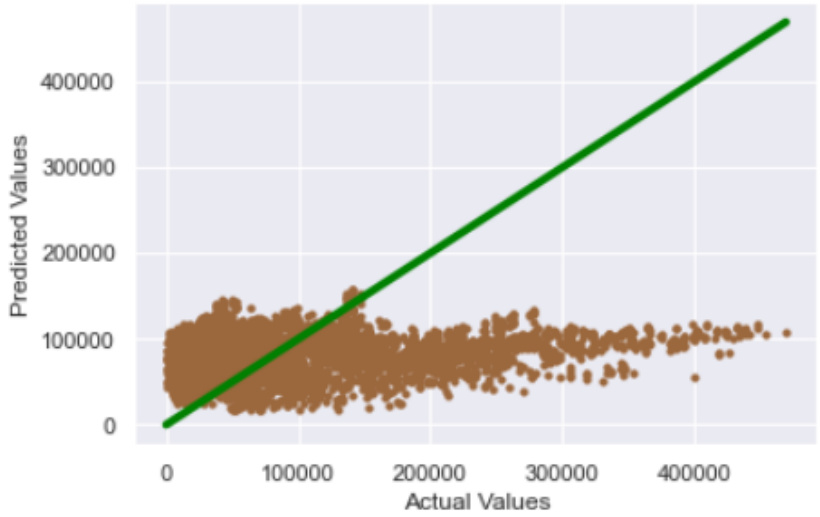
a. Linear Regression

Linear regression predicts the relationship between two variables by assuming a linear connection between the independent and dependent variables. It seeks the optimal line that minimizes the sum of squared differences between predicted and actual values. Applied in various domains like economics and finance, this method analyzes and forecasts data trends. It can extend to multiple linear regression involving several independent variables and logistic regression, suitable for binary classification problems



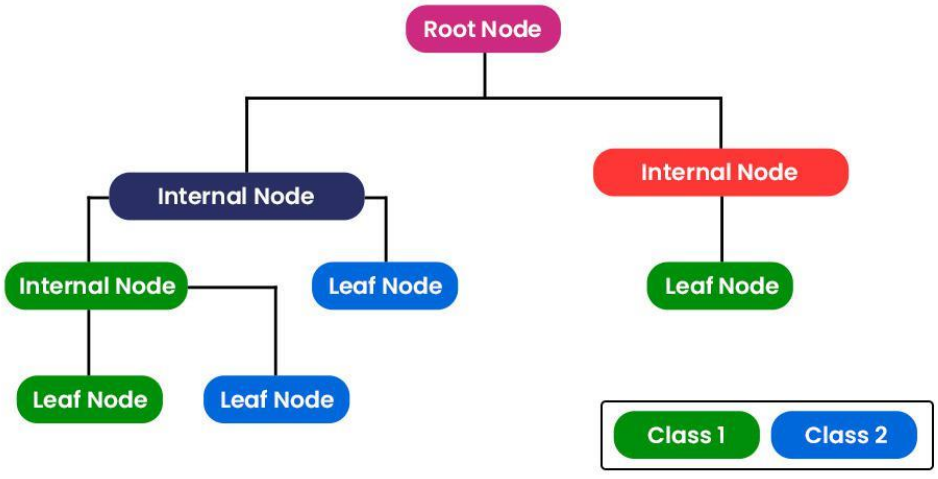
Applying this model in the project,

```
The accuracy of the Linear Regression Model Train is 7.39
The accuracy of the  Linear Regression Model Test is 7.37
```



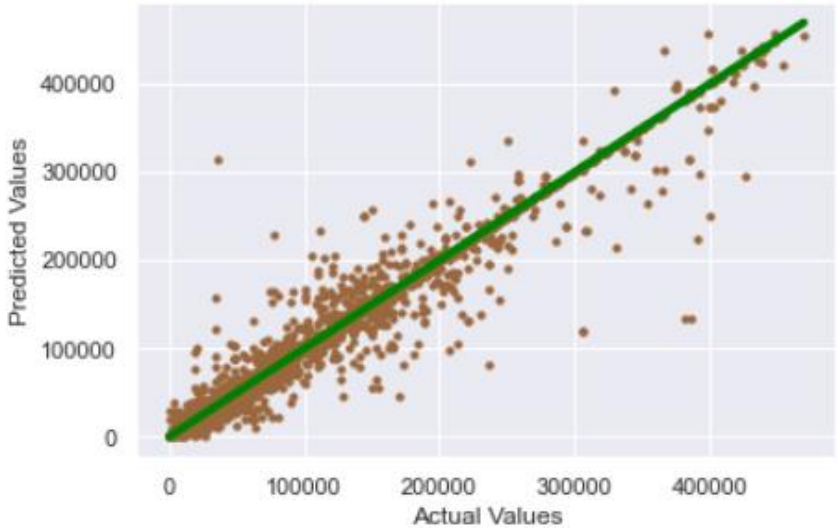
b. Decision Tree

The decision tree is considered among the most effective and common algorithms for classification and future prediction. DT is a conceptual tree-alike model, where each internal node represents a feature that best split the data into subsets using statistical measures such as information gain and gain ratio, the process of splitting data is a recursive process until reaching the leaf (normally one of class labels). In machine learning, DT is one of the supervised learning algorithms.



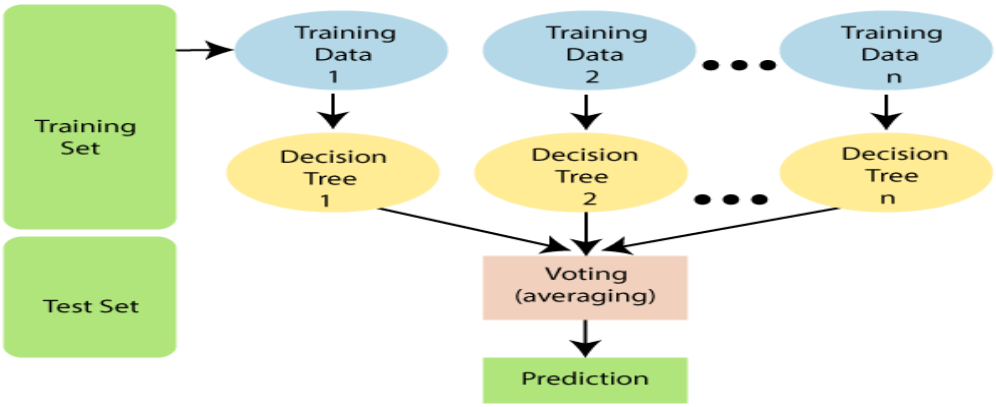
Applying this model in the project,

```
The accuracy of the Decision Tree Model Train is 100.00
The accuracy of the  Decision Tree Model Test is 97.62
```



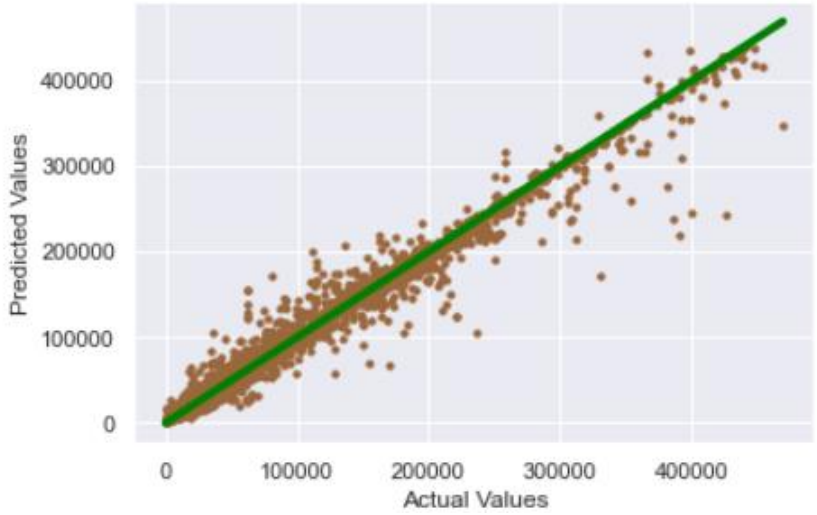
c. Random Forest

It is one of the most popular machine-learning algorithms for regression and classification tasks. RF creates a number of decision trees called forest trees to enhance the prediction process and produce higher accuracy. Building an RF tree is similar to a decision tree (DT) using information gain or other measures. Since RF is a set of DTs; each tree obtains a certain output and RF will choose the majority output produced by DTs or the mean in case of a regression problem.



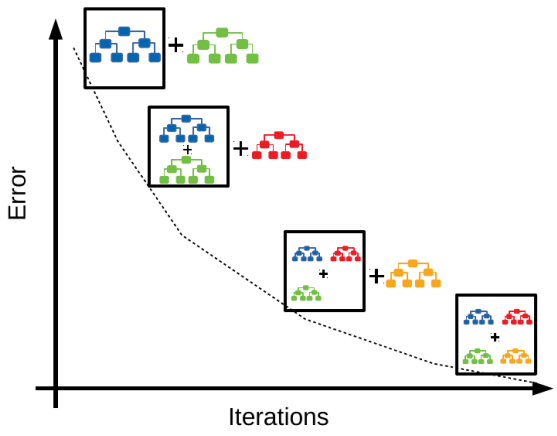
Applying this model in the project,

```
The accuracy of the Random Forest Model Train is 99.81
The accuracy of the  Random Forest Model Test is 98.58
```



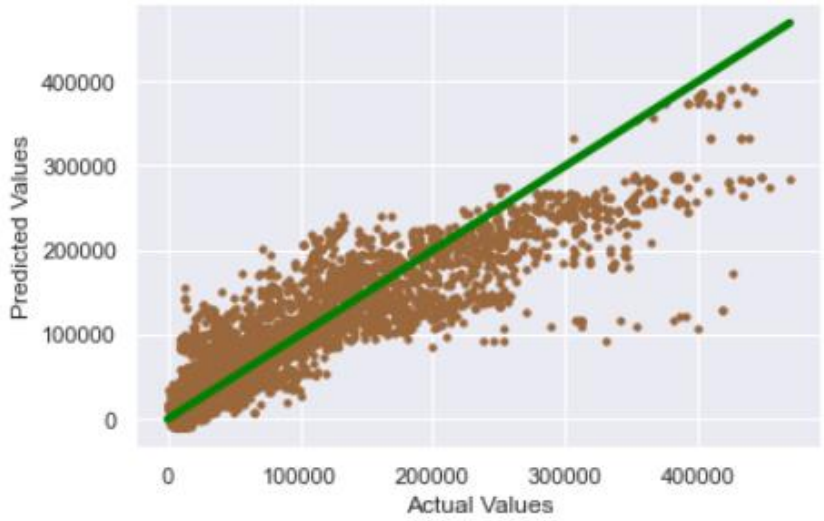
d. Gradient Boost

Gradient boosting is a machine learning technique used in regression and classification tasks. It provides a prediction model in the form of an ensemble of weak prediction models, or models that, usually consisting of straightforward decision trees, make very few assumptions about the data. Gradient-boosted trees is the name of the resulting algorithm when a decision tree serves as the weak learner; it often performs better than random forest. Similar to previous boosting techniques, a gradient-boosted trees model is constructed step-by-step. However, it differs from the other techniques in that it permits optimisation of any differentiable loss function.



Applying this model in the project,

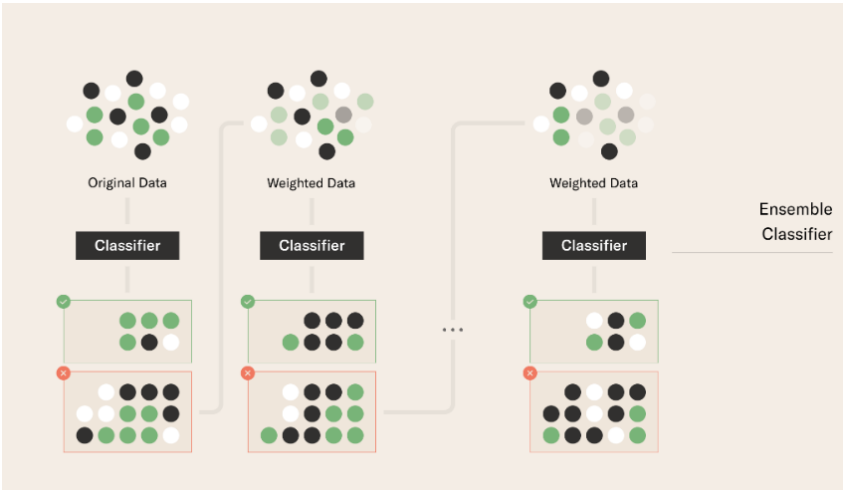
```
The accuracy of the Gradient Boost Model Train is 84.47
The accuracy of the  Gradient Boost Model Test is 83.11
```



e. XG Boost

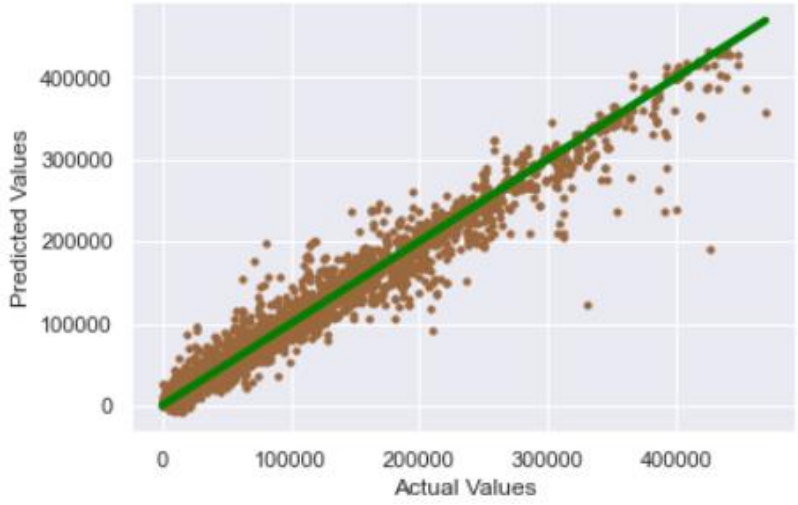
XGBoost, which stands for eXtreme Gradient Boosting, is a popular and powerful machine learning algorithm that belongs to the ensemble learning family. Ensemble learning involves combining the predictions of multiple models to create a more robust and accurate model. XGBoost, in particular, is an implementation of gradient boosted decision trees designed for speed and performance.

Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset's features.



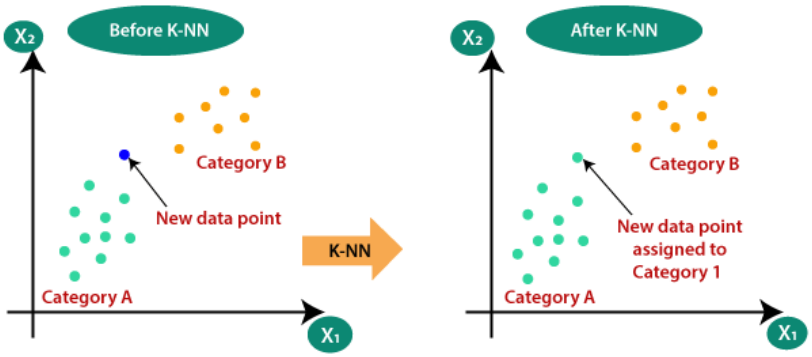
Applying this model in the project,

```
The accuracy of the XGBoost Model Train is 98.82
The accuracy of the  XGBoost Model Test is 97.43
```



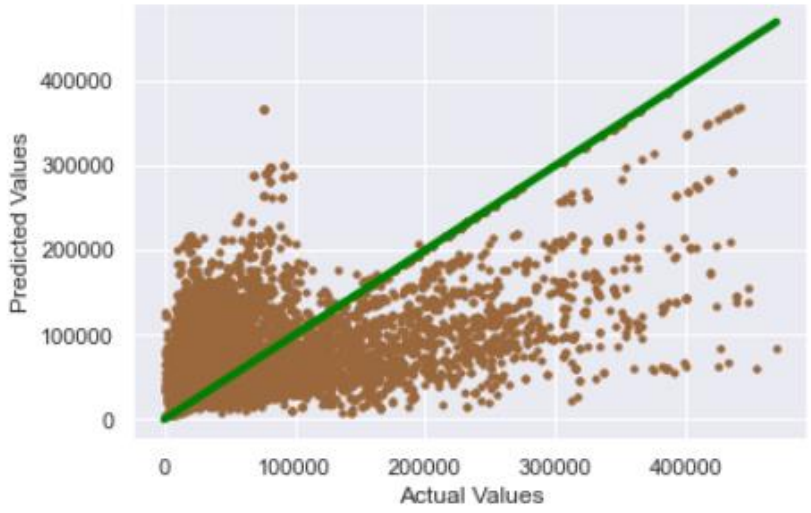
f. K-Nearest Neighbor (KNN)

K-Nearest Neighbors (KNN) is a simple machine learning algorithm for classification and regression tasks. It operates by utilising feature space to locate the k training samples that are closest to a given input, where "k" is a user-defined value. The projected class for classification is determined by the predominant class among the neighbours. The algorithm calculates the average in regression (for continuous output). KNN relies on the assumption that similar inputs have similar outputs. It is non-parametric and instance-based, making it versatile but sensitive to noise. The choice of k influences model performance, and distance metrics like Euclidean or Cosine are crucial for neighbor identification.



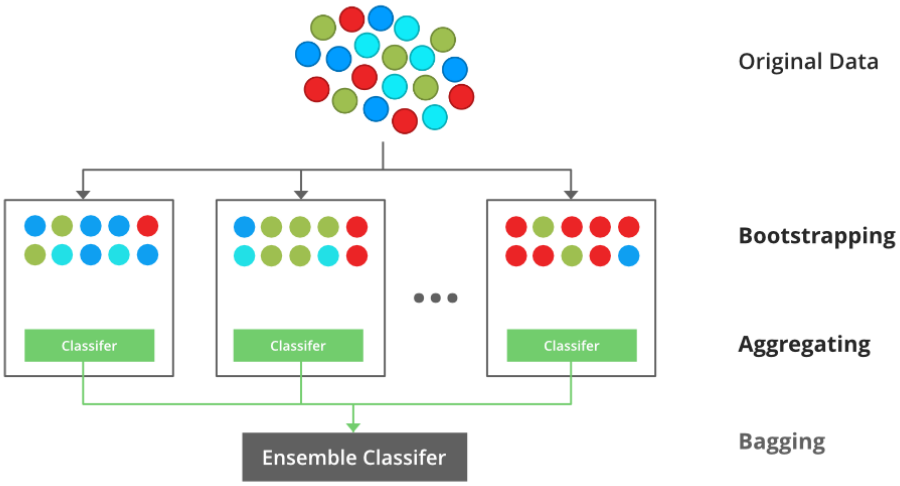
Applying this model in the project,

```
The accuracy of the KNN Model Train is 56.73
The accuracy of the  KNN Model Test is 28.90
```



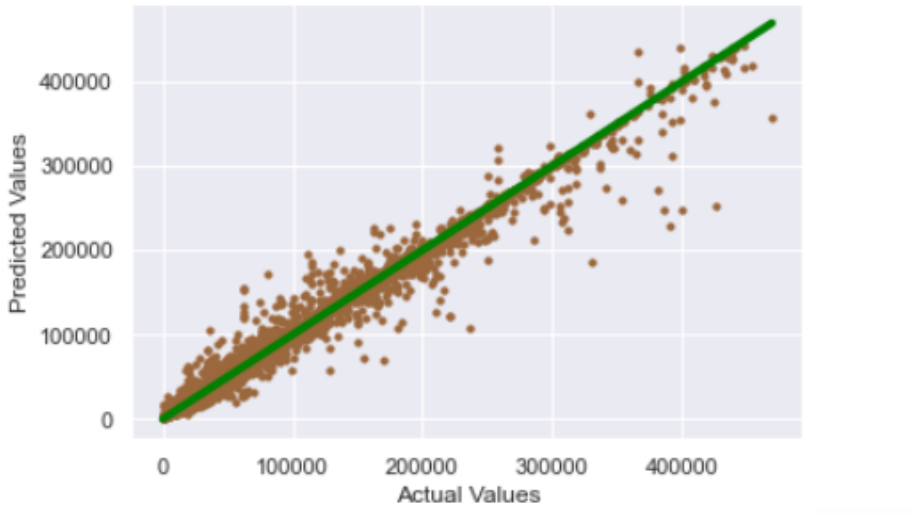
g. Bagging Regressor

A Bagging Regressor is an ensemble learning method that belongs to the bagging (Bootstrap Aggregating) family of algorithms. The idea behind bagging is to train multiple instances of the same base model on different subsets of the training data and then aggregate their predictions (either by voting or by averaging) to reduce overfitting and improve generalization.



Applying this model in the project,

```
The accuracy of the Bagging Regressor Model Train is 99.81
The accuracy of the  Bagging Regressor Model Test is 98.59
```



6. Performance metrics

To analyze the performance of a machine learning model we need some metrics. These metrics are statistical criteria that can be used to measure and monitor the performance of a model.

Root Mean Square Error(RMSE)

Root Mean Squared Error (RMSE) is another common metric used to evaluate the accuracy of a regression model, particularly in the context of predictive modeling and forecasting. It measures the average magnitude of the errors between predicted values and actual values. It shows how far predictions fall from measured true values using Euclidean distance.

To compute RMSE, calculate the residual (difference between prediction and truth) for each data point, compute the norm of residual for each data point, compute the mean of residuals and take the square root of that mean. RMSE is commonly used in supervised learning applications, as RMSE uses and needs true measurements at each predicted data point.

Root mean square error can be expressed as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}},$$

where N is the number of data points, y(i) is the i-th measurement, and ŷ(i) is its corresponding prediction.

R2 Score

R^2 score also known as Coefficient of determination is used to evaluate the performance of a linear regression model. It is the amount of the variation in the output dependent attribute which is predictable from the input independent variable(s). It is used to check how well-observed results are reproduced by the model, depending on the ratio of total deviation of results described by the model. It ranges from 0 to 1.

Mathematical Formula:

$R^2 = 1 - SS_{res} / SS_{tot}$

The machine learning models are validated by comparing the performance metrics.

5. RESULTS

In [42]:
```
dff = pd.DataFrame(results, columns=['Model', 'Accuracy', 'MSE', 'R2_score'])
df_styled_best = dff.style.highlight_max(subset=['Accuracy','R2_score'], color='green').
highlight_min(subset=['MSE'], color='green').highlight_max(subset=['MSE'], color='red').
highlight_min(subset=['Accuracy','R2_score'], color='red')
display(df_styled_best)
```

	Model	Accuracy	MSE	R2_score
0	Linear Regression	0.073724	79332.959921	0.073724
1	Decision Tree	0.976174	12723.613777	0.976174
2	Random Forest	0.985828	9812.822175	0.985828
3	Gradient Boost	0.831140	33872.492929	0.831140
4	XGBoost	0.974317	13210.005690	0.974317
5	KNN	0.289040	69503.430377	0.289040
6	Bagging Regressor	0.985881	9794.589505	0.985881

Accuracies for models

After seeing the result, we can say that Bagging Regressor is performing best with highest accuracy and least error among all models.

6. CONCLUSION

In conclusion, "**Crop Yield Prediction**" plays a pivotal role in modern agriculture, offering valuable insights to farmers, researchers, and policymakers. The integration of advanced technologies, such as machine learning algorithms, satellite imagery, and weather data, has significantly improved the accuracy and efficiency of yield forecasting. These predictive models enable farmers to make informed decisions regarding crop management, resource allocation, and risk mitigation.

The benefits of crop yield prediction extend beyond individual farm management, impacting global food security and economic stability. Accurate forecasts empower stakeholders to anticipate potential challenges, adapt to changing environmental conditions, and optimize agricultural practices. Moreover, the data-driven approach enhances sustainability by minimizing resource wastage and environmental impact.

7. Future Work

- Develop models that can adapt and learn continuously over time. This involves creating systems that can update themselves with new data, allowing for more accurate and timely predictions as agricultural conditions evolve.
- Improve ML models to better predict and adapt to changing climate patterns, including extreme weather events. This can involve refining algorithms to handle non-linear and dynamic relationships between climate variables and crop yields.
- We can further ensemble two or more machine learning algorithms and process large data to get more accurate results.
- Developing a UI for the project, so any farmer can predict the crop yield beforehand by entering the details.

8. References

[1]. Crop yield prediction using machine learning: A systematic literature review https://www.sciencedirect.com/science/article/pii/S0168169920302301

[2]. An interaction regression model for crop yield prediction by Javad Ansarifar, Lizhi Wang & Sotirios V. Archontoulis https://www.nature.com/articles/s41598-021-97221-7

[3]. A Comprehensive Review of Crop Yield Prediction Using Machine Learning Approaches https://ieeexplore.ieee.org/abstract/document/9410627

[4]. Kaggle for Dataset: https://www.kaggle.com/datasets/noorsaeed/crop-yield-prediction-dataset