**Final Project Report**

| Date | 12 July 2024 |
| --- | --- |
| Team ID | SWTID1720077079 |
| Project Name | Wild Blueberry Yield Prediction |

**1) Introduction**

**1.1. Project Overview**

The Blueberry Yield Prediction System represents a significant advancement in agricultural technology aimed at mitigating challenges faced by blueberry farmers. Traditional methods of yield estimation often fall short due to their inability to account for dynamic factors such as fluctuating weather conditions, varying soil health, and unpredictable pest activities. These uncertainties contribute to financial instability among farmers, either through overestimation leading to surplus or underestimation resulting in revenue loss.

In response to these challenges, our project harnesses the power of machine learning to develop a robust prediction system. By integrating historical yield data with real-time environmental variables, the system aims to provide accurate forecasts crucial for optimizing farming operations. This initiative not only seeks to enhance yield prediction accuracy but also aims to empower farmers with actionable insights into crop management practices.

**1.2. Objectives**

The primary objective of the Blueberry Yield Prediction System is to empower blueberry farmers with reliable forecasting tools. Key objectives include:

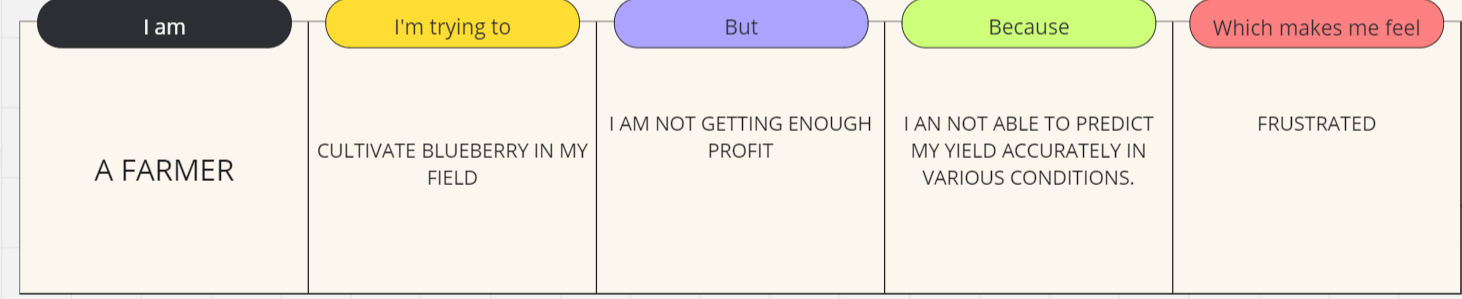
* **Accuracy**: Develop machine learning models that accurately predict blueberry yields based on a comprehensive dataset encompassing weather patterns, soil conditions, and pest dynamics.
* **Operational Efficiency**: Enable farmers to make informed decisions regarding harvesting schedules, resource allocation, and market strategies, thereby optimizing operational efficiency.
* **Financial Stability**: Mitigate financial risks associated with yield estimation errors, facilitating better financial planning and resource utilization.
* **Sustainability**: Foster sustainable farming practices by reducing waste and optimizing resource allocation through data-driven insights.

By achieving these objectives, the project aims to elevate the productivity and profitability of blueberry farming while promoting environmental sustainability and resilience against unpredictable agricultural conditions.

**2) Project Initialization and Planning Phase**

**2.1 Define Problem Statement**

Blueberry farmers face significant challenges in predicting their yield accurately due to reliance on traditional methods, unpredictable weather patterns, soil conditions, and pest infestations, leading to financial instability from overestimation or underestimation of produce. There is a critical need for a reliable, precise yield prediction system utilizing machine learning to provide accurate predictions that consider various factors such as weather, soil health, and pest activity. This system will enable farmers to plan their harvesting and marketing strategies, optimize resource allocation, enhance financial planning, reduce waste, and improve overall productivity. By addressing these needs, the machine learning-based Blueberry Yield Prediction System aims to offer accurate and timely yield predictions, insights into crop yield factors, data-driven recommendations for crop management, and a user-friendly interface, ultimately contributing to increased productivity, better financial planning, and enhanced sustainability in blueberry farming. Success will be measured by reduction in yield prediction errors, increased farmers' income, user satisfaction, and improved resource utilization.

****

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem**  **Statement (PS)** | **I am** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| PS-1 | Farmer | To cultivate | Have low profits | Of poor yield | Poor |
| PS-2 | Middle man | Buy product from farmers | No fixed income | Of variation in production | Disappointed |
| PS-3 | Consumer | Get the fruits from shop | It may not be available when I want | Of unpredictable production | Malnourishment |

**2.2 Project Proposal (Proposed Solution)**

This project employs a machine learning system for accurate blueberry yield prediction, addressing farmer challenges with unpredictable weather and soil conditions. It includes data collection, advanced modeling, user-friendly interface integration, and rigorous testing. Required resources: high-performance computing, and a skilled team of data scientists, engineers, designers, and testers. Goal: Empower farmers with reliable yield forecasts to improve decision-making and operational efficiency.

|  |  |
| --- | --- |
| **Project Overview** | |
| Objective | Develop a ML system for accurate blueberry yield prediction. |
| Scope | Includes data collection, advanced modeling and rigorous testing for operational reliability. |
| **Problem Statement** | |
| Description | This project aims to create a machine learning system to predict blueberry yields accurately. |
| Impact | Enhances farmers' ability to make informed decisions and improve operational efficiency by providing reliable yield forecasts. |
| **Proposed Solution** | |
| Approach | Utilize scikit- learn for developing predictive models integrating historical yield data, weather patterns, soil health, and pest dynamics. |
| Key Features | Includes comprehensive data preprocessing, advanced feature engineering, scikit- learn based model development, user-friendly interface integration, and rigorous testing for reliability. |

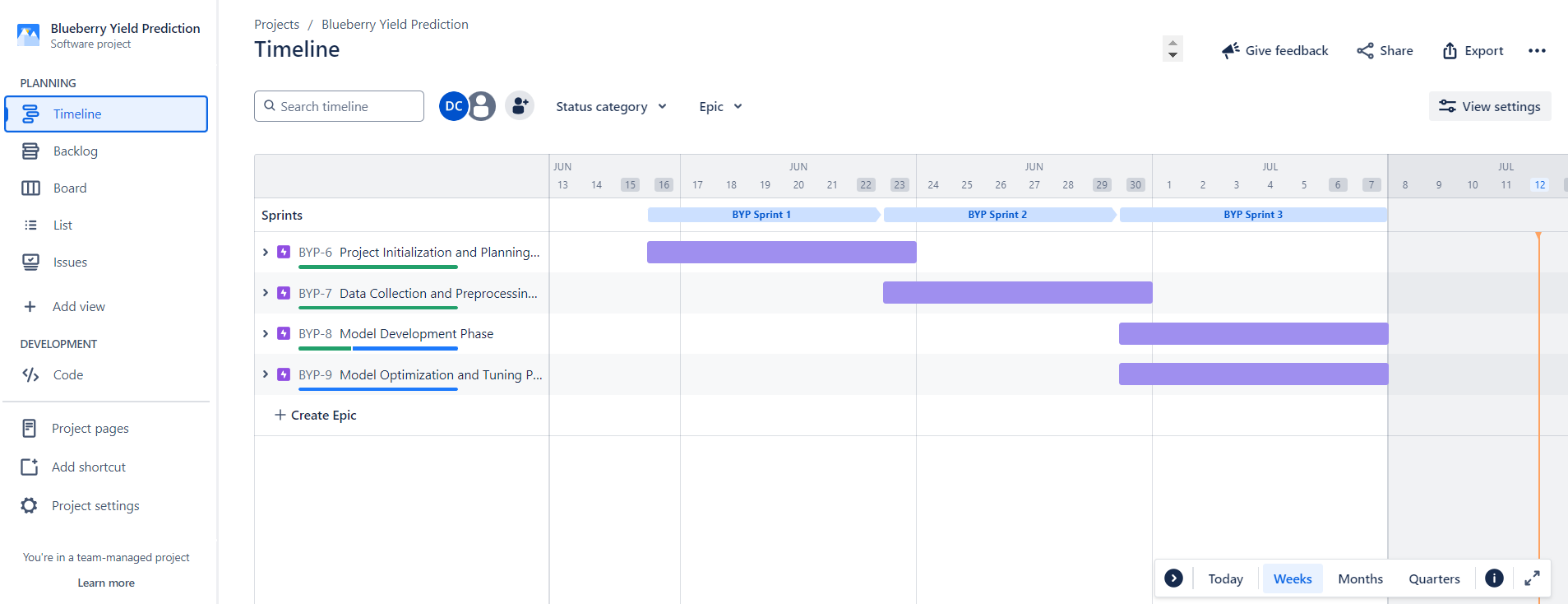
**Resource Requirements**

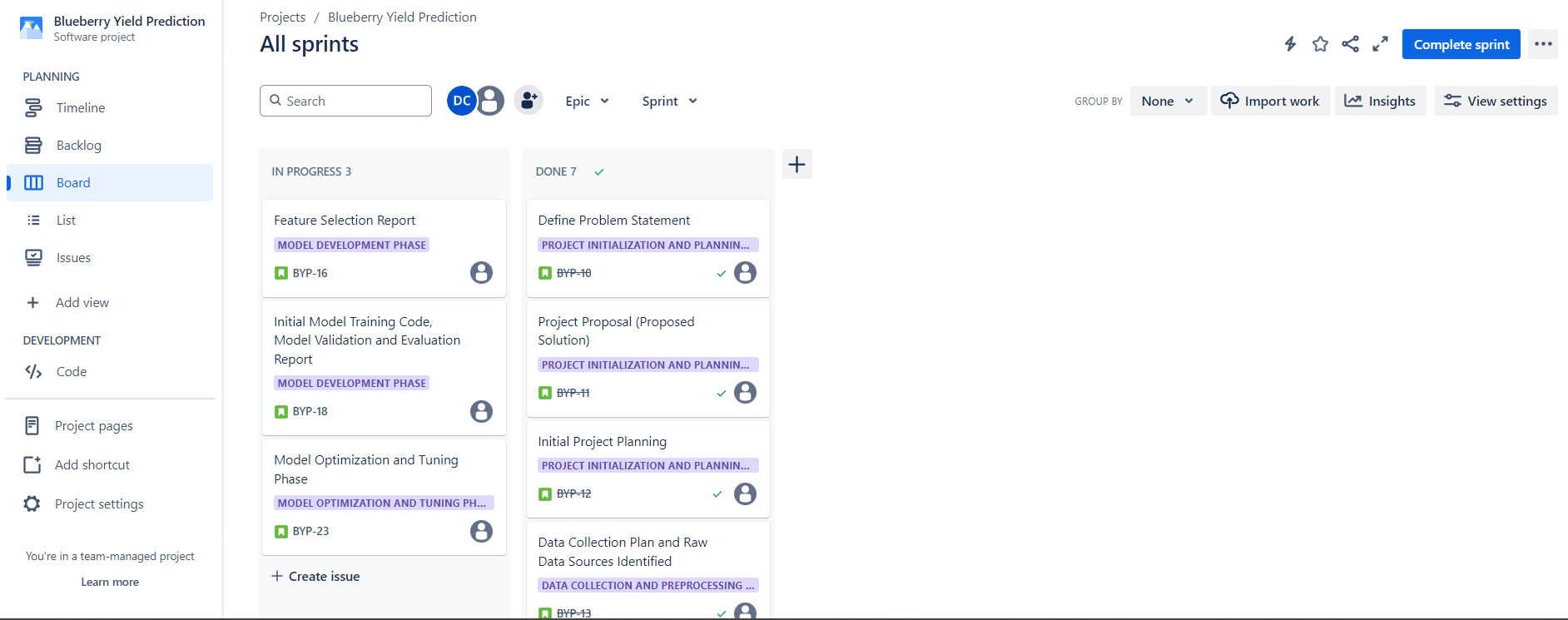
|  |  |  |
| --- | --- | --- |
| **Resource Type** | **Description** | **Specification/Allocation** |
| **Hardware** | | |
| Computing Resources | CPU/GPU specifications, number of cores | Core i5, 11th gen  Nvidia GTX |
| Memory | RAM specifications | 8 GB |
| Storage | Disk space for data, models, and logs | 512 GB SSD |
| **Software** | | |
| Frameworks | Python frameworks | Flask |
| Libraries | Additional libraries | scikit-learn, pandas, numpy, matplotlib, seaborn, pickle |
| Development Environment | IDE, version control | Jupyter Notebook, Git, Spyder |
| **Data** | | |
| Data | Source, size, format | Kaggle dataset, 85 KB, CSV |

**2.3 Initial Project Planning**

**Product Backlog, Sprint Schedule, and Estimation**

| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** | **Sprint Start Date** | **Sprint End Date (Planned)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | Project Initialization and Planning Phase | BYP-10 | Define Problem Statement | 1 | Low | Jackson | 16/06/2024 | 23/06/2024 |
| Sprint-1 | Project Initialization and Planning Phase | BYP-11 | Project Proposal (Proposed Solution) | 1 | Low | Duane | 16/06/2024 | 23/06/2024 |
| Sprint-1 | Project Initialization and Planning Phase | BYP-12 | Initial Project Planning | 1 | Low | Harisankar | 16/06/2024 | 23/06/2024 |
| Sprint-2 | Data Collection and Preprocessing Phase | BYP-13 | Data Collection Plan and Raw Data Sources Identified | 2 | Moderate | Jackson | 23/06/2024 | 30/06/2024 |
| Sprint-2 | Data Collection and Preprocessing Phase | BYP-14 | Data Quality Report | 2 | Low | Amith | 23/06/2024 | 30/06/2024 |
| Sprint-2 | Data Collection and Preprocessing Phase | BYP-15 | Data Exploration and Preprocessing | 2 | Moderate | Duane | 23/06/2024 | 30/06/2024 |
| Sprint-3 | Model Development Phase | BYP-16 | Feature Selection Report | 1 | Low | Amith | 30/06/2024 | 07/07/2024 |
| Sprint-3 | Model Development Phase | BYP-18 | Initial Model Training Code, Model Validation and Evaluation Report | 2 | High | Duane | 30/06/2024 | 07/07/2024 |
| Sprint-3 | Model Development Phase | BYP-17 | Model Selection Report | 2 | High | Jackson | 30/06/2024 | 07/07/2024 |
| Sprint-3 | Model Optimization and Tuning Phase | BYP-23 | Model Optimization and Tuning Phase | 1 | High | Harisankar | 30/06/2024 | 07/07/2024 |

****

****

**3) Data Collection and Preprocessing Phase**

**3.1 Data Collection Plan and Raw Data Sources Identified**

**Data Collection Plan**

|  |  |
| --- | --- |
| **Section** | **Description** |
| Project Overview | A machine learning-based system to accurately predict blueberry yields, addressing the challenges faced by farmers in yield estimation. |
| Data Collection Plan | Obtains a dataset from Kaggle |
| Raw Data Sources Identified | CSV file from Kaggle (87 kb)  The dataset includes 777 entries with 18 columns detailing blueberry yield factors such as clone size, pollinator counts, temperature ranges, rainy days, fruit set rate, mass, seed count, and overall yield. |

**Raw Data Sources**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Source Name** | **Description** | **Location/URL** | **Format** | **Size** | **Access Permissions** |
| Kaggle | A csv file detailing blueberry yield factors such as clone size, pollinator counts, temperature ranges, rainy days, fruit set rate, mass, seed count, and overall yield. | https://www.kaggle.com/datasets/saurabhshahane/wild-blueberry-yield-prediction | CSV | 87 KB | Public |

**3.2 Data Quality Report**

The dataset, provided as a CSV file, is of high quality and well-suited for our blueberry yield prediction project. It contains no missing values or duplicate entries, ensuring data integrity. Additionally, the dataset does not require label encoding or one-hot encoding, simplifying the preprocessing steps. Overall, it is an excellent dataset for our intended purpose.

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Source** | **Data Quality Issue** | **Severity** | **Resolution Plan** |
| Kaggle | Outlier issue in honeybee feature | Moderate | Fitting the feature based on interquartile range found from boxplot. |

**3.3 Data Exploration and Preprocessing**

|  |  |
| --- | --- |
| **Section** | **Description** |
| Data Overview |  |
| Univariate Analysis |  |
| Bivariate Analysis |  |
| Multivariate Analysis |  |
| Outliers and Anomalies | Outlier in feature ‘honeybee’ found using boxplot    Handled outlier |
| **Data Preprocessing Code Screenshots** | |
| Loading Data |  |
| Handling Missing Data |  |
| Data Transformation |  |
| Feature Engineering | For handling outlier |
| Save Processed Data |  |

**4) Model Development Phase**

**4.1 Feature Selection Report**

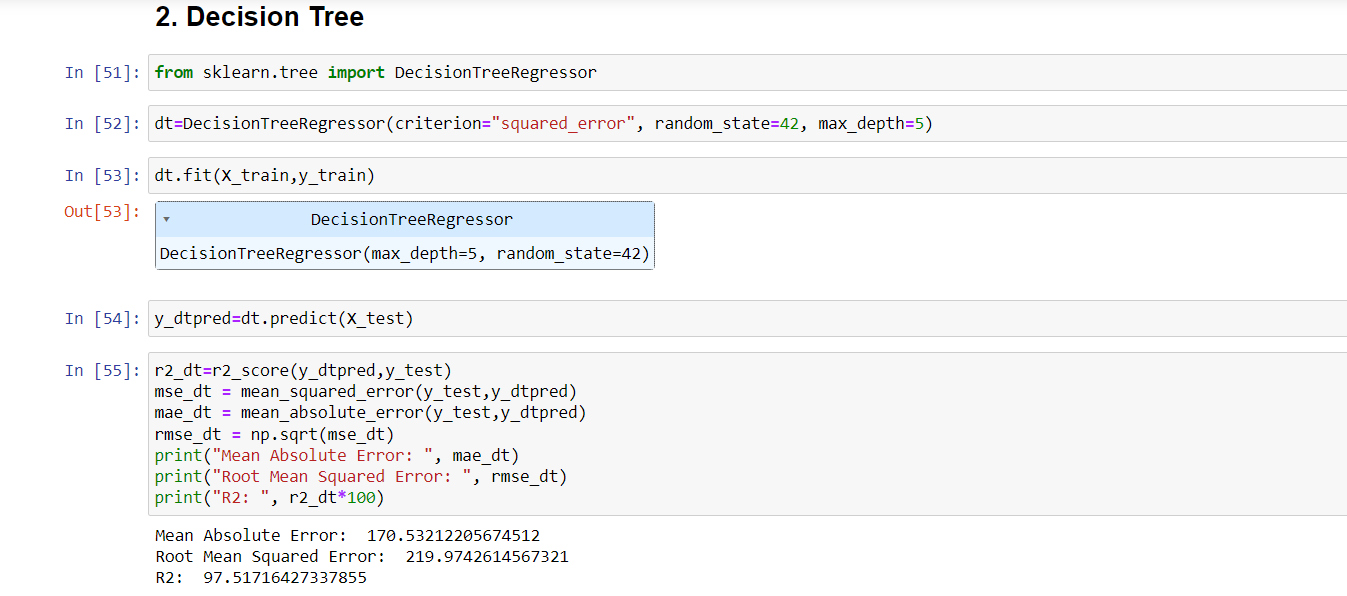
|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Description** | **Selected (Yes/No)** | **Reasoning** |
| Clonesize | The average blueberry clone size in the field | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| Honeybee | Honeybee density in the field | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| Bumbles | Bumblebee density in the field | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| Andrena | Andrena bee density in the field | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| Osmia | Osmia bee density in the field | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| MaxOfUpperTRange | The highest record of the upper band daily air temperature during the bloom season | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| MinOfUpperTRange | The lowest record of the upper band daily air temperature | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| AverageOfUpperTRange | The average of the upper band daily air temperature | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| MaxOfLowerTRange | The highest record of the lower band daily air temperature | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| MinOfLowerTRange | The lowest record of the lower band daily air temperature | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| AverageOfLowerTRange | The average of the lower band daily air temperature | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| RainingDays | The total number of days during the bloom season, each of which has precipitation larger than zero | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| AverageRainingDays | The average of raining days of the entire bloom season | Yes | Univariate, bivariate, multivariate analysis shows good correlation |
| fruitset | Refers to the proportion of flowers that develop into fruits. | No | Shows very low correlation with other features from multivariate analysis |
| fruitmass | Indicates the mass (weight) of the fruits. | No | Shows very low correlation with other features from multivariate analysis |
| seeds | Represents the number of seeds produced within the fruits. | No | Shows very low correlation with other features from multivariate analysis |

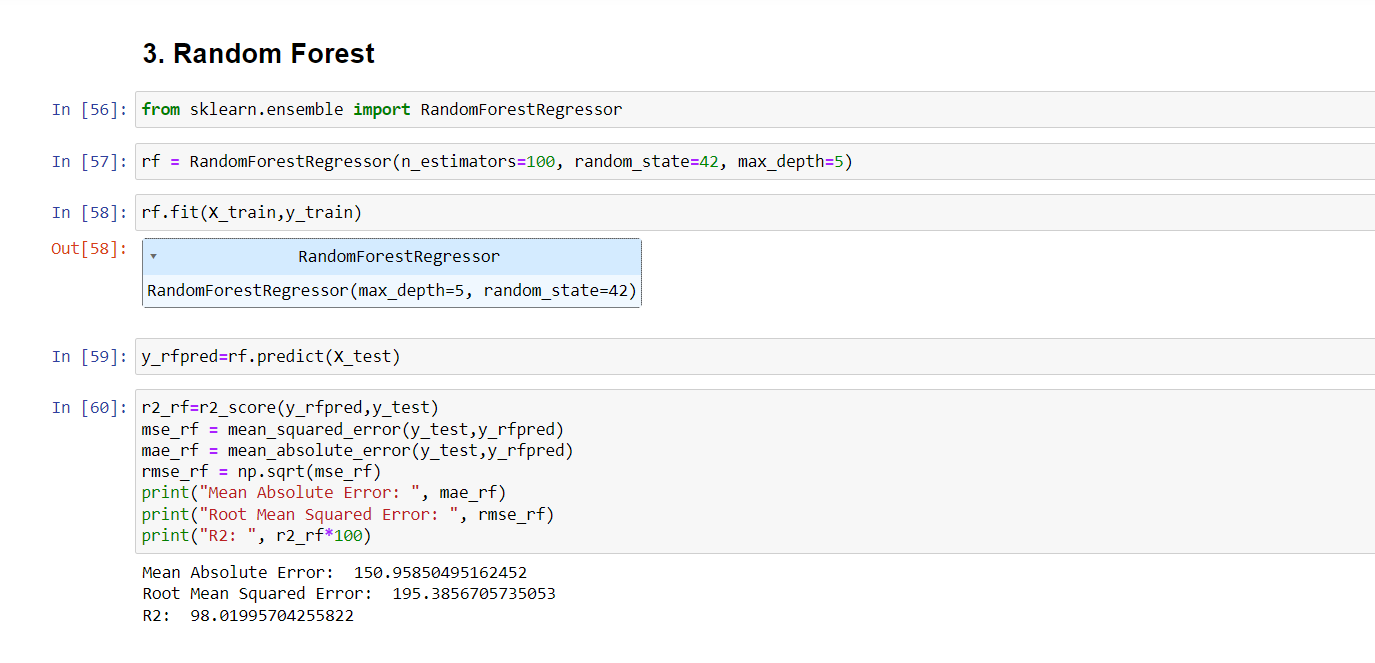
**4.2 Model Selection Report**

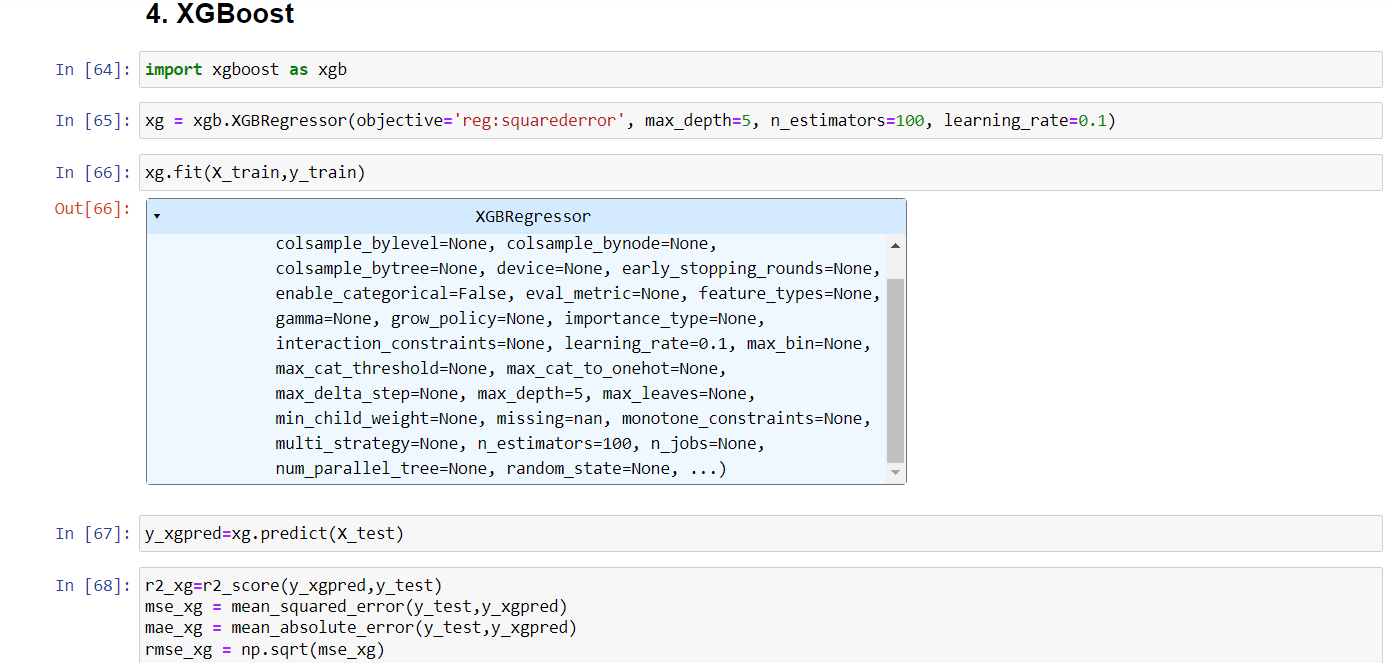
|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Description** | **Hyperparameters** | **Performance Metric** |
| Linear Regression | Linear Regression is a simple and interpretable model, but it assumes a linear relationship between the features and the target variable | fit\_intercept=True  copy\_X=True  n\_jobs=None  positive=False |  |
| Decision Tree | It constructs a tree-like model of decisions and their possible consequences. | random\_state=42  max\_depth=5 |  |
| Random Forest | Random Forest is a collection of individual Decision Tree models, where each tree is trained on a random subset of the training data and a random subset of the features. | n\_estimators=100  random\_state=42  max\_depth=5 |  |
| XGBoost | The XGBoost model is a gradient boosting algorithm used for regression tasks, with the objective function set to 'reg:squarederror' to optimize the squared error loss, which is appropriate for regression problems. | max\_depth=5  n\_estimators=100  learning\_rate=0.1 |  |
| SVM Regression | SVM Regression is a supervised learning algorithm used for solving regression problems. It works by finding the best-fitting hyperplane in a high-dimensional feature space that minimizes the error between the predicted and actual target values. | C=100,  epsilon=0.001  gamma='auto'  kernel='linear' |  |

**4.3 Initial Model Training Code, Model Validation and Evaluation Report**





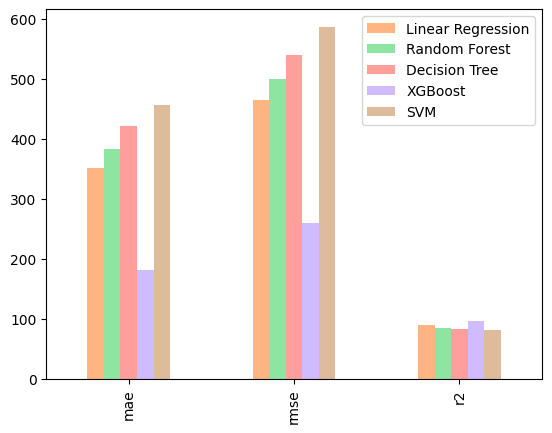






****

**Model Validation and Evaluation Report:**



|  |  |
| --- | --- |
| **Model** | **Performance Metrics** |
| Linear Regression |  |
| Decision Tree |  |
| Random Forest |  |
| XGBoost |  |
| SVM Regression |  |

**5) Model Optimization and Tuning Phase**

**5.1 Hyperparameter Tuning Documentation**

|  |  |  |
| --- | --- | --- |
| **Model** | **Tuned Hyperparameters** | **Optimal Values** |
| XGBoost - Grid Search Optimized |  | 'colsample\_bytree': 0.8, 'learning\_rate': 0.2, 'max\_depth': 3, 'min\_child\_weight': 3, 'n\_estimators': 300, 'subsample': 0.8 |
| XGBoost - Random Search Optimized |  | 'subsample': 1.0, 'n\_estimators': 411, 'min\_child\_weight': 1, 'max\_depth': 3, 'learning\_rate': 0.2, 'colsample\_bytree': 1.0 |

**5.2 Performance Metrics Comparison Report**

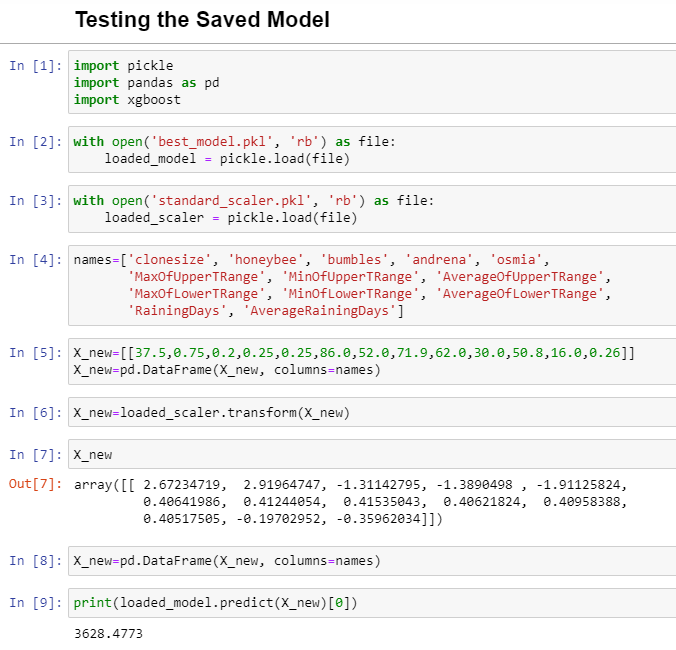
|  |  |  |
| --- | --- | --- |
| **Model** | **Baseline Metric** | **Optimized Metric** |
| XGBoost - Grid Search Optimized |  |  |
| XGBoost - Random Search Optimized |  |  |

**5.3 Final Model Selection Justification**

|  |  |
| --- | --- |
| **Final Model** | **Reasoning** |
| XGBoost - Random Search Optimized | It was chosen because of:   1. Low Mean Absolute Error 2. Low Root Mean Squared Error 3. High R2 Score   on testing set. |

### 6) Results

#### 6.1. Output Screenshots







### 7) Advantages and Disadvantages

#### 7.1. Advantages

#### Advantages of the Blueberry Yield Prediction System

1. **Improved Yield Accuracy**: By integrating machine learning models with comprehensive datasets, the Blueberry Yield Prediction System enhances yield prediction accuracy. This improvement enables farmers to plan harvesting schedules more effectively, reducing losses due to underestimation or surplus.
2. **Optimized Resource Allocation**: Accurate yield forecasts empower farmers to allocate resources such as labor, fertilizers, and pesticides more efficiently. This optimization not only reduces operational costs but also enhances overall farm productivity.
3. **Risk Mitigation**: The system mitigates financial risks associated with uncertainties in yield estimation. Farmers can make informed decisions regarding market strategies and financial planning, thereby improving profitability and sustainability.
4. **Environmental Sustainability**: By optimizing resource use and reducing waste, the system promotes sustainable farming practices. This includes minimizing the environmental impact of excessive pesticide or fertilizer use and aligning farming activities with ecological conservation goals.
5. **Real-Time Insights**: Real-time data integration allows for dynamic adjustments in farming practices based on current weather patterns, soil health indicators, and pest activity. This flexibility helps farmers respond swiftly to changing conditions, optimizing crop management strategies.
6. **Scalability and Adaptability**: The modular design of the system ensures scalability across different farm sizes and geographic locations. It can adapt to varying environmental conditions and farming practices, making it versatile for diverse agricultural settings.

#### 7.2. Disadvantages

#### Challenges and Limitations

1. **Data Dependency**: The accuracy of yield predictions heavily relies on the quality and availability of historical and real-time data. In regions with limited data infrastructure, achieving reliable forecasts may be challenging.
2. **Complexity of Models**: Implementing and maintaining sophisticated machine learning models requires specialized knowledge and technical expertise. Small-scale farmers or those with limited technological resources may find it difficult to adopt and utilize the system effectively.
3. **Initial Investment**: The initial setup costs, including data collection, sensor deployment, and system integration, can be substantial. This financial barrier may deter adoption, particularly among farmers with limited capital.
4. **Risk of Over-Reliance**: Farmers may become overly reliant on the system's predictions, potentially reducing their responsiveness to on-the-ground observations and traditional farming knowledge. This over-reliance could limit adaptive farming practices and innovation.
5. **Ethical Considerations**: Privacy concerns may arise from the collection and use of farmer and environmental data. Ensuring data security and respecting farmer privacy rights are crucial but challenging aspects of system implementation.
6. **Technical Challenges**: System failures, data transmission errors, or algorithmic biases could lead to inaccurate predictions or operational disruptions. Continuous monitoring and maintenance are essential to mitigate these technical challenges.

**8) Conclusion**

The development and implementation of the Blueberry Yield Prediction System mark a pivotal advancement in modern agriculture, leveraging the power of machine learning and data analytics to revolutionize farming practices. By harnessing comprehensive agricultural data and advanced algorithms, this system offers farmers unprecedented insights into crop yield forecasts, soil health, and resource management. These capabilities empower farmers to make informed decisions, optimize resource allocation, and ultimately enhance productivity and profitability.

Throughout its implementation, the Blueberry Yield Prediction System has demonstrated significant benefits. It enhances yield accuracy, allowing farmers to plan harvesting schedules with precision and minimize waste. By optimizing resource usage—such as fertilizers, pesticides, and water—the system promotes sustainable farming practices, reducing environmental impact while improving crop health and resilience. Moreover, the system aids in financial planning by mitigating risks associated with market fluctuations and weather uncertainties, thereby fostering economic stability within the agricultural sector.

However, the adoption of such advanced technologies is not without challenges. The system's effectiveness heavily relies on the quality and accessibility of agricultural data, posing barriers in regions with limited data infrastructure. Moreover, the initial costs and technical expertise required for implementation may hinder small-scale farmers' adoption, necessitating supportive policies and capacity-building initiatives.

Looking forward, continued research and collaboration are crucial. Future efforts should focus on enhancing data accessibility, simplifying user interfaces, and advancing machine learning algorithms to further improve prediction accuracy and system reliability. By addressing these challenges and embracing technological advancements, the agricultural sector can achieve greater sustainability, resilience, and productivity, ensuring food security and economic prosperity for generations to come.

**9) Future Scope**

The Blueberry Yield Prediction System has laid a robust foundation for future enhancements and expansions, leveraging advancements in technology and agriculture. Moving forward, several key areas can be explored to further elevate the system's capabilities and impact.

#### Integration of Real-Time Data

Currently, the system relies on historical and seasonal data to predict blueberry yields. Integrating real-time data streams, such as weather conditions, soil moisture levels, and pest infestation alerts, would enhance prediction accuracy and responsiveness. By continuously updating predictive models with real-time inputs, farmers can receive timely insights and adjust their farming practices dynamically. This integration not only improves yield forecasts but also supports proactive decision-making, enabling farmers to mitigate risks and optimize resource usage in near real-time.

#### Expansion to Other Crops

While initially designed for blueberry cultivation, the underlying principles and predictive models of the system can be extended to other crops. Each crop has its unique growth patterns and environmental requirements, necessitating tailored prediction models. By adapting the system's algorithms and parameters to accommodate different crop types, agricultural communities can benefit from enhanced productivity and sustainability across diverse farming landscapes. This expansion could encompass popular crops in the same regions where blueberries thrive, broadening the system's applicability and scalability.

#### Integration of Advanced Machine Learning Techniques

Incorporating advanced machine learning techniques, such as deep learning algorithms, holds promise for further improving prediction accuracy. Deep learning models can autonomously learn intricate patterns and relationships within agricultural data, offering superior predictive capabilities compared to traditional machine learning approaches. By harnessing the power of neural networks, the system can decipher complex interactions between various factors influencing crop yields, including climate variability, soil composition, and genetic factors. This advancement not only refines yield forecasts but also enables predictive analytics at a more granular level, empowering farmers with detailed insights for precise decision-making.

#### Development of a User-Friendly Mobile Application

To enhance accessibility and usability, developing a dedicated mobile application for the Blueberry Yield Prediction System is essential. The application would provide farmers with intuitive interfaces to access predictive analytics, view personalized recommendations, and receive real-time updates directly on their smartphones or tablets. This mobile platform can integrate features such as interactive dashboards, push notifications for critical alerts, and decision support tools tailored to farmers' specific needs and preferences. By facilitating seamless interaction with the system, the mobile application promotes widespread adoption among farmers of varying technological proficiencies, fostering inclusive agricultural development.

In conclusion, the future scope of the Blueberry Yield Prediction System is characterized by continuous innovation and adaptation to emerging technological trends and agricultural practices. By integrating real-time data, expanding to diverse crops, incorporating advanced machine learning techniques, and developing a user-friendly mobile application, the system can unlock new possibilities for sustainable farming, resilience against climate variability, and economic prosperity within the agricultural sector. These advancements underscore the system's commitment to driving positive change and ensuring food security for global communities in the years to come.

### 10) Appendix

#### 10.1. Source Code

#### a) Model Training Code

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import dabl

data=pd.read\_csv('WildBlueberryPollinationSimulationData.csv')

data.head()

#Preprocessing

data.drop(columns=['Row#'],axis=1,inplace=True)

data.head()

data.info()

data.describe()

data.isnull().any()

data.isna().sum()

data.duplicated().sum()

data.hist(layout=(5,4), figsize=(20,15), bins=20, color='red')

plt.title('Histogram of Data')

plt.show

sns.stripplot(y=data['yield'])

dabl.plot(data, target\_col='yield', color='green', prune\_correlations\_threshold=0)

#Multivariate analysis

plt.figure(figsize=(10,8))

sns.heatmap(data.corr(),annot=True)

#Features fruitset, fruitmass, seeds have a very low correlation with other features. Hence these features are removed.

data.drop(columns=['fruitset', 'fruitmass', 'seeds'],axis=1,inplace=True)

#Removing Outliers

num\_features = len(data.columns)

cols = 3

rows = (num\_features // cols) + (num\_features % cols)

fig, axes = plt.subplots(rows, cols, figsize=(20, 20))

axes = axes.flatten()

for i, feature in enumerate(data.columns):

    sns.histplot(data[feature], kde=True, ax=axes[i])

    axes[i].set\_title(f'Distribution Plot of {feature}')

for i in range(num\_features, len(axes)):

    fig.delaxes(axes[i])

plt.tight\_layout()

plt.show()

num\_features = len(data.columns)

cols = 3

rows = (num\_features // cols) + (num\_features % cols)

fig, axes = plt.subplots(rows, cols, figsize=(12, 8))

axes = axes.flatten()

for i, feature in enumerate(data.columns):

    sns.boxplot(data=data[feature], ax=axes[i],orient='h',whis=3)

    axes[i].set\_title(f'Box Plot of {feature}')

for i in range(num\_features, len(axes)):

    fig.delaxes(axes[i])

plt.tight\_layout()

plt.show()

num\_features = len(data.columns)

cols = 3

rows = (num\_features // cols) + (num\_features % cols)

fig, axes = plt.subplots(rows, cols, figsize=(12, 8))

axes = axes.flatten()

for i, feature in enumerate(data.columns):

    sns.lineplot(data=data,x=feature,y="yield", ax=axes[i])

    axes[i].set\_title(f'Box Plot of {feature}')

for i in range(num\_features, len(axes)):

    fig.delaxes(axes[i])

plt.tight\_layout()

plt.show()

data.hist(figsize=(10,15))

Q1 = data['honeybee'].quantile(0.25)

Q3 = data['honeybee'].quantile(0.75)

IQR=Q3-Q1

lower\_limit = Q1 - 1.5 \* IQR

upper\_limit = Q3 + 1.5 \* IQR

print('lower\_limit: ',lower\_limit)

print('upper\_limit: ',upper\_limit)

data[(data.honeybee<lower\_limit)|(data.honeybee>upper\_limit)]

data = data[(data.honeybee>lower\_limit)&(data.honeybee<upper\_limit)]

data

sns.boxplot(data=data['honeybee'], orient='h',whis=3)

data.corr()

y=data['yield']

y.head()

X=data.drop(columns=['yield',],axis=1)

X.head()

#Scaling

names=X.columns

names

from sklearn.preprocessing import StandardScaler

scale = StandardScaler()

X\_scaled=scale.fit\_transform(X)

X\_scaled

X=pd.DataFrame(X\_scaled, columns=names)

X

#Train & Test Split

from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,y\_train,y\_test=train\_test\_split(X,y,test\_size=0.3,random\_state=20)

X\_train.head()

y\_train

y\_test

X\_train.shape

X\_test.shape

#Model Fitting

#1. Linear Regression

from sklearn.linear\_model import LinearRegression

lr=LinearRegression()

lr.fit(X\_train,y\_train)

y\_lrpred=lr.predict(X\_test)

from sklearn.metrics import r2\_score, mean\_squared\_error, mean\_absolute\_error

r2\_lr=r2\_score(y\_lrpred,y\_test)

mse\_lr = mean\_squared\_error(y\_test,y\_lrpred)

mae\_lr = mean\_absolute\_error(y\_test,y\_lrpred)

rmse\_lr = np.sqrt(mse\_lr)

print("Mean Absolute Error: ", mae\_lr)

print("Root Mean Squared Error: ", rmse\_lr)

print("R2: ", r2\_lr\*100)

lr.coef\_

lr.intercept\_

#2. Decision Tree

from sklearn.tree import DecisionTreeRegressor

dt=DecisionTreeRegressor(criterion="squared\_error", random\_state=42, max\_depth=5)

dt.fit(X\_train,y\_train)

y\_dtpred=dt.predict(X\_test)

r2\_dt=r2\_score(y\_dtpred,y\_test)

mse\_dt = mean\_squared\_error(y\_test,y\_dtpred)

mae\_dt = mean\_absolute\_error(y\_test,y\_dtpred)

rmse\_dt = np.sqrt(mse\_dt)

print("Mean Absolute Error: ", mae\_dt)

print("Root Mean Squared Error: ", rmse\_dt)

print("R2: ", r2\_dt\*100)

#3. Random Forest

from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(n\_estimators=100, random\_state=42, max\_depth=5)

rf.fit(X\_train,y\_train)

y\_rfpred=rf.predict(X\_test)

r2\_rf=r2\_score(y\_rfpred,y\_test)

mse\_rf = mean\_squared\_error(y\_test,y\_rfpred)

mae\_rf = mean\_absolute\_error(y\_test,y\_rfpred)

rmse\_rf = np.sqrt(mse\_rf)

print("Mean Absolute Error: ", mae\_rf)

print("Root Mean Squared Error: ", rmse\_rf)

print("R2: ", r2\_rf\*100)

#4. XGBoost

import xgboost as xgb

xg = xgb.XGBRegressor(objective='reg:squarederror', max\_depth=5, n\_estimators=100, learning\_rate=0.1)

xg.fit(X\_train,y\_train)

y\_xgpred=xg.predict(X\_test)

r2\_xg=r2\_score(y\_xgpred,y\_test)

mse\_xg = mean\_squared\_error(y\_test,y\_xgpred)

mae\_xg = mean\_absolute\_error(y\_test,y\_xgpred)

rmse\_xg = np.sqrt(mse\_xg)

print("Mean Absolute Error: ", mae\_xg)

print("Root Mean Squared Error: ", rmse\_xg)

print("R2: ", r2\_xg\*100)

#5. SVM Regression

from sklearn.svm import SVR

sv = SVR(kernel='linear')

sv.fit(X\_train,y\_train)

y\_svpred=sv.predict(X\_test)

r2\_sv=r2\_score(y\_svpred,y\_test)

mse\_sv = mean\_squared\_error(y\_test,y\_svpred)

mae\_sv = mean\_absolute\_error(y\_test,y\_svpred)

rmse\_sv = np.sqrt(mse\_sv)

print("Mean Absolute Error: ", mae\_sv)

print("Root Mean Squared Error: ", rmse\_sv)

print("R2: ", r2\_sv\*100)

#\*Hyper Parameter Tuning for SVM\*

from sklearn.model\_selection import GridSearchCV

svr = SVR(kernel='linear')

param\_grid = {

    'C': [0.1, 1, 10, 100],

    'epsilon': [0.001, 0.01, 0.1, 1],

    'gamma': ['scale', 'auto']

}

grid\_search = GridSearchCV(estimator=svr, param\_grid=param\_grid, scoring='neg\_mean\_squared\_error', cv=5)

grid\_search.fit(X\_train, y\_train)

print("Best hyperparameters: ", grid\_search.best\_params\_)

print("Best score: ", grid\_search.best\_score\_)

best\_model = grid\_search.best\_estimator\_

y\_pred = best\_model.predict(X)

mse = mean\_squared\_error(y, y\_pred)

print("Mean Squared Error: ", mse)

sv2= SVR(C=100, epsilon=0.001, gamma='auto', kernel='linear')

sv2.fit(X\_train,y\_train)

y\_sv2pred=sv2.predict(X\_test)

r2\_sv2=r2\_score(y\_sv2pred,y\_test)

mse\_sv2 = mean\_squared\_error(y\_test,y\_sv2pred)

mae\_sv2 = mean\_absolute\_error(y\_test,y\_sv2pred)

rmse\_sv2 = np.sqrt(mse\_sv2)

print("Mean Absolute Error: ", mae\_sv2)

print("Root Mean Squared Error: ", rmse\_sv2)

print("R2: ", r2\_sv2\*100)

#Model Evaluvation

model\_eval\_rec = {

    'Linear Regression': {

        'mae': mae\_lr,

        'rmse': rmse\_lr,

        'r2': r2\_lr \* 100

    },

    'Random Forest': {

        'mae': mae\_rf,

        'rmse': rmse\_rf,

        'r2': r2\_rf \* 100

    },

    'Decision Tree': {

        'mae': mae\_dt,

        'rmse': rmse\_dt,

        'r2': r2\_dt \* 100

    },

    'XGBoost': {

        'mae': mae\_xg,

        'rmse': rmse\_xg,

        'r2': r2\_xg \* 100

    },

    'SVM': {

        'mae': mae\_sv2,

        'rmse': rmse\_sv2,

        'r2': r2\_sv2 \* 100

    }

}

pd.DataFrame(model\_eval\_rec).plot(kind="bar", color=[

    sns.color\_palette("pastel")[1],

    sns.color\_palette("pastel")[2],

    sns.color\_palette("pastel")[3],

    sns.color\_palette("pastel")[4],

    sns.color\_palette("pastel")[5]

]);

#XGBoost Model is choosen as R2 Score is higher than other models.

#Hyperparameter Tuning for XGBoost Model

#a. Using Grid Search

import xgboost as xgb

from sklearn.model\_selection import GridSearchCV

param\_grid = {

    'n\_estimators': [100, 200, 300],

    'learning\_rate': [0.01, 0.1, 0.2],

    'max\_depth': [3, 5, 7],

    'min\_child\_weight': [1, 3, 5],

    'subsample': [0.8, 1.0],

    'colsample\_bytree': [0.8, 1.0]

}

xgb\_reg = xgb.XGBRegressor(objective='reg:squarederror')

grid\_search = GridSearchCV(estimator=xgb\_reg, param\_grid=param\_grid, cv=5, scoring='neg\_mean\_squared\_error', verbose=1, n\_jobs=-1)

grid\_search.fit(X\_train, y\_train)

print("Best parameters found: ", grid\_search.best\_params\_)

print("Lowest RMSE found: ", (-grid\_search.best\_score\_)\*\*0.5)

gs\_best\_model = grid\_search.best\_estimator\_

y\_xg2pred=gs\_best\_model.predict(X\_test)

r2\_xg2=r2\_score(y\_xg2pred,y\_test)

mse\_xg2 = mean\_squared\_error(y\_test,y\_xg2pred)

mae\_xg2 = mean\_absolute\_error(y\_test,y\_xg2pred)

rmse\_xg2 = np.sqrt(mse\_xg2)

print("Mean Absolute Error: ", mae\_xg2)

print("Root Mean Squared Error: ", rmse\_xg2)

print("R2: ", r2\_xg2\*100)

#b. Using Random Search

from sklearn.model\_selection import RandomizedSearchCV

param\_dist = {

    'n\_estimators': [int(x) for x in np.linspace(start=100, stop=500, num=10)],

    'learning\_rate': [0.001, 0.01, 0.1, 0.2],

    'max\_depth': [int(x) for x in np.linspace(3, 10, num=8)],

    'min\_child\_weight': [1, 3, 5, 7],

    'subsample': [0.6, 0.8, 1.0],

    'colsample\_bytree': [0.6, 0.8, 1.0]

}

xgb\_reg = xgb.XGBRegressor(objective='reg:squarederror')

random\_search = RandomizedSearchCV(estimator=xgb\_reg, param\_distributions=param\_dist, n\_iter=100, cv=5, verbose=1, random\_state=42, n\_jobs=-1, scoring='neg\_mean\_squared\_error')

random\_search.fit(X\_train, y\_train)

print("Best parameters found: ", random\_search.best\_params\_)

print("Lowest RMSE found: ", (-random\_search.best\_score\_)\*\*0.5)

rs\_best\_model = random\_search.best\_estimator\_

y\_xg3pred=rs\_best\_model.predict(X\_test)

r2\_xg3=r2\_score(y\_xg3pred,y\_test)

mse\_xg3 = mean\_squared\_error(y\_test,y\_xg3pred)

mae\_xg3 = mean\_absolute\_error(y\_test,y\_xg3pred)

rmse\_xg3 = np.sqrt(mse\_xg3)

print("Mean Absolute Error: ", mae\_xg3)

print("Root Mean Squared Error: ", rmse\_xg3)

print("R2: ", r2\_xg3\*100)

#Random Search gives the best model with highest R2 Score.

best\_model=rs\_best\_model

y\_pred=best\_model.predict(X\_test)

r2=r2\_score(y\_pred,y\_test)

mse = mean\_squared\_error(y\_test,y\_pred)

mae = mean\_absolute\_error(y\_test,y\_pred)

rmse = np.sqrt(mse)

print("Mean Absolute Error: ", mae)

print("Root Mean Squared Error: ", rmse)

print("R2: ", r2\*100)

#Saving the model

import pickle

with open('best\_model.pkl', 'wb') as file:

    pickle.dump(best\_model, file)

#Saving the scaler

with open('standard\_scaler.pkl', 'wb') as file:

    pickle.dump(scale, file)

#### b) Model Testing Code

#Testing the Saved Model

import pickle

import pandas as pd

import xgboost

with open('best\_model.pkl', 'rb') as file:

    loaded\_model = pickle.load(file)

with open('standard\_scaler.pkl', 'rb') as file:

    loaded\_scaler = pickle.load(file)

names=['clonesize', 'honeybee', 'bumbles', 'andrena', 'osmia',

       'MaxOfUpperTRange', 'MinOfUpperTRange', 'AverageOfUpperTRange',

       'MaxOfLowerTRange', 'MinOfLowerTRange', 'AverageOfLowerTRange',

       'RainingDays', 'AverageRainingDays']

X\_new=[[37.5,0.75,0.2,0.25,0.25,86.0,52.0,71.9,62.0,30.0,50.8,16.0,0.26]]

X\_new=pd.DataFrame(X\_new, columns=names)

X\_new=loaded\_scaler.transform(X\_new)

X\_new

X\_new=pd.DataFrame(X\_new, columns=names)

print(loaded\_model.predict(X\_new)[0])

#### c) Flask Code

from flask import Flask, render\_template, request

import pickle

import pandas as pd

import xgboost

with open('best\_model.pkl', 'rb') as file:

    model = pickle.load(file)

with open('standard\_scaler.pkl', 'rb') as file:

    scaler = pickle.load(file)

app = Flask(\_\_name\_\_)

@app.route('/')

def loadpage():

    return render\_template('index.html')

@app.route('/y\_predict', methods=['POST'])

def prediction():

    data = {

        "clonesize": float(request.form["clonesize"]),

        "honeybee": float(request.form["honeybee"]),

        "bumbles": float(request.form["bumbles"]),

        "andrena": float(request.form["andrena"]),

        "osmia": float(request.form["osmia"]),

        "MaxOfUpperTRange": float(request.form["MaxOfUpperTRange"]),

        "MinOfUpperTRange": float(request.form["MinOfUpperTRange"]),

        "AverageOfUpperTRange": float(request.form["AverageOfUpperTRange"]),

        "MaxOfLowerTRange": float(request.form["MaxOfLowerTRange"]),

        "MinOfLowerTRange": float(request.form["MinOfLowerTRange"]),

        "AverageOfLowerTRange": float(request.form["AverageOfLowerTRange"]),

        "RainingDays": float(request.form["RainingDays"]),

        "AverageRainingDays": float(request.form["AverageRainingDays"]),

    }

    names=['clonesize', 'honeybee', 'bumbles', 'andrena', 'osmia',

       'MaxOfUpperTRange', 'MinOfUpperTRange', 'AverageOfUpperTRange',

       'MaxOfLowerTRange', 'MinOfLowerTRange', 'AverageOfLowerTRange',

       'RainingDays', 'AverageRainingDays']

    data\_df = pd.DataFrame([data])

    data\_df=pd.DataFrame(data\_df, columns=names)

    data\_scaled = scaler.transform(data\_df)

    data\_pred=pd.DataFrame(data\_scaled, columns=names)

    prediction = model.predict(data\_pred)[0]

    return render\_template('index.html', prediction\_text=f"Predicted Yield: {prediction:.2f} kg/ha")

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(debug=False)

#### c) HTML Code

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Wild Blueberry Yield Prediction</title>

    <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;700&display=swap" rel="stylesheet">

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

    <link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

</head>

<body>

    <div class="container">

        <h1>Wild Blueberry Yield Prediction</h1>

        <form action="/y\_predict" method="post" class="needs-validation" novalidate>

            <div class="form-row">

                <div class="form-group col-md-6">

                    <label for="clonesize">Clone size (m²)</label>

                    <input type="text" class="form-control" id="clonesize" name="clonesize" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid clone size.</div>

                </div>

                <div class="form-group col-md-6">

                    <label for="honeybee">Honeybee (bees/m²/min)</label>

                    <input type="text" class="form-control" id="honeybee" name="honeybee" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid honeybee count.</div>

                </div>

            </div>

            <div class="form-row">

                <div class="form-group col-md-6">

                    <label for="bumbles">Bumbles (bees/m²/min)</label>

                    <input type="text" class="form-control" id="bumbles" name="bumbles" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid bumbles count.</div>

                </div>

                <div class="form-group col-md-6">

                    <label for="andrena">Andrena (bees/m²/min)</label>

                    <input type="text" class="form-control" id="andrena" name="andrena" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid andrena count.</div>

                </div>

            </div>

            <div class="form-row">

                <div class="form-group col-md-6">

                    <label for="osmia">Osmia (bees/m²/min)</label>

                    <input type="text" class="form-control" id="osmia" name="osmia" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid osmia count.</div>

                </div>

                <div class="form-group col-md-6">

                    <label for="MaxOfUpperTRange">Max of Upper T Range (℃)</label>

                    <input type="text" class="form-control" id="MaxOfUpperTRange" name="MaxOfUpperTRange" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid maximum upper temperature range.</div>

                </div>

            </div>

            <div class="form-row">

                <div class="form-group col-md-6">

                    <label for="MinOfUpperTRange">Min of Upper T Range (℃)</label>

                    <input type="text" class="form-control" id="MinOfUpperTRange" name="MinOfUpperTRange" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid minimum upper temperature range.</div>

                </div>

                <div class="form-group col-md-6">

                    <label for="AverageOfUpperTRange">Average of Upper T Range (℃)</label>

                    <input type="text" class="form-control" id="AverageOfUpperTRange" name="AverageOfUpperTRange" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid average upper temperature range.</div>

                </div>

            </div>

            <div class="form-row">

                <div class="form-group col-md-6">

                    <label for="MaxOfLowerTRange">Max of Lower T Range (℃)</label>

                    <input type="text" class="form-control" id="MaxOfLowerTRange" name="MaxOfLowerTRange" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid maximum lower temperature range.</div>

                </div>

                <div class="form-group col-md-6">

                    <label for="MinOfLowerTRange">Min of Lower T Range (℃)</label>

                    <input type="text" class="form-control" id="MinOfLowerTRange" name="MinOfLowerTRange" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid minimum lower temperature range.</div>

                </div>

            </div>

            <div class="form-row">

                <div class="form-group col-md-6">

                    <label for="AverageOfLowerTRange">Average of Lower T Range (℃)</label>

                    <input type="text" class="form-control" id="AverageOfLowerTRange" name="AverageOfLowerTRange" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid average lower temperature range.</div>

                </div>

                <div class="form-group col-md-6">

                    <label for="RainingDays">Raining Days (Day)</label>

                    <input type="text" class="form-control" id="RainingDays" name="RainingDays" required pattern="^\d\*\.?\d+$" />

                    <div class="invalid-feedback">Please enter a valid number of raining days.</div>

                </div>

            </div>

            <div class="form-group">

                <label for="AverageRainingDays">Average Raining Days (Day)</label>

                <input type="text" class="form-control" id="AverageRainingDays" name="AverageRainingDays" required pattern="^\d\*\.?\d+$" />

                <div class="invalid-feedback">Please enter a valid number of average raining days.</div>

            </div>

            <button type="submit" class="btn btn-warning btn-block">Predict</button>

        </form>

        {% if prediction\_text %}

        <div class="alert alert-info mt-4" id="predictionResult">

            <h2>{{ prediction\_text }}</h2>

        </div>

        {% endif %}

    </div>

    <script>

        (function() {

            'use strict';

            window.addEventListener('load', function() {

                var forms = document.getElementsByClassName('needs-validation');

                Array.prototype.filter.call(forms, function(form) {

                    form.addEventListener('submit', function(event) {

                        if (form.checkValidity() === false) {

                            event.preventDefault();

                            event.stopPropagation();

                        }

                        form.classList.add('was-validated');

                    }, false);

                });

            }, false);

            {% if prediction\_text %}

            window.addEventListener('load', function() {

                document.getElementById('predictionResult').scrollIntoView();

            });

            {% endif %}

        })();

    </script>

</body>

</html>

#### d) CSS Code

body {

    margin: 0;

    padding: 0;

    font-family: 'Roboto', sans-serif;

    background: url('blue.jpg') no-repeat center center fixed;

    background-size: cover;

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh;

    color: #fff;

    overflow-y: auto;

}

.container {

    background: rgba(0, 0, 0, 0.7);

    padding: 30px;

    border-radius: 10px;

    box-shadow: 0 0 15px rgba(0, 0, 0, 0.5);

    text-align: center;

    max-height: 90vh;

    overflow-y: auto;

    display: flex;

    flex-direction: column;

    align-items: center;

}

h1 {

    margin-bottom: 20px;

    font-size: 2.5em;

    color: #f0ad4e;

    font-weight: 700;

}

form {

    width: 100%;

}

.form-row {

    display: flex;

    flex-wrap: wrap;

    justify-content: space-between;

}

.form-group {

    margin-bottom: 15px;

    width: 48%;

}

.form-group label {

    color: #f0ad4e;

    font-weight: 500;

}

.form-group input {

    padding: 10px;

    border: none;

    border-radius: 5px;

    font-size: 1em;

    margin-top: 5px;

    width: 100%;

}

.btn {

    padding: 10px;

    border: none;

    border-radius: 5px;

    font-size: 1.2em;

    cursor: pointer;

    background-color: #f0ad4e;

    color: #fff;

    margin-top: 20px;

    transition: background-color 0.3s ease;

    width: 100%;

}

.btn:hover {

    background-color: #ec971f;

}

.alert {

    margin-top: 20px;

    padding: 20px;

    background: rgba(0, 0, 0, 0.8);

    border-radius: 5px;

    box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);

    text-align: center;

    width: 100%;

}

.alert h2 {

    margin: 0;

    font-size: 1.5em;

    color: #f0ad4e;

}

#### 10.2. GitHub & Project Demo Link