Impact of Climate and Environment Factors on Crop Yields in the World

We give consent for this to be used as a teaching resource.

Executive Summary

In this project, it was established that Machine Learning can effectively predict crop yields based on historical climate and environmental data. Several Machine Learning models were tested which included Ensemble Methods, Neural Network and Standard Regression. These were subsequently evaluated based on metrics such as MSE, MAE, R^2 and RMSE.

Results found Neural Network to be the best performing out of all the models tested. However, Bagged Decision Tree was ultimately selected as it has better interpretability and transparency.

There are certain limitations to the model, which involves estimating future environmental predictors like rainfall and temperature. Currently it is done using a simple linear regression, but more sophisticated models will need to be explored for a more accurate prediction.

Heat map and time lapse forecasts were generated and showed changing crop yields across different regions across time. There were more drastic changes observed in Top 10 Developing Countries compared to Top 10 Developed Countries.

The Food Security Index was calculated as a critical metric to assess whether there will be a sufficient food supply to meet human consumption needs by 2030, aligning with the United Nations Global Sustainable Development Goals (specifically Goal 2).

Results indicated that the goal is on track to being met, but not in all food classes. For example, there will be an abundance of potatoes but not rice. There could potentially be an opportunity for countries to assist each other, especially those with surplus. This will require a coordinated effort between all countries, and the United Nations will have an important role to play in this.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Problem Definition

The global agriculture landscape plays an important role in the 21st century. Today there are 8 billion of us in the world, with the human population expected to grow at a rapid pace until the end of the century. Agriculture, as the bedrock of food security and a pivotal contributor to global economies, faces an unprecedented era of uncertainty.

The challenge of ensuring food security and eradicating hunger is foundational to the attainment of global sustainable development goals. However, the growing sensitivity of food production to the impacts of climate and environmental factors present a challenge. The intricate relationships between rising temperatures, shifting precipitation patterns, pesticides, greenhouse gas emissions and other environmental factors have brought forth a pressing question – How will the world's agricultural systems adapt and respond to these multifaceted challenges?

Answering this question through predicting crop yields for the Top 10 produce in the world just might help address the challenges ahead.

The problem is so significant that it was foundational to the United Nations Global Sustainable Development Goals - which aims to 'end hunger, achieve food security and improved nutrition and promote sustainable agriculture' by 2030 (United Nations 2023).

The outcome of this research hopes to inform agricultural practices, guide policy formulation, prepare us against the uncertainties of the future, ultimately contributing to the welfare of nations and the preservation of our planet.

1.2 Research Questions

There are several questions that the group will explore further in this project:

- Can Machine Learning effectively predict crop yields based on historical climate and environmental data?
- Do variations in temperature and precipitation patterns correlate with changes in crop yields across different regions and time periods?
- Are there any countries that are at risk of food security by 2030?

The answers will be presented in Section 4 on Main Findings.

2. Methodology

2.1 Overview

The analysis in this project report will follow the 5 Step Data Science process (Figure 1). This includes the following key steps:

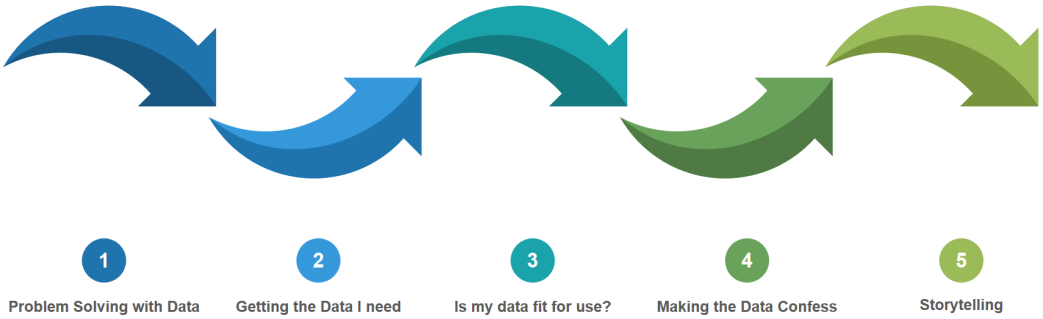


Figure 1: Data Science Process Flowchart

Note that the process is used as a guide. The project went through several rounds of iterations and content may overlap across the different steps. A greater emphasis is placed on the Machine Learning aspect of the project as this is the main learning objective of the course.

2.2 Data Source

There are several data sources used in the project as listed below:

- **Food and Agriculture Organisation (FAO):** http://www.fao.org/home/en/
  Comprehensive repository of agricultural and climate-related data by country. This includes historical crop yield data, information on crop types, climate variables, pesticides, insecticides, and geographical information.

- **World Bank - Country Climate and Development Report (CCDR):**
  https://databank.worldbank.org/
  Wealth of data related to climate, development, population growth, and human food consumption by country. Greenhouse gas emissions come from this source and would require data processing.

These data sources are selected based on their reputation to contain reliable datasets that will help inform any decision making.

2.3 Data Cleaning

Data cleaning is a critical step in any data analysis process. Our data sources are primarily drawn from multiple datasets on crop yields and environmental indicators that help us understand agricultural trends and environmental impacts in different countries and regions. Our final dataset was merged from these data sources, and it contains 50,591 rows and 26 columns. These columns cover everything from country names to yields of different crops, as well as environmentally relevant indicators. In order to maintain consistency in the data, we focus on the 1990-2013 period, as this is the common time period for multiple datasets.

When dealing with missing values, we find that only the column "Temperature_Change(Degree Celsius)" has 365 missing entries for 1990-2013. Considering the time-series nature of the data, we decided to use forward and backward padding. This approach considers the autocorrelation of the time series data, i.e., the continuity of the previous and subsequent observations in the time series, which justifies the use of these observations to fill in the missing values.

Further, in order to obtain a comprehensive and integrated dataset, we merged different datasets based on country names or country codes. In terms of data conversion, we used the One-Hot Encoding method to convert textual data to numerical data to facilitate modelling and analysis.

Finally, to improve the readability of the dataset, we changed some of the column names. For example, we changed Value to Yield_hg/ha and changed Pesticides (total) | 00001357 || Use per area of cropland | 005159 || Kilograms per hectare to PesticidesTotal_kg/ha. This change simplifies the column names while ensuring that the data are descriptive and complete.

After this series of data cleaning and preprocessing steps, we now have a complete, consistent, and accurate dataset suitable for in-depth machine learning modelling and analysis (Figure 2).

	Country Name	Item_Maize	Item_Plantains	Year	Yield_hg/ha	CH4_kt	CO2_kt	AvgPrecipitation_mm/year	AvgTemp_DegC
0	Angola	0.0	0.0	1990	7995.0	4668.7865	51537.5257	1010.0	24.12
1	Angola	0.0	0.0	1991	8000.0	4678.9787	51531.9568	1010.0	24.02
2	Angola	0.0	0.0	1992	8000.0	4690.8651	51476.6432	1010.0	23.96
3	Angola	0.0	0.0	1993	8862.0	4676.1835	51487.3957	1010.0	24.15
4	Angola	0.0	0.0	1994	10000.0	4681.5529	51576.0095	1010.0	24.04

Figure 2: Sub-sample of Final Dataset

2.4 Exploratory Data Analysis

In our initial exploration of the data, the dataset included records from 1990 to 2013, and the time series of crop yields over this period exhibited non-stationarity, as further evidenced by a p-value significantly greater than 0.05 (Figure 3).

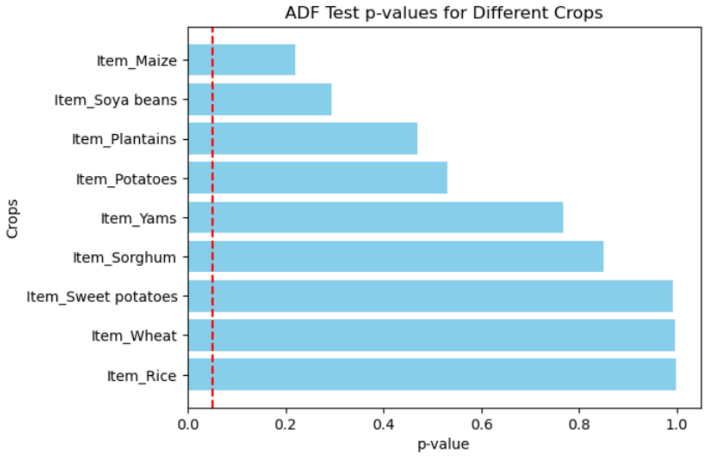


Figure 3: ADF Test P-Values for Different Crops

As we delve deeper into the relationship between agriculture and the environment, we observe a significant positive correlation between greenhouse gases, which coincides with the relationship between agricultural production and greenhouse gas emissions. There is evidently some synchronisation between pesticide use and GHG emissions, which further emphasises the strong link between agriculture and the environment (Figure 4).

However, things are not always so obvious. While precipitation has some effect on agricultural production, the relationship with other factors, such as GHG emissions, is relatively weak. This may mean that precipitation is not the main driver dominating changes in GHG emissions and other environmental factors. Similarly, temperature, although it has a significant effect on crop growth, has a low correlation with other environmental indicators, suggesting that temperature changes are not the main driver of changes in other environmental variables (Figure 4).

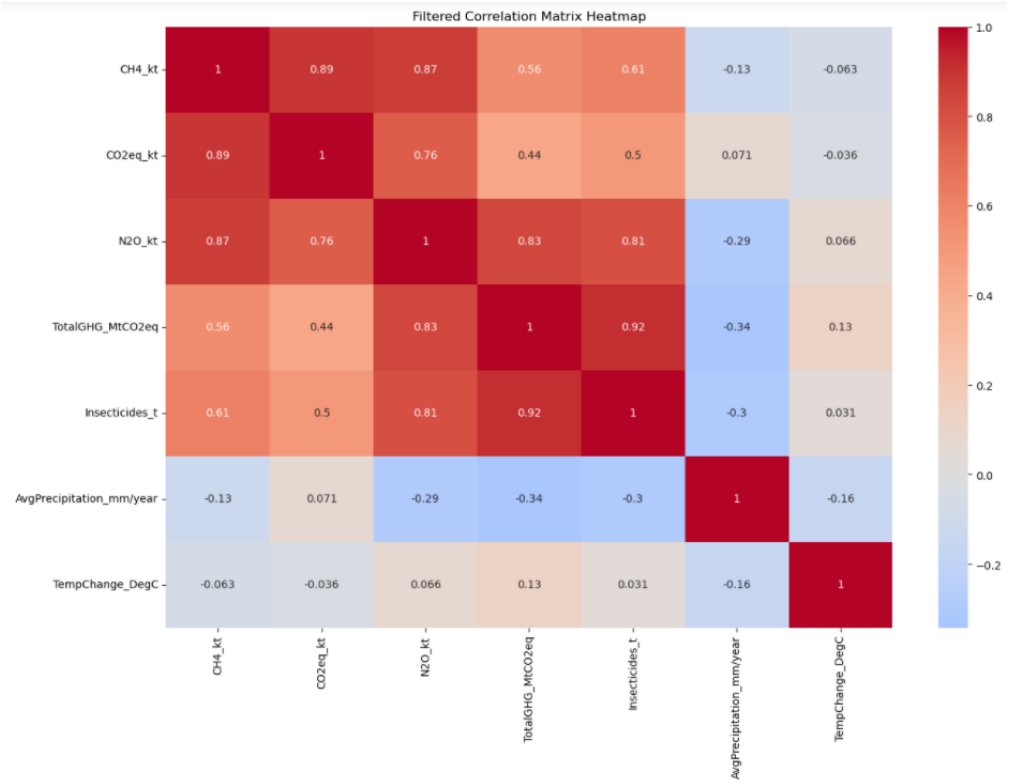


Figure 4: Filtered Correlation Matrix Heatmap

As for the distribution of covariates, it is observed that most countries maintain a relatively low level of GHG emissions, but there are a few countries that have significantly higher than average emissions. This may be related to their level of industrialization, geographical location, or specific agricultural practices. At the same time, the distribution of mean temperatures is close to normal, indicating that most countries are affected by the generalised effects of global climate change. This emphasises the pervasive nature of global climate change and its pervasive impact on countries. The use of pesticides and insecticides is relatively low in most countries, but there are some notable outliers, which may imply that certain countries use more pesticides and insecticides due to their larger agricultural areas, higher yields or more pest threats.

In addition, the data on average annual precipitation are right skewed, which suggests that most of the countries are at a low level, but there are some countries that have exceptionally high levels of precipitation, which may be related to their geographic location and climate type.

Overall, our analysis provides valuable insights for further research and highlights the close relationship between agriculture and the environment (Figure 5).

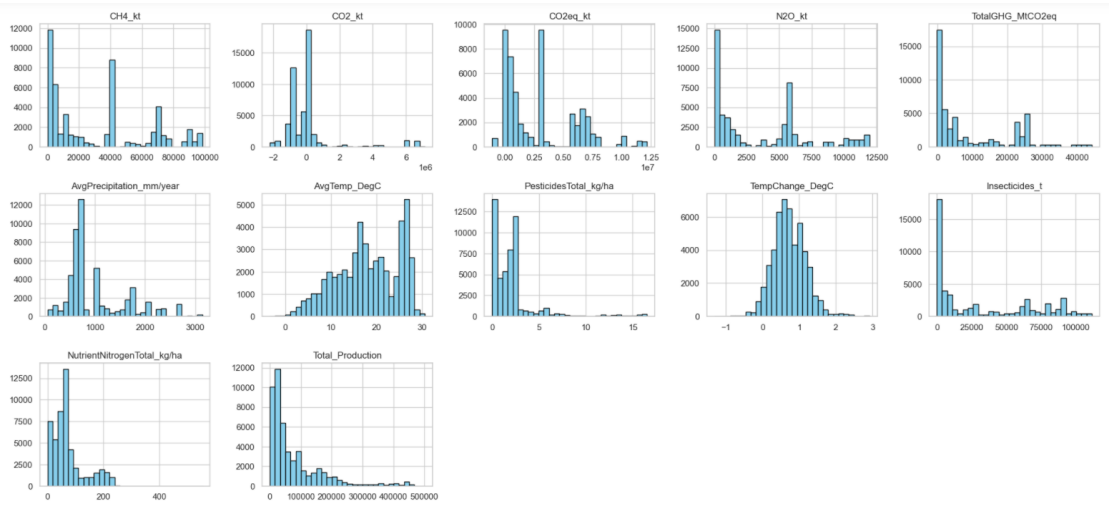


Figure 5: Distribution of Covariates

Overall, our analysis reveals the complex relationship between agriculture and the environment. This not only enhances our understanding of the interactions between agricultural practices and environmental change, but also points us in the direction of subsequent machine learning modelling analyses, emphasising the importance of considering these interrelationships in model selection, feature engineering, and data preprocessing to ensure more accurate and insightful predictions.

3. Model Selection

3.1 Linear and Polynomial Regression

As a starting point, linear regression and polynomial regression were used to find the most suitable model for our project. It was found that polynomial model performed better on both the test and training datasets. The polynomial model had an R^2 value of 0.854, which means that it was able to capture 85.4% of the variation in the data, compared to only 67.2% for the linear model. The error metrics such as Mean Square Error (MSE), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE) also verify that the polynomial model has a higher prediction accuracy. These statistics clearly indicate that the non-linear relationships present in the data are better captured by the polynomial model (Table 1).

Table 1: Evaluation Metric Table - Linear and Polynomial Regression

Metric	Linear Regression (Test)	Polynomial Regression (Test)	Linear Regression (Train)	Polynomial Regression (Train)
MSE	2.743×10^9	1.220×10^9	2.721×10^9	1.215×10^9
RMSE	5.237×10^4	3.492×10^4	$5.217x\ 10^4$	3.485×10^4
MAE	3.545×10^4	2.260×10^4	3.553×10^4	2.262×10^4
R^2	0.672	0.854	0.674	0.854

Nonetheless, we still need to be wary of the potential risk of overfitting, especially given the choice of higher order polynomial models. Fortunately, in our experiments, the MSE of the 2nd-order polynomial model on cross-validation is about 1.232×10^9, which is very close to the MSE of 1.221×10^9 on the test data, suggesting that the model performs relatively stable on both the training subset and the test data, with no obvious signs of overfitting (Figure 6).

```python
from sklearn.model_selection import KFold, cross_val_score
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Define a function for cross-validation
def evaluate_poly_degree(degree, X, y, n_splits=5):
    # Create a pipeline that first applies polynomial transformation and then linear regression
    model = make_pipeline(PolynomialFeatures(degree=degree), LinearRegression())
    # Use cross-validation to get the negative mean squared error for each split
    scores = cross_val_score(model, X, y, cv=KFold(n_splits=n_splits, shuffle=True, random_state=42),
                             scoring='neg_mean_squared_error')
    return -scores.mean()  # Return the positive MSE

# Cross-validate the 2-degree polynomial model
degree = 2
mse = evaluate_poly_degree(degree, X_train, y_train)

# Train the 2-degree polynomial model on the entire training data and evaluate on the test set
model = make_pipeline(PolynomialFeatures(degree=degree), LinearRegression())
model.fit(X_train, y_train)
predictions = model.predict(X_test)

# Calculate MSE on the test set
mse_test = mean_squared_error(y_test, predictions)

mse, mse_test
```

```
(1231606621.6227875, 1219750101.0076404)
```

Figure 6: Code Snippet for Cross Validation

However, it can be observed from the actual vs predicted plots (Figure 7) that the predictions of the linear model are relatively dispersed, while the polynomial model, although somewhat closer to the actual values, also suffers from significant deviations. The residual plots (Figure 8) further reveal some pattern in both models, implying that the model errors are not entirely random, but rather somehow fail to capture the complex relationship between climate, environmental factors, and crop yield.

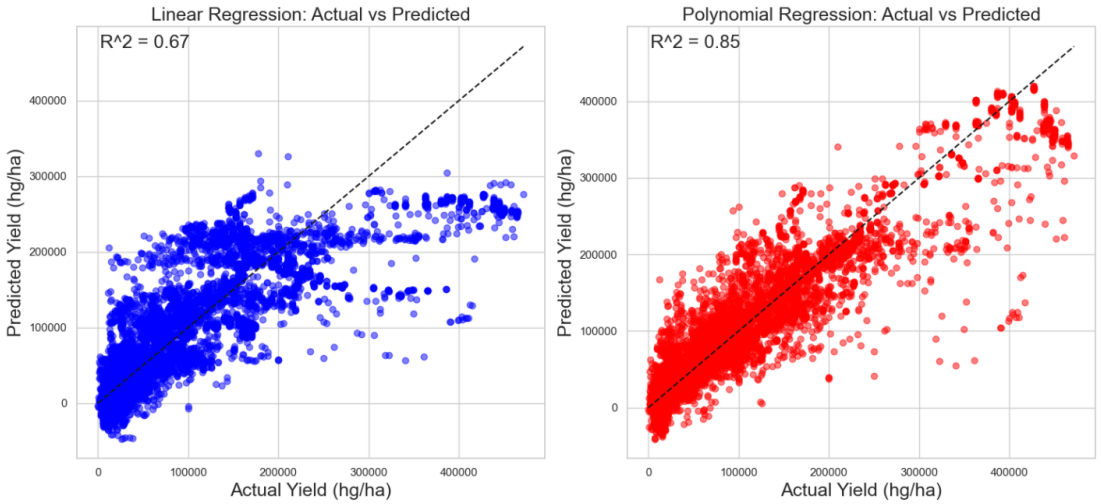


Figure 7: Actual vs Predicted Yield for Linear and Polynomial Regression

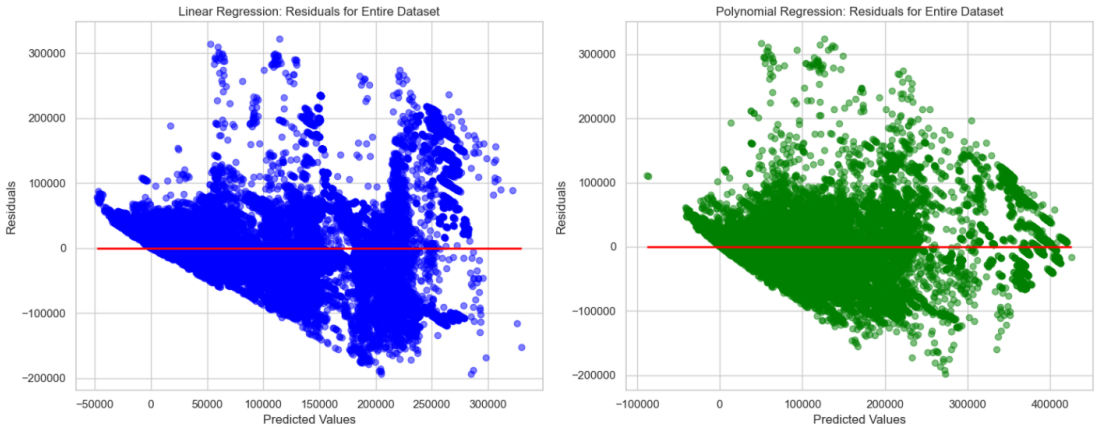


Figure 8: Residual vs. Predicted Yield for Linear and Polynomial Regression

These findings further emphasise the key message that although polynomial regression outperforms linear regression in some ways, we may need to explore other, more sophisticated modelling approaches to capture the information and patterns more accurately in the data.

3.2 Random Forest

Random Forest is a powerful and versatile ensemble learning technique widely used in machine learning for both classification and regression tasks. It is particularly popular for its effectiveness in predictive modelling and its ability to handle large

datasets with high dimensionality (Yiu 2019). In this project, we utilised Random Forest as one of the candidate models to address the critical issue of predicting crop yield, which plays a pivotal role in alleviating global hunger.

The keys steps for random forest are importing libraries, feature selection, feature scaling, data splitting, hyperparameter tuning, model creation and model training, model evaluation, feature importance, and data visualisation.

Features were then selected for the model from the dataset as shown in Figure 9. “Yield_hg/ha” is defined as our target variable, and the rest as features.

```
target = 'Yield_hg/ha'
features = [
'Item_Maize', 'Item_Plantains', 'Item_Potatoes', 'Item_Rice', 'Item_Sorghum','Item_Soya beans',
'Item_Sweet potatoes', 'Item_Wheat', 'Item_Yams', 'CH4_kt', 'CO2_kt', 'CO2eq_kt', 'N2O_kt',
'TotalGHG_MtCO2eq', 'AvgPrecipitation_mm/year','AvgTemp_DegC', 'PesticidesTotal_kg/ha', 'TempChange_DegC',
'Insecticides_t','NutrientNitrogenTotal_kg/ha']
```

Figure 9: Feature Selection for Random Forest

Before splitting the data into training and testing sets, we applied feature scaling using the StandardScaler. This step is crucial to ensure that all features have a mean of 0 and a standard deviation of 1. By standardising the features, we made sure that they are on a consistent scale, preventing any undue influence of features with larger scales on the model's performance. This prepares the data for our Random Forest model, contributing to its robustness and accuracy in predicting crop yield, a key element in addressing global hunger and food security. Then the data was split into training and testing sets with an 80-20 ratio, using the train_test_split function. This allowed us to train the model on a subset of the data and evaluate its performance on unseen data.

Hyperparameter Tuning

Hyperparameter tuning was performed to optimise the Random Forest model using randomsearchcv. The following hyperparameters were tuned:

- n_estimators: The number of trees in the forest.

- max_depth: The maximum depth of each tree.

The Random Forest model was subsequently created and trained using the optimal hyperparameters. The model was built with 167 estimators and a maximum depth of 20. The code for hyperparameter tuning and model creation are as follows:

```
param_dist = {
    'n_estimators': randint(100, 200),
    'max_depth': [None, 10, 20, 30],
}

random_search = RandomizedSearchCV(estimator=random_forest, param_distributions=param_dist,
                                   scoring='neg_mean_squared_error', cv=5, n_iter=10)
random_search.fit(X_train, y_train)
best_random_forest = random_search.best_estimator_
```

Figure 10: Hyperparameter Tuning for Random Forest

Evaluation

Now, we evaluated the model using 4 evaluators, there are mean squared error (MSE), mean absolute error (MAE), R-squared (R2) score, and root mean squared error (RMSE). We performed the evaluation on both the training set and testing set, the model evaluation metrics will be compared with other models later (Table 2).

Table 2: Evaluation Metric Table - Random Forest

Evaluator	Training set metrics	Testing set metrics
MSE	136,513,658.55	324,103,259.19
MAE	4,601.56	7,223.89
R2	0.9836	0.9612
RMSE	11,683.91	18,002.87

Figure 11 shows the actual vs predicted plot for random forest. The plot shows a promising result as the predicted data matches actual data, even at high crop yields.

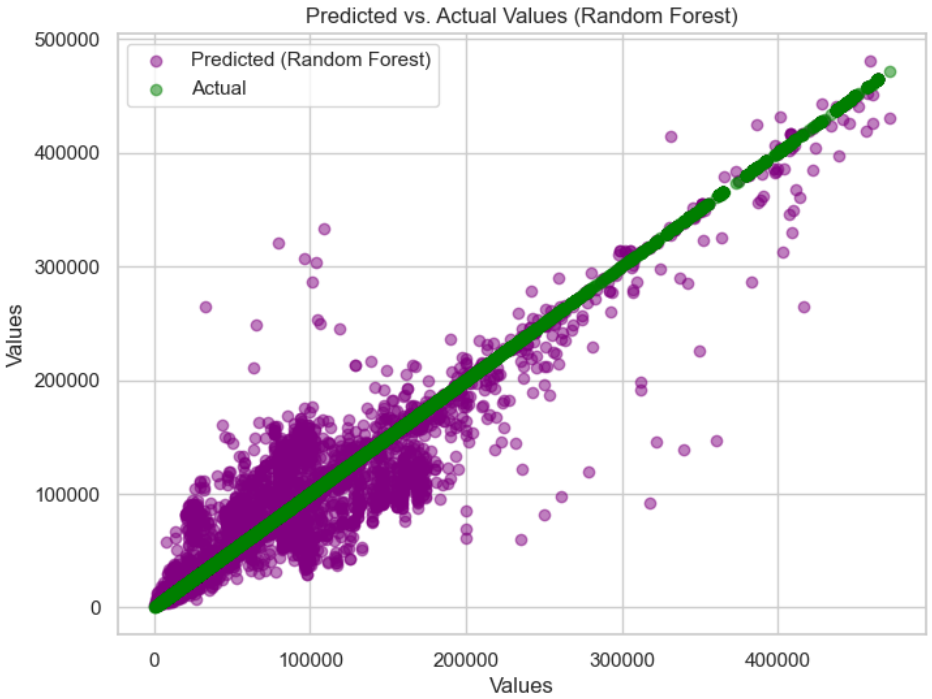


Figure 11: Actual vs Predicted Yield for Random Forest

The feature importance comparison is shown in Figure 12.

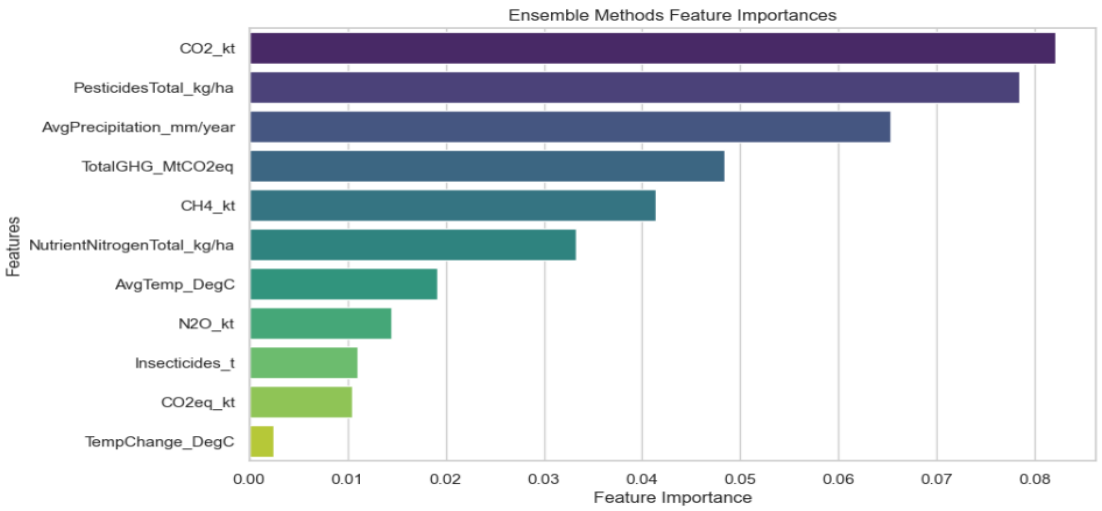


Figure 12: Feature Importance for Random Forest

It is observed that carbon dioxide, pesticides, and average precipitation have the highest importance in the model.

3.3 Bagged Decision Tree

Bagging Decision Tree is used as an alternative model for predicting crop yield. Bagging, which stands for Bootstrap Aggregating, is an ensemble technique that combines multiple Decision Tree regressors to enhance the accuracy and robustness of predictions (Chelliah 2021).

Hyperparameter Tuning

Bagged Decision Tree and Random Forest share many similar approaches to modelling. Firstly, hyperparameter tuning is performed for the Bagging Decision Trees using randomsearchcv. The n_estimators hyperparameter is optimised to determine the number of decision tree estimators in the ensemble. The best parameters are found to be n_estimators of 183 and maximum depth of 20. The Bagging Decision Trees model is created with 183 decision tree estimators, each with a maximum depth of 20. This ensemble approach leverages the diversity of multiple decision trees to improve predictive performance. The code for hyperparameter tuning and model creation are as follows:

```
param_dist = {
    'n_estimators': randint(100, 200),
}

random_search_bd = RandomizedSearchCV(estimator=bagged_decision_trees, param_distributions=param_dist,
                                      scoring='neg_mean_squared_error', cv=5, n_iter=10)
random_search_bd.fit(X_train, y_train)
best_bagged_decision_trees = random_search_bd.best_estimator_
print("Best Bagged Decision Trees parameters:", random_search_bd.best_params_)
```

Figure 13: Hyperparameter Tuning for Bagged Decision Trees

Evaluation

Now, we evaluated the model using 4 evaluators, there are mean squared error (MSE), mean absolute error (MAE), R-squared (R2) score, and root mean squared error (RMSE). We performed the evaluation on both the training set and testing set, the model evaluation metrics will be compared with other models later.

Table 3: Evaluation Metric Table - Bagged Decision Tree

Evaluator	Training set metrics	Testing set metrics
MSE	135,679,325.48	321,504,686.61
MAE	4,587.43	7,187.06
R2	0.97	0.96
RMSE	11,648.14	17,930.55

Figure 14 shows the actual vs predicted plot for Bagged Decision Tree. The plot shows a promising result as the predicted data matches actual data, even at high crop yields.

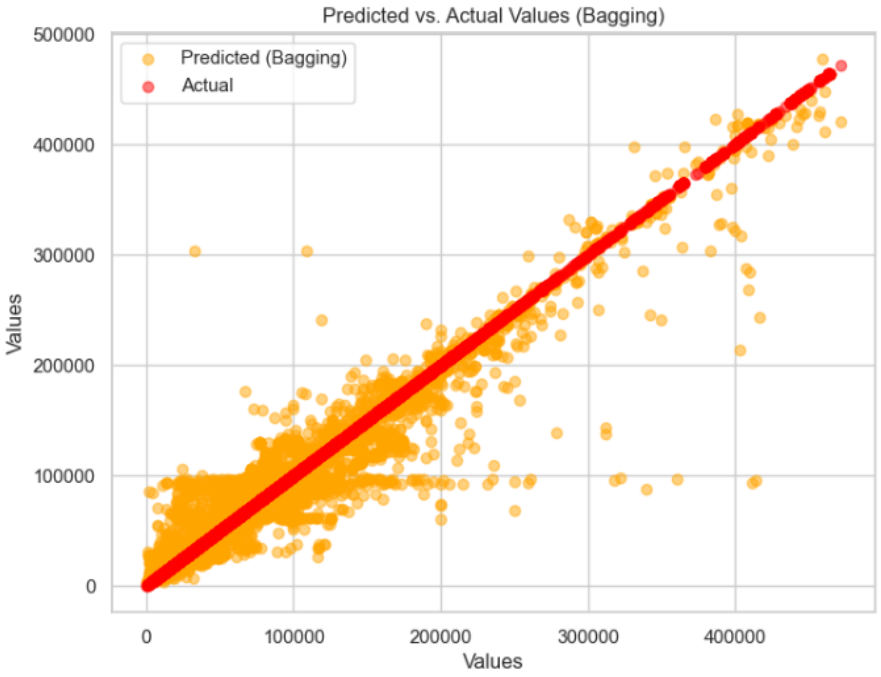


Figure 14: Actual vs Predicted Yield for Bagged Decision Tree

The graph exhibits a positive outcome, with the predicted data closely mirroring the actual data throughout the entire timeline. This suggests that the model holds significant potential as a top contender for predicting crop yield to address global hunger. However, to draw a definitive conclusion, we must engage in a comprehensive assessment by comparing various model evaluation metrics.

The feature importance comparison is shown in Figure 15.

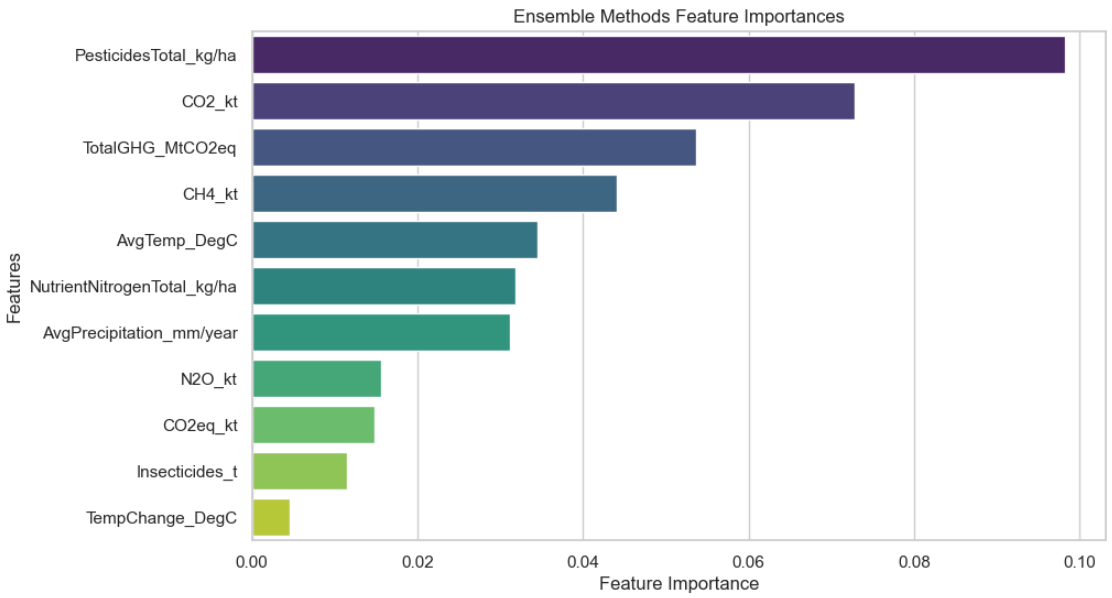


Figure 15: Feature Importance for Bagged Decision Tree

It is observed that pesticides, carbon dioxide and Total Greenhouse Gas have the highest importance in the model.

3.4 XGBoost and LightGBM

This section delves into the utilisation of two prominent gradient boosting algorithms: XGBoost and LightGBM. XGBoost, which stands for Extreme Gradient Boosting, is renowned for its robust optimisation and regularisation techniques. Conversely, LightGBM is praised for its remarkable speed and efficiency, attributed to its innovative data sampling methods. We conducted trials with both methods to assess their performance in predicting time-based data.

Hyperparameter Tuning

To commence, the process began with the search for the best set of hyperparameters for these models. This task was automated using RandomizedSearchCV. The resulting table provides a detailed breakdown of the hyperparameters that were fine-tuned, along with their respective optimal values (Table 4).

Table 4: List of hyperparameters tuned for gradient boosting methods

XGBoost Hyperparameters	Best Values		LightGBM Hyperparameters	Best Values
n_estimators	241		num_iterations	490
learning_rate	0.4307		learning_rate	0.1468
max_depth	3		max_depth	3
min_child_weight	23		num_leaves	40
subsample	0.7160		min_data_in_leaf	2
colsample_bytree	0.6705		bagging_fraction	0.4170
gamma	7		feature_fraction	0.7203
alpha	0.5478		lambda_l1	0.0023

lambda	9.1441		lambda_l2	6.0467
			min_gain_to_split	0.1863
			min_sum_hessian_in_leaf	0.3456

Evaluation

With the optimal hyperparameter sets in place, both ensemble methods were trained using the training dataset and then tested with the testing dataset. The root mean squared errors (RMSE) for both training and testing phases are shown in Table 5. For additional evaluation metrics, please refer to Section 3.7.

The training and testing RMSE values for both XGBoost and LightGBM are notably similar, with training errors being lower than testing errors. This lower training error compared to the testing error suggests the possibility of overfitting, where the models may be fitting the training data too closely but might not generalise well to unseen data. While the models exhibit similar performance, the discrepancy in errors between the training and testing phases indicates the need for further investigation into potential overfitting issues.

Table 5: Training and testing error

	Training RMSE	Testing RMSE
XGBoost	20,075.17	33,809.82
LightGBM	21,552.26	32,300.90

In addition to the RMSE results, we conducted an examination of residual plots to gain further insights into the model performance. In our analysis of the residual plot, as depicted in Figure 16, we observe an interesting trend: for small predicted values, the residuals tend to cluster closely around zero, indicating that the model performs relatively well in this range. However, as the predicted values increase, the residuals become more scattered and exhibit greater variability. This pattern implies that the gradient boosting methods' performance is robust when making predictions in the lower range but faces challenges when dealing with higher predicted values. This shift from denser residuals to increased scatter emphasises the need for model improvement, particularly in addressing the growing variability and errors associated with larger predictions.

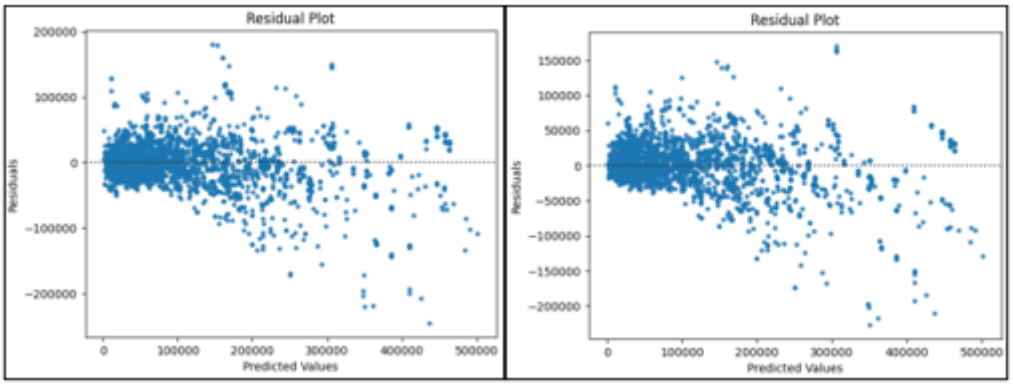


Figure 16: XGBoost (left) & LightGBM (right) Residual Plots

3.5 Time-series Forecasting LSTM

This section explores the use of Long Short-Term Memory (LSTM) neural networks for time series forecasting of crop yield. LSTM networks are well-suited for sequential data and can capture complex temporal patterns (Keith 2022).

The project leverages several essential libraries, including TensorFlow and Keras, to implement and train the LSTM model. These libraries provide powerful tools for deep learning and time series analysis.

A set of features, including environmental and agricultural variables, are selected for the time series forecasting model. Prior to training, the features are standardised using the StandardScaler. Standardisation ensures that all features have a mean of 0 and a standard deviation of 1, preparing the data for effective LSTM training.

The data is split into training and testing sets, with a consideration for a history size of 5. A Sequential model with LSTM and Dense layers is constructed to capture temporal dependencies in the data. The model's architecture is tailored to accommodate the input data format.

Hyperparameter Tuning

Hyperparameter tuning is conducted to optimise the LSTM model's architecture. The tuner explores different hyperparameter combinations, such as the number of units in the LSTM layer, to enhance the model's forecasting capabilities. The best model architecture is chosen based on its validation loss. The code for hyperparameter tuning is as follows:

```
X = []
y = []

for i in range(len(data) - history_size):
    X.append(data[i:i+history_size])
    y.append(target[i+history_size])

X = np.array(X)
y = np.array(y)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

def build_lstm_model(hp):
    model = Sequential()
    model.add(LSTM(units=hp.Int('units', min_value=32, max_value=256, step=32), activation='relu', input_shape=(history_size, len(features))))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mse')
    return model

tuner = RandomSearch(
    build_lstm_model,
    objective='val_loss',
    max_trials=5,
    directory='my_dir',
    project_name='my_project'
)

tuner.search(X_train, y_train, epochs=10, validation_split=0.2)
best_model = tuner.get_best_models(num_models=1)[0]
y_pred_best = best_model.predict(X_test)
```

Figure 17: Hyperparameter Tuning for LTSM

Evaluation

Once we applied hyperparameter tuning and created the model based on it, we proceed with model evaluation (Table 6).

Table 6: Model Evaluation for LTSM

Type	MSE	RMSE	R2	MAE
Training	685,135,476.68	26,175.09	0.918	13,687.15
Test	851,964,791.73	29,188.44	0.897	14,767.47

Finally, we did data visualisation of the LSTM model, here's the model's prediction against the actual data.

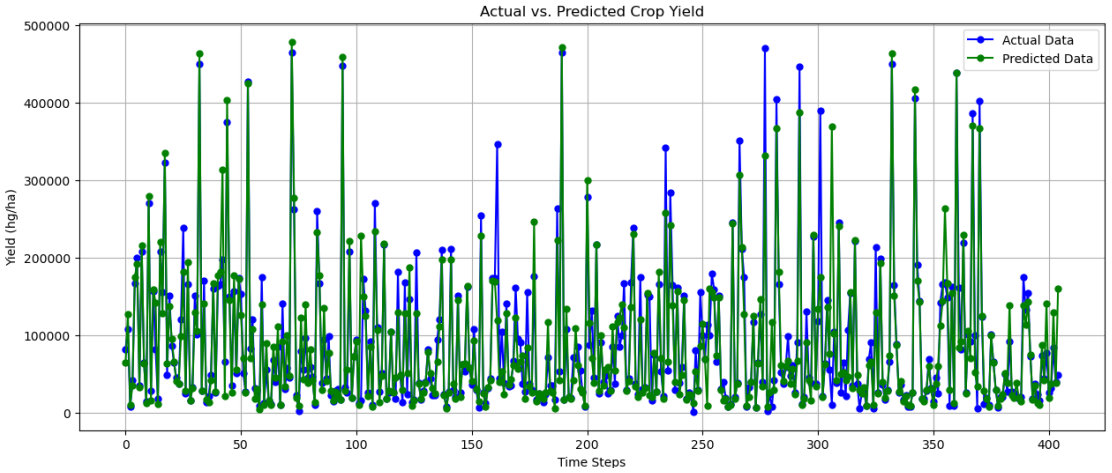


Figure 18: Model Visualisation for LTSM

It can be observed that the predicted data matches the actual relatively well. But it is not perfect, as there are still some areas that are too high or too low.

3.6 Neural Network

In this model, the fastai library was used to generate a Neural Network model using the problem dataset (fastai 2023). Fastai is one of the deep learning libraries that can provide cutting-edge results by mixing and matching in building new approaches efficiently. Fastai works flexibly to make things easier for users.

The data consists of 14 parameters (11 Numerical data types, 3 Categorical data types) and 1 target parameter ('Yield_hg/ha'). After applying Pearson analysis, the high correlated value were dropped. Final dataset parameters to be used in the model consist of: 'Country_Code', 'Item_Code', 'Year', 'Yield_hg/ha', 'CO2_kt', 'AvgPrecipitation_mm/year', 'AvgTemp_DegC', 'PesticidesTotal_kg/ha', 'TempChange_DegC', 'NutrientNitrogen Total_kg/ha'.

There are several unique features in fastai to help with tabular data collection. A series of processes to structure was applied to the dataset preparation pipeline including:

- Determine which parameters are classified as continuous or categorical data types.

- Determine the dependent variable (y value): 'Yield_hg/ha'.
- Define a set of process transformations to apply to the tabular dataset:
    - Normalize: Normalize the continuous variables (subtract the mean and divide by the std).
    - Categorify: Take every categorical variable and make a map from integer to unique categories, then replace the values by the corresponding index.

Next, the TabularsDataLoaders object was defined to describe what the data will look like with all the predefined information. It contains information about how much should be fed to the model at once (batch size), how many processes should be performed to load the data.

```
dls = TabularDataLoaders.from_df(train_df,
    procs=[Categorify, Normalize],
    cont_names=cont_names,
    cat_names = cat_names,
    y_names=dep_var,
    y_block=RegressionBlock(),
    valid_idx=list( np.random.permutation(len(train_df))[:int(len(train_df)*.1)]),
    bs=64)
```

The show_batch() function was used to see a sample batch including x and y values as shown in Figure 19 below:

```
dls.show_batch()
```

Year	Item_Code	CO2_kt	AvgPrecipitation_mm/year	AvgTemp_DegC	PesticidesTotal_kg/ha	TempChange_DegC	NutrientNitrogenTotal_kg/ha	Yield_hg/ha
2001	27	7.583253e+03	854.000002	16.200001	6.430000e+00	1.165	42.729999	10.977175
2003	116	9.593239e+03	281.999989	27.990000	9.999944e-03	1.238	7.330002	12.064976
1998	236	-1.144622e+04	346.000011	18.240000	1.270000e+00	1.483	17.640000	8.294049
2011	15	4.836838e+04	494.999985	15.490000	2.160000e+00	0.446	33.669997	10.409009
2008	56	-7.231116e+05	645.000016	5.840000	2.340000e+00	1.090	210.630005	10.925237
2008	122	4.066582e+04	494.000001	24.890000	3.900000e-01	0.362	96.589997	11.621573
2008	83	5.644198e+04	2348.000003	26.940000	1.910000e+00	0.574	38.180002	10.292349
2001	83	6.570536e+06	1782.000006	25.590000	2.710000e+00	0.746	29.379998	9.833816
2011	56	2.605896e+05	1071.000000	26.560000	5.360374e-10	0.723	4.979998	9.488199
2002	56	1.456998e+05	2274.000063	22.230000	5.000000e-01	0.777	50.889999	9.569343

Figure 19: Snapshot of sample batch from TabularDataLoaders

After that, a model can be defined using the tabular_learner method. When our model is defined, fastai will try to infer the loss function based on our y_names earlier.

```
learn = tabular_learner(dls, metrics=exp_rmspe, config=tc, loss_func=MSELossFlat())
```

The summary function gives us a way to see in detail the layers that fastai generates for a model (Figure 20).

```
1 learn.summary()
```

```
TabularModel (Input shape: 64 x 3)
================================================================================
Layer (type)         Output Shape         Param #    Trainable
================================================================================
                     64 x 22
Embedding                                 2442       True
________________________________________________________________________________
                     64 x 10
Embedding                                 250        True
________________________________________________________________________________
                     64 x 6
Embedding                                 72         True
Dropout
BatchNorm1d                               12         True
________________________________________________________________________________
                     64 x 200
Linear                                    8800       True
ReLU
BatchNorm1d                               400        True
Dropout
________________________________________________________________________________
                     64 x 100
Linear                                    20000      True
ReLU
BatchNorm1d                               200        True
Dropout
________________________________________________________________________________
                     64 x 1
Linear                                    101        True
SigmoidRange
________________________________________________________________________________

Total params: 32,277
Total trainable params: 32,277
Total non-trainable params: 0

Optimizer used: <function Adam at 0x000002024EDB7B50>
Loss function: FlattenedLoss of MSELoss()

Callbacks:
  - TrainEvalCallback
  - CastToTensor
  - Recorder
  - ProgressCallback
```

Figure 20: Model summaries consist of detail layers generated for the model

All the following parameters are automatically generated by fastai:

1. TabularModel consists of 3 Embedding layers, where each layer represents the data transformation process from categorical to numerical.

Categorical data	
Parameter	Distinct value
'Country_Code'	111
'Item_Code'	11
Year'	23

   Note: Due to batch size set, it is possible that not all distinct values are included in the batch set.

2. The first Linear layer consists of 200 nodes with ReLU activation function, BatchNorm1d, and Dropout.
3. The second Linear Layer consists of 100 nodes with ReLU activation function, BatchNorm1d, and Dropout.
4. The last Liner layer produces a single output node with the SigmoidRange function.
5. Optimizer: Adam
6. Loss function: MSE Loss
7. Callbacks

Once the model design is complete, we continue to train the model with parameters; number of epochs, learning rate, and weight decay (Figure 21).

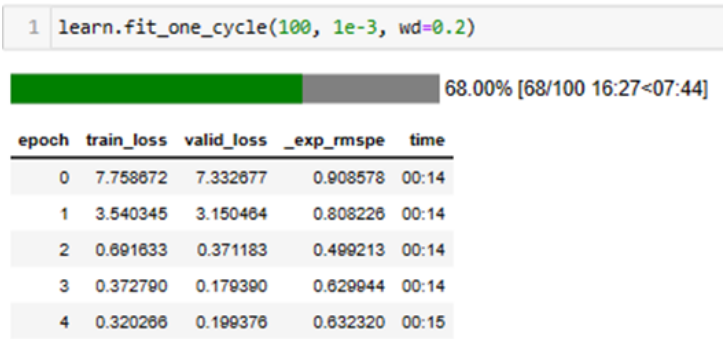


Figure 21: Snapshot of loss values development on iteration

Hyperparameter Tuning

In this step, several experiments were conducted to determine the optimal model as the selected outcome. Different combinations of parameters, number of epochs and learning rate were selected (Figure 22).

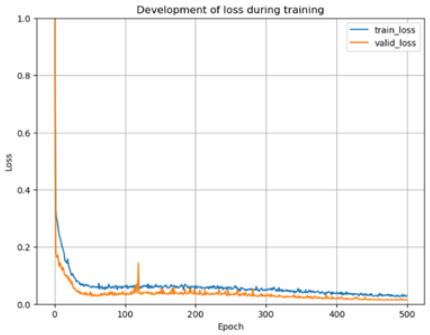
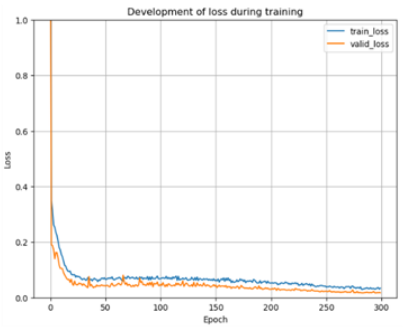
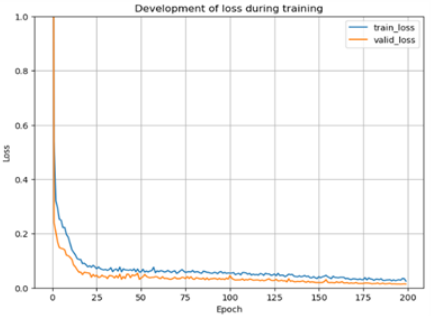
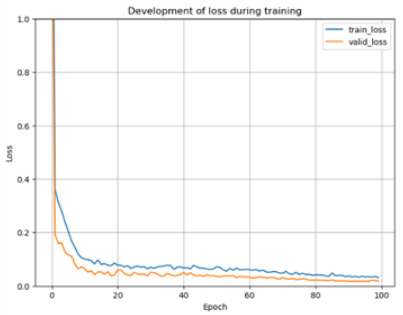
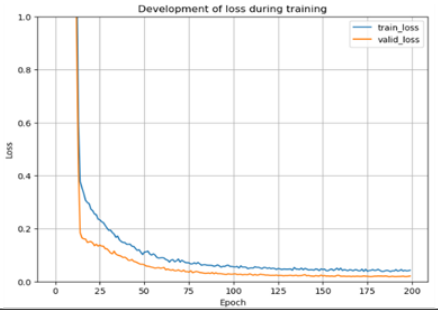
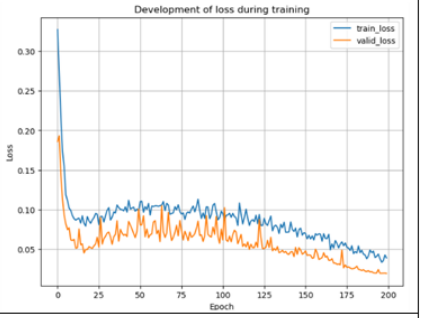
Dataset Numerical	CO2_kt', 'TotalGHG_MtCO2eq', 'AvgPrecipitation_mm/year', 'AvgTemp_DegC', 'PesticidesTotal_kg/ha', 'TempChange_DegC', 'NutrientNitrogenTotal_kg/ha'	
Dataset Categorical	'Country_Code', 'Item_Code', 'Year'	
Label	Yield_hg/ha'	
		
Epoch	500	300
Learning rate	0.003	0.003
Weight decay	0.2	0.2
Batch size	64	64
		
Epoch	200	100
Learning rate	0.003	0.003
Weight decay	0.2	0.2
Batch size	64	64
		
Epoch	200	200
Learning rate	0.0003	0.03
Weight decay	0.2	0.2
Batch size	64	64

Figure 22: Comparison of loss function in training models based on different hyperparameter selection

Epoch	Learning rate	Weight decay	Batch size	Dataset	MSE	RMSE	R2	MAE	MAPE
500	0.003	0.2	64	Training	56,676,830.337	7,528.401	0.993	3,801.759	0.0558
				Test	88,629,354.733	9,414.316	0.989	4,414.057	0.0712
300	0.003	0.2	64	Training	79,431,211.602	8,912.419	0.99	4,785.703	0.0722
				Test	104,468,063.522	10,220.962	0.987	4,919.310	0.0789
200	0.003	0.2	64	Training	50,948,031.816	7,137.789	0.994	3,521.056	0.0497
				Test	86,437,292.407	9,297.166	0.989	4,191.945	0.066
100	0.003	0.2	64	Training	79,431,211.602	8,912.419	0.99	4,785.703	0.072
				Test	108,643,401.030	10,423.214	0.987	5,215.000	0.083
200	0.03	0.2	64	Training	9,079,310.078	9,528.543	0.989	4,675.142	0.0716
				Test	122,597,964.967	11,072.396	0.985	5,123.829	0.0817
200	0.0003	0.2	64	Training	83,941,356.723	9,161.952	0.989	4,858.444	0.0829
				Test	115,313,910.131	10,738.431	0.986	5,368.931	0.0962

Figure 23: Evaluation model comparison results between different hyperparameter selection

The comparison results of the development of the loss function in the training model show that after 100-200 epochs, no improvement was obtained. Moreover, training and validation have the same pattern, so there is no need for further training. The comparison results between the training and testing datasets show that 200 epochs with hyperparameter selection is the optimal result achieved by the model.

Model evaluation is carried out by plotting the predicted and actual values of the dependent variable 'Yield_hg/ha' (Figure 24).

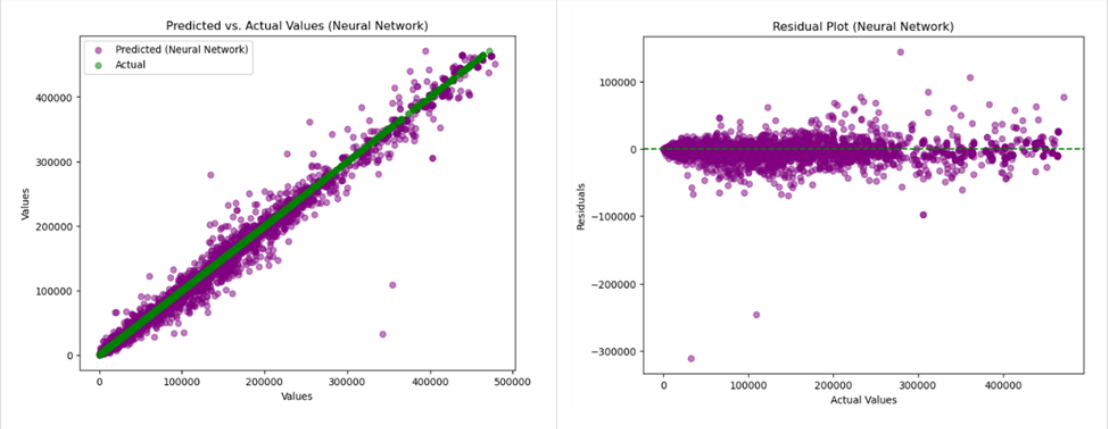


Figure 24: Actual vs Predicted Yield (Left) and residual Plot (Right) for Neutral Network

Table 7: Model Evaluation for Neural Network

	MSE	RMSE	R2	MAE	MAPE
Training	50,948,031.816	7,137.789	0.994	3,521.056	0.0497
Test	86,437,292.407	9,297.166	0.989	4,191.945	0.066

For further analysis, we tried to look at more detailed data where we calculated models of each Crop type ('Item_code'):

Yield_hg/ha	Rice	Potato	Maize	Sorghum	Wheat	Soya	Sweet Potato	Cassava	Yam
MSE	6,723,283.13	181,109,377.15	19,724,388.45	16,040,604.46	7,004,137.57	2,496,699.12	115,419,113.36	131,269,308.47	209,907,905.58
RMSE	2,594.68	13,457.68	4,441.22	4,005.07	2,464.53	1,580.10	10,743.33	11,457.28	14,488.19
MAE	1,325.26	8,520.54	2,257.67	1,398.94	1,490.52	738.34	5,876.02	5,307.21	5,807.21
R2	0.98	0.987	0.976	0.934	0.968	0.951	0.974	0.977	0.932
MAPE	0.042	0.051	0.058	0.093	0.063	0.067	0.053	0.049	0.067

Figure 25: Model evaluation results from in-depth analysis based on Crop type

The results show that there are quite high error values produced from potatoes, sweet potatoes, cassava, and yam. After conducting several analyses of outlier detection and in-depth analysis of several countries, we still cannot find out what parameters contribute to the level of error rates. If we had more time to analyse, we might try another approach like choosing a Stratified train-test split dataset and others to solve the problem.

3.7 Final Selection

In this study, evaluation of the regression from tabular data using machine learning models are conducted and compared for linear, polynomial regression, random forest, bagging decision tree, time-series forecasting LSTM, XGBoost, LightGBM, and Neural Network. All these regression models were tested using test dataset from crop yield dataset. The obtained results for the crop yield datasets are shown in Table 8 for mean squared error (MSE), mean absolute error (MAE), R-squared (R^2) and root mean squared error (RMSE) respectively.

The results from the comparative studies showed that the Neural Network is found to be the best to use as a regression model and outperformed other machine learning techniques. However, to answer the research questions, better interpretive modelling techniques were needed. We choose the feature importance results of the bagging decision tree as parameters to describe the relationship between the parameters and the model results.

Table 8: Model Evaluation Comparison from Different Machine Learning Techniques

Models	Mean Squared Error (MSE)	Mean Absolute Error (MAE)	R-squared (R2) Score	Root Mean Squared Error (RMSE)
Random Forest	378,139,675.67	7,573.62	0.95	19,445.81
Bagged Decision Tree	321,504,686.61	7,187.06	0.96	17,930.55
Time-Series Forecasting LSTM	851,964,791.73	14,767.47	0.90	29,188.43
XGBoost	1,143,104,198.23	20,790.90	0.89	33,809.82
LightGBM	1,043,348,163.57	19,205.07	0.90	32,300.90
Neural Network	108,643,601.50	5,215.00	0.98	10,423.21
Linear Regression	2,743,055,000.00	35,452.20	0.67	52,374.19
Polynomial Regression	1,219,750,000.00	22,604.18	0.85	34,924.92

4. Main Findings

In this section, we present the main findings of the research using effective visualisation techniques. As mentioned in Section 3.7, we selected the bagging decision tree to predict and visualise the outcomes. There are a total of 10 crop types that can be analysed; however, we will focus on the top 3 crops: rice, potatoes, and wheat, for conciseness. Moreover, we mention the limitations of the study at the end.

4.1 Heat Map

At the outset of our analysis, we generated heat maps to visualise and gain a comprehensive overview of anticipated crop yields worldwide for the year 2030. These heat maps, based on our predictive models, offer a geospatial representation of crop yield projections across various regions. This initial step allows us to visualise and understand the global distribution of crop yields.

In our analysis of rice crop yields (Figure 26), we observe distinct patterns across different regions. Notably, countries such as the USA, Australia, Argentina, and China stand out as major producers, consistently yielding significant quantities of rice. In contrast, regions including Africa, Europe, and many Middle Eastern countries exhibit lower crop yields compared to their counterparts.

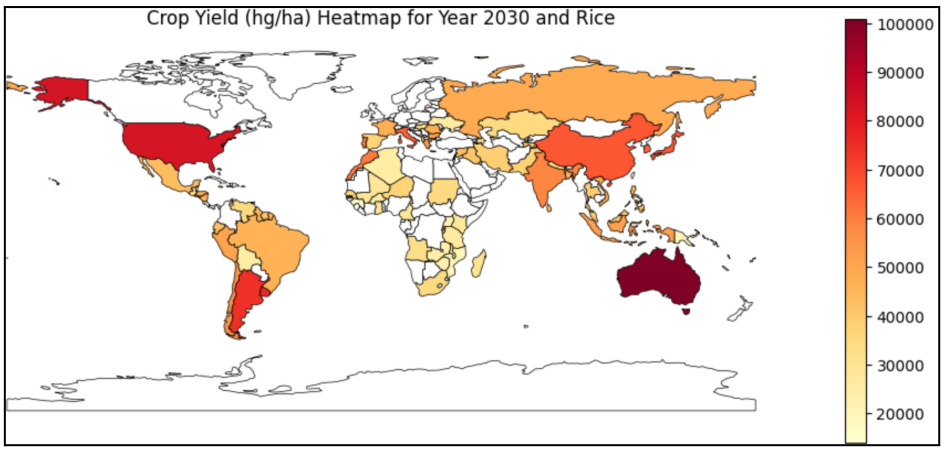


Figure 26: Crop yield heat map for year 2030 and rice

In the context of potato production, as depicted in Figure 27, our observations reveal a different landscape. Countries such as the USA, Australia, Canada, South Africa, and numerous European nations emerge as primary contributors to global potato yields. Potato cultivation is quite widespread, involving many countries in global production. Unlike rice, which is mainly produced by a limited set of countries, potatoes see a more universal participation, with a wider range of nations contributing to the global supply. This reflects the adaptability and popularity of potatoes in various agricultural contexts around the world.

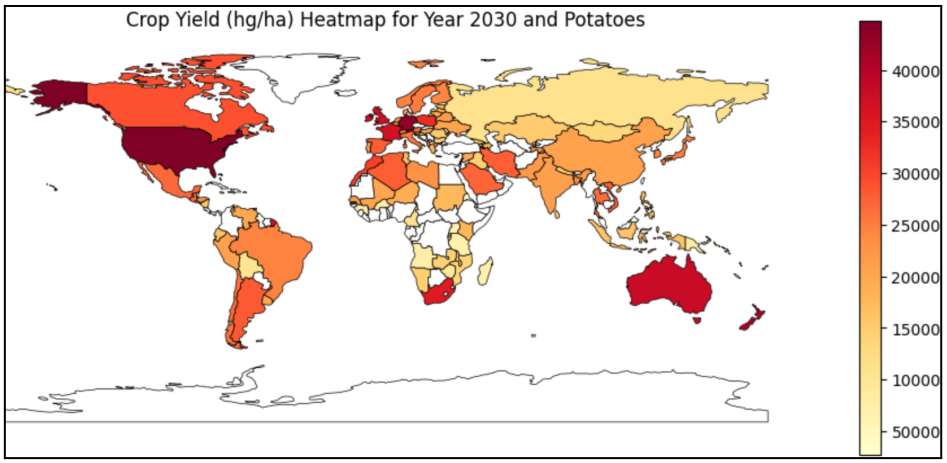


Figure 27: Crop yield heat map for year 2030 and potatoes

Wheat production, much like potatoes, is a global effort involving numerous nations. However, apart from certain European countries such as Germany, France, and Poland, as well as China and New Zealand, as shown in Figure 28, many other

countries contribute comparably smaller quantities to the global wheat supply. These select regions stand out as major players in the wheat production landscape, while most nations participate with smaller-scale contributions.

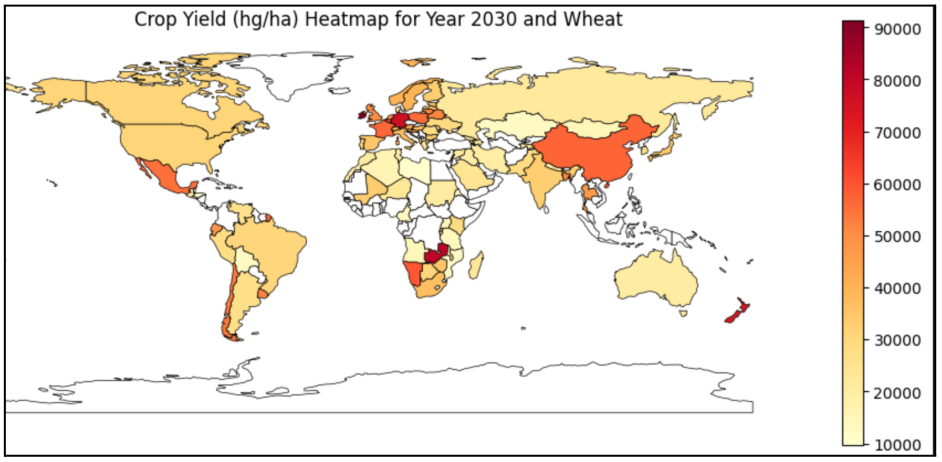


Figure 28: Crop yield heat map for year 2030 and wheat

4.2 Time Lapse Forecast

We conducted an analysis of crop yield forecasts for the top 10 developed and developing countries in the world, utilising predicted data from 2014 to 2030. Our objective was to observe and analyse the trends in crop yield projections based on our predictions over this period. Using our selected model, it allows us to assess the long-term trends in crop production for both developed and developing nations.

The countries are based on the United Nations criteria:

- Top 10 Developed Countries
    - United States, Canada, United Kingdom, Germany, France, Japan, Australia, Sweden, Switzerland, and Singapore.
- Top 10 Developing Countries
    - China, India, Brazil, Mexico, South Africa, Nigeria, Indonesia, Turkey, Vietnam, and Thailand.

From 2014 to 2030, as depicted in Figure 29, our prediction indicates that for the top 10 developed countries, the crop yield for rice remains relatively stable for most

nations. There are noteworthy exceptions, with Australia experiencing an increase in rice production, while Germany shows a slight decline. However, these fluctuations in rice yield are not excessively dramatic, suggesting a generally consistent trend across these developed nations over this period.

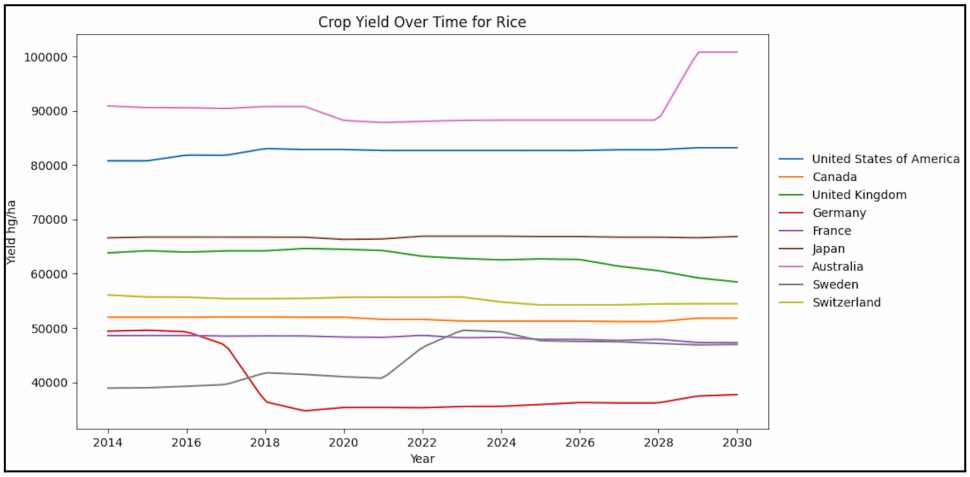


Figure 29: Top 10 developed countries - crop yield over time for rice

Figure 30 illustrates a notable trend in potato production from 2014 to 2030, where many countries are experiencing a decreasing trend in potato yields during this period.

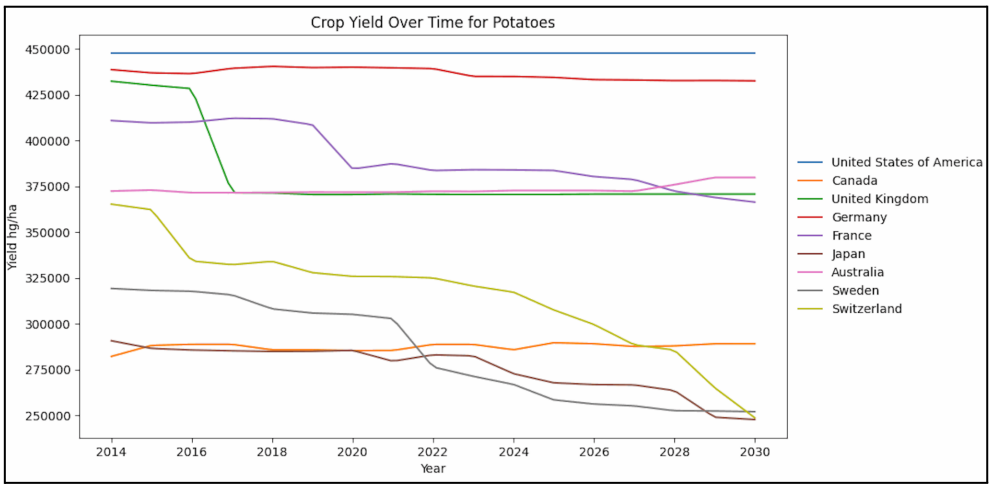


Figure 30: Top 10 developed countries - crop yield over time for potatoes

As highlighted in Figure 31, our analysis indicates that, for wheat production, most

countries are showing either a decreasing or consistent trend in wheat yields from 2014 to 2030. It is important to note that Australia stands out as the sole exception, demonstrating a notable increase in its wheat production during this period.

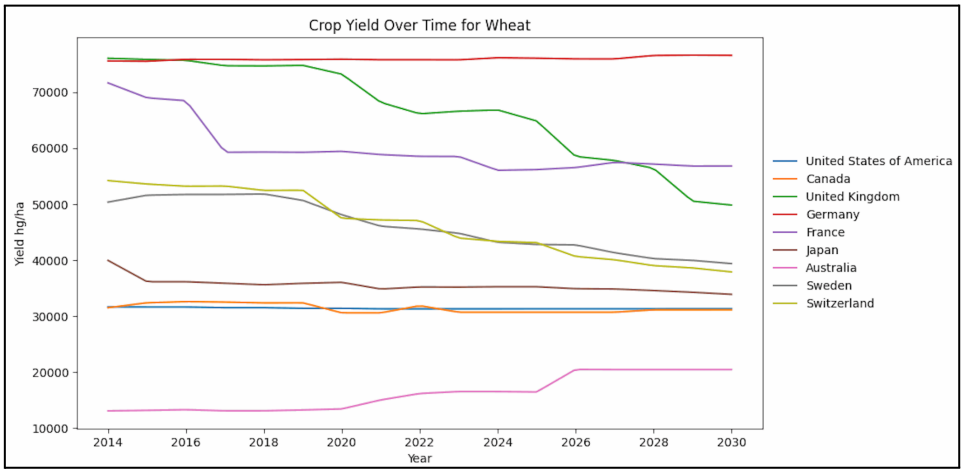


Figure 31: Top 10 developed countries - crop yield over time for wheat

In Figure 32, our analysis of the top 10 developing countries reveals an encouraging trend in rice production from 2014 to 2030. Specifically, most of these nations are either experiencing an increasing trend or maintaining consistent rice yields. Notably, there is no significant decrease observed in rice production across this group of developing countries.

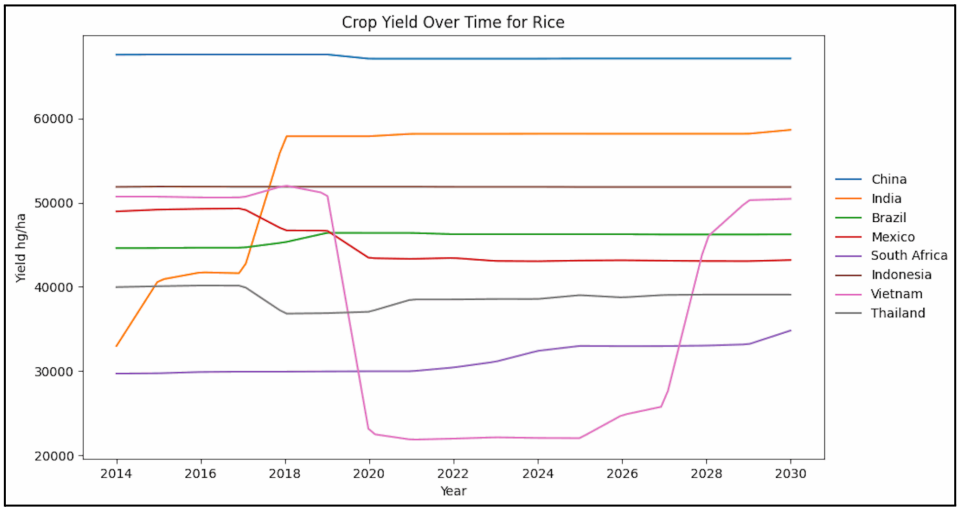


Figure 32: Top 10 developing countries - crop yield over time for rice

As demonstrated in Figure 33, our analysis of the top 10 developing countries reveals a particularly robust trend in potato production from 2014 to 2030. Much like rice, potato production exhibits both a consistent and increasing pattern, with no significant decreases observed across these regions.

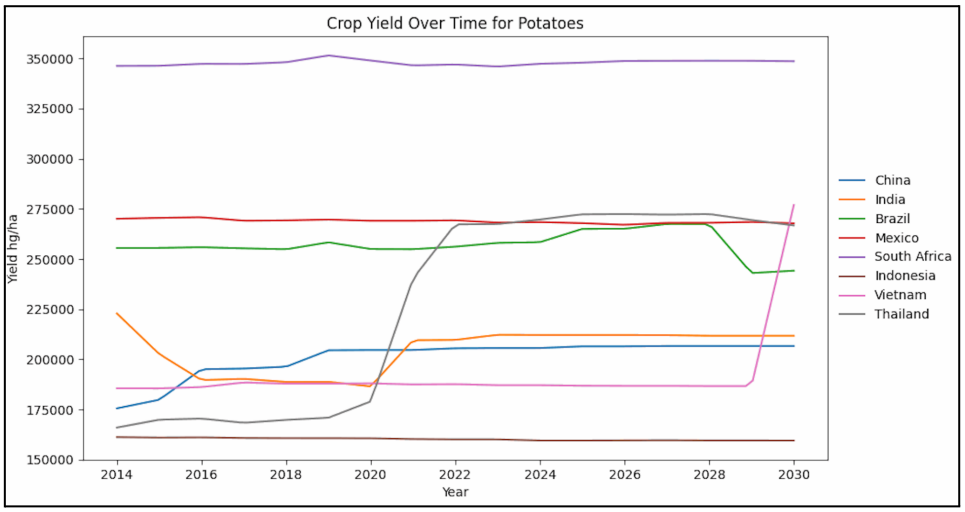


Figure 33: Top 10 developing countries - crop yield over time for potatoes

As illustrated in Figure 34, our analysis of the top 10 developing countries showcases a predominantly consistent trend in wheat production from 2014 to 2030. Each nation in this group exhibits stable or consistent wheat yields during this period, except for Thailand. Thailand stands out with remarkable growth in wheat production, which is notably distinct from the steady trends observed in the other countries. This significant surge in wheat production within Thailand is an interesting outlier that highlights the country's notable contribution to the global wheat supply among developing nations.

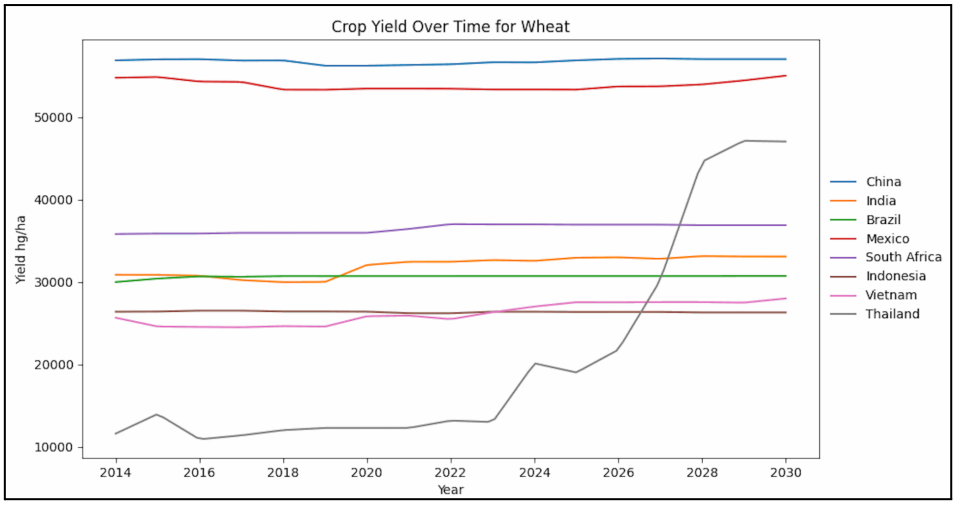


Figure 34: Top 10 developing countries - crop yield over time for wheat

4.3 Food Security Index

Furthermore, we extended our analysis by calculating the Food Security Index. This index serves as a critical metric to assess whether there will be a sufficient food supply to meet human consumption needs by 2030, aligning with the United Nations' Sustainable Development Goal 2. Our calculation considers the predicted crop yield data, area harvested, and human consumption per capita, seen in Equation 1, offering insights into the overall food security status for the selected countries. By evaluating our findings against UN Goal 2, we aim to contribute valuable insights into the progress and challenges related to global food security in the coming years.

$$food\_security\_index = \frac{yield * area\_harvested}{human\_consumption\_per\_capita}$$

The following criteria are employed to label countries as either insufficient, sufficient, or surplus:

- Insufficient: $food\_security\_index < 200$
- Sufficient: $200 \leq food\_security\_index < 1000$
- Surplus: $food\_security\_index > 1000$

In the context of the Food Security Index, Figure 35 illustrates a notable pattern. Our analysis reveals that rice production surpasses consumption in only two countries, namely India and China. However, a significant number of other nations face the challenge of rice insufficiency, where their production falls short of meeting the essential consumption needs.

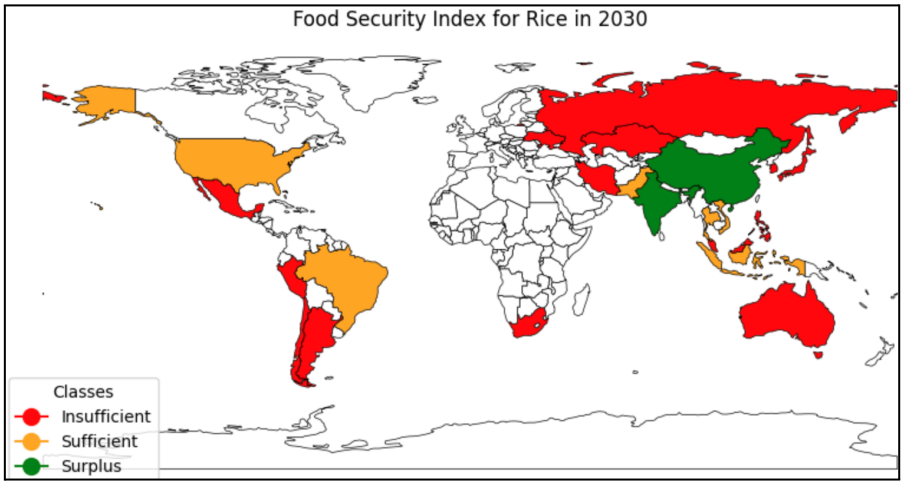


Figure 35: Food security index for rice in 2030

As depicted in Figure 36, most countries exhibit a surplus in potato production, meaning their output exceeds their consumption needs. However, there are exceptions among a select few nations, including the United Kingdom, Japan, and South Korea, which show a concerning trend of insufficient potato production to meet their consumption requirements.

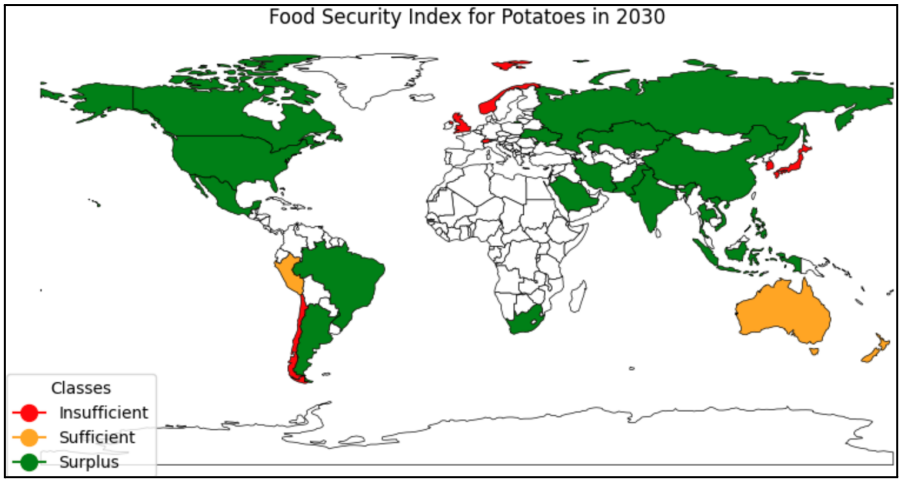


Figure 36: Food security index for potatoes in 2030

Surplus wheat production is observed primarily in two significant nations, China and India, as displayed in Figure 37. However, in the case of several substantial countries like the USA, Australia, Canada, and Russia, the wheat production levels are deemed sufficient to meet their consumption requirements. In contrast, many other nations on the global stage grapple with the challenge of insufficient wheat production, pointing to disparities in food security.

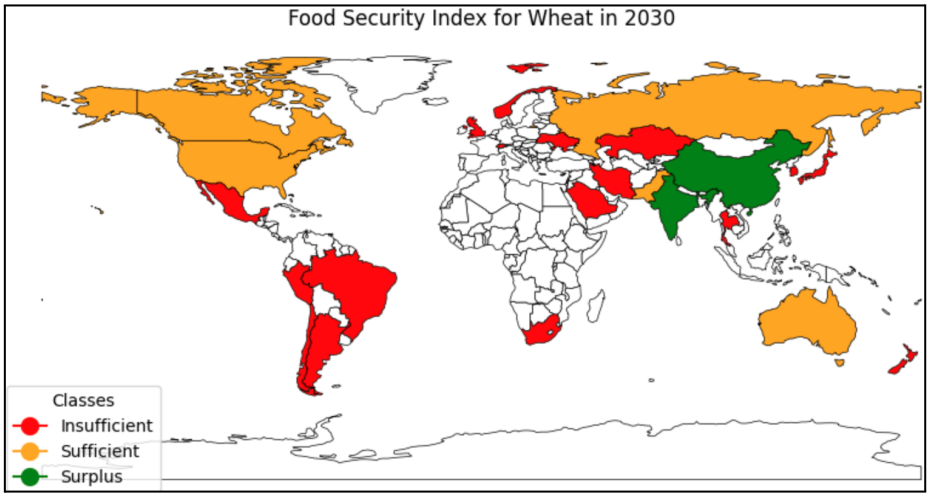


Figure 37: Food security index for wheat in 2030

Observing the Food Security Index for all three major crops – rice, potatoes, and wheat, as depicted in Figure 38, it becomes apparent that there is a general trend. Countries with larger land areas tend to exhibit higher food security, as evidenced by surplus production. Notably, these nations, often characterised by extensive agricultural land, are more adept at ensuring that their production meets or exceeds consumption requirements.

Conversely, smaller-sized countries, limited by land availability and other resource constraints, tend to have lower food security levels. This is reflected in their insufficient or deficient food production, leaving them more vulnerable to potential food shortages and the need for imports to bridge the gap between supply and demand.

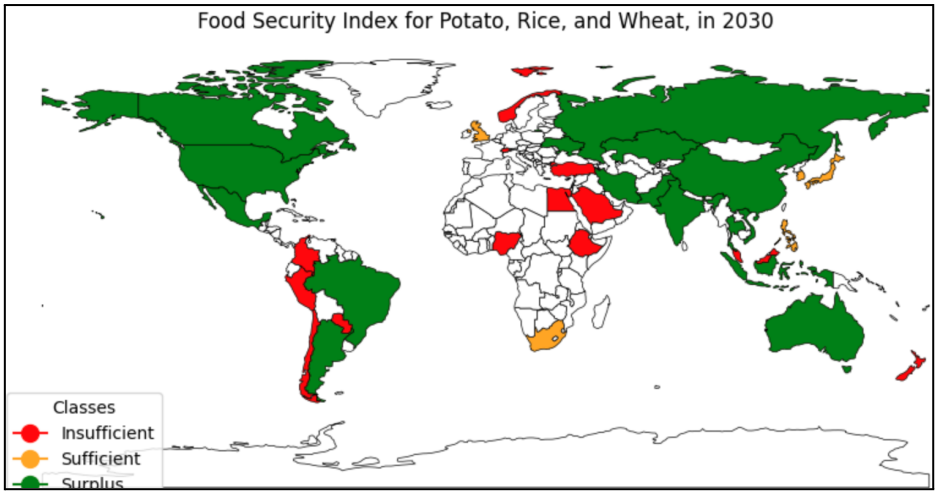


Figure 38: Food security index for rice, potatoes, and wheat, in 2030

4.4 Limitations

In this section, the limitations of the prediction results are discussed. To predict future crop yields effectively, we need to estimate future environmental predictors like rainfall and temperature. However, these predictors tend to fluctuate significantly over time, posing a challenge for our predictions. We attempted to account for the intricate relationships between these factors, such as how temperature affects pesticide use and greenhouse gas emissions.

However, predicting the future values of these interconnected factors, like future temperatures for estimating pesticide use, is a formidable task. This complexity led us to choose linear regression as a practical method, allowing us to make predictions for each factor individually. As observed in Figure 39, the patterns of the predictors are quite dynamic, but the estimation from the year 2014 could not capture these trends suitably.

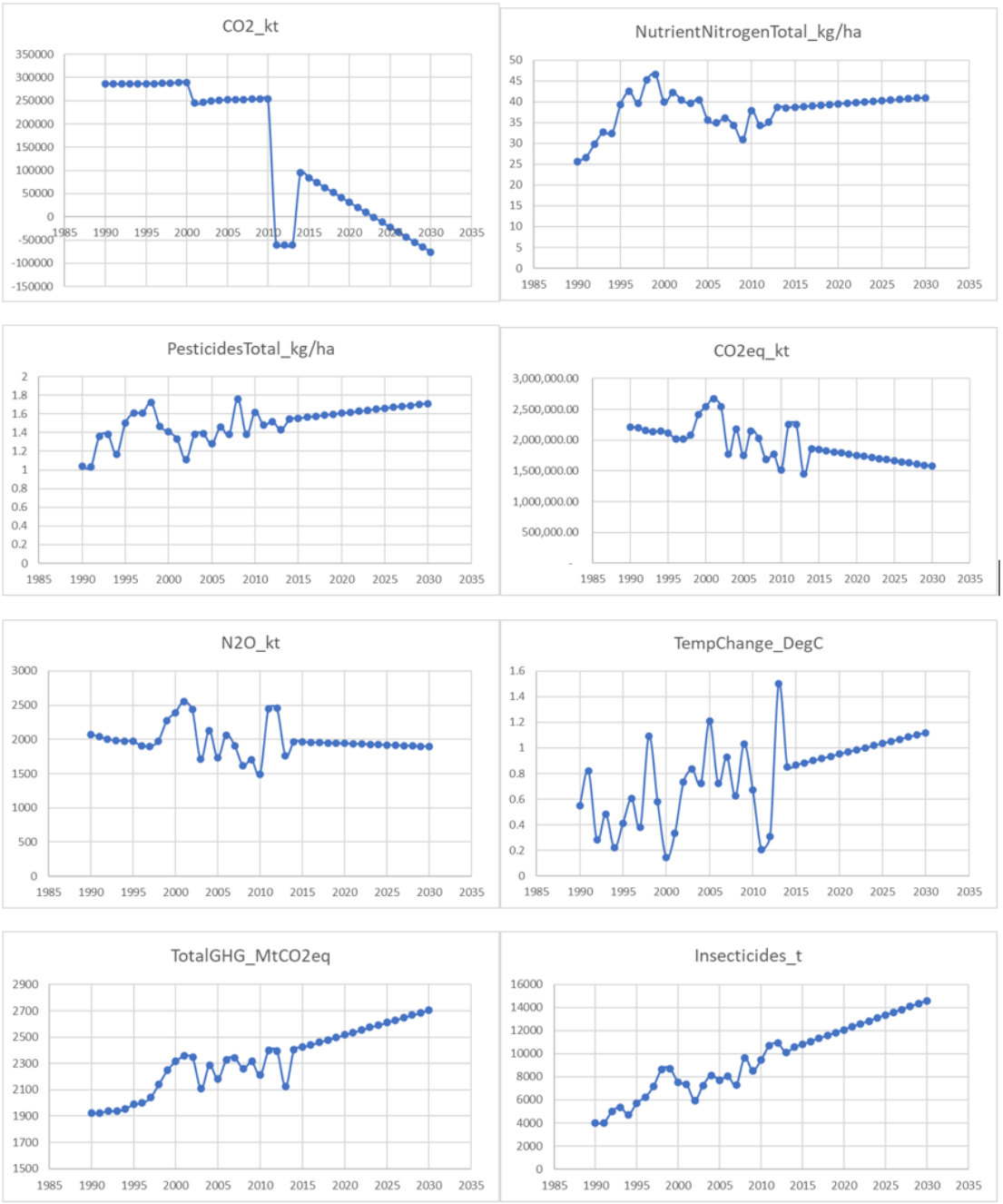


Figure 39: Estimation of future predictors using linear regression

To overcome this challenge, future research should delve into more sophisticated machine learning techniques and harness the expertise of specialists in the field. We also need to improve data quality and collection methods to develop more robust predictive models. Acknowledging these limitations is a crucial step towards enhancing our understanding of how environmental factors affect crop production.

3. Conclusion

In this project, it was established that Machine Learning can effectively predict crop yields based on historical climate and environmental data. Several Machine Learning models were tested which includes Ensemble Methods, Neural Network and Standard Regression. These were subsequently evaluated based on metrics such as MSE, MAE, R^2 and RMSE.

Results found Neural Network to be the best performing out of all the models tested. However, Bagged Decision Tree was ultimately selected as it has better interpretability and transparency.

There are certain limitations to the model, which involves estimating future environmental predictors like rainfall and temperature. Currently it is done using a simple linear regression, but more sophisticated models will need to be explored for a more accurate prediction.

Heat map and time lapse forecasts were generated and showed changing crop yields across different regions across time. There were more drastic changes observed in Top 10 Developing Countries compared to Top 10 Developed Countries.

The Food Security Index was calculated as a critical metric to assess whether there will be a sufficient food supply to meet human consumption needs by 2030, aligning with the United Nations Global Sustainable Development Goals (specifically Goal 2).

Results indicated that the goal is on track to being met, but not in all food classes. For example, there will be an abundance of potatoes but not rice. There could potentially be an opportunity for countries to assist each other, especially those with surplus. This will require a coordinated effort between all countries, and the United Nations will have an important role to play in this.

Reference

United Nations. 2023. “The Sustainable Development Goals Report Special Edition 2023.” https://unstats.un.org/sdgs/report/2023/The-Sustainable-Development-Goals-Report-2023.pdf.

Yiu, Tony. 2019. “Understanding Random Forest.” Medium. Towards Data Science. June 12, 2019. https://towardsdatascience.com/understanding-random-forest-58381e0602d2.

Chelliah, Indhumathy. 2021. “Bagging Decision Trees — Clearly Explained.” Medium. September 6, 2021. https://towardsdatascience.com/bagging-decision-trees-clearly-explained-57d4d19ed2d3.

Keith, Michael. 2022. “Exploring the LSTM Neural Network Model for Time Series.” Medium. October 7, 2022. https://towardsdatascience.com/exploring-the-lstm-neural-network-model-for-time-series-8b7685aa8cf.

Fastai. 2023 “Tabular Data Documentation.” Fastai. October 30, 2023. https://docs.fast.ai/tabular.data.html