**MLP Project Report**:

Predicting Export Value of Agricultural Products

By

Name:

Date

Table of Contents

[Chapter One 1](#_Toc165706764)

[**Performance** 1](#_Toc165706765)

[Performance Evaluation Methodology 1](#_Toc165706766)

[Mathematical Formulas for Performance Metrics 1](#_Toc165706767)

[Model Performance Evaluation 2](#_Toc165706768)

[Model Training and Testing Sets 2](#_Toc165706769)

[Chapter 2 3](#_Toc165706770)

[**MLP Model** 3](#_Toc165706771)

[Description of Multilayer Perceptron (MLP) Model 3](#_Toc165706772)

[Steps to Prevent Overfitting 4](#_Toc165706773)

[Chapter Three 5](#_Toc165706774)

[**Derivation of Feature and Labels from Given Data** 5](#_Toc165706775)

[Label Derivation 5](#_Toc165706776)

[Selection Rationale for Features 5](#_Toc165706777)

[Chapter Four 13](#_Toc165706778)

[**Preprocessing of Features for Model Building** 13](#_Toc165706779)

[Feature Engineering and Data Merge 13](#_Toc165706780)

[Handling Missing Values 26](#_Toc165706781)

[Encoding Categorical Variables 26](#_Toc165706782)

[Feature Scaling 26](#_Toc165706783)

# Chapter One

## **Performance**

### Performance Evaluation Methodology

The ability of the multilayer perceptron (MLP) model to accurately predict the export value of agricultural products three years into the future was hinged on the data merge methodology and training procedure. The primary metrics used for evaluation were Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R²) score. These metrics provide valuable insights into the accuracy and precision of the model's predictions.

### Mathematical Formulas for Performance Metrics

Mean Absolute Error (MAE): The MAE is the average of the absolute differences between the actual values of the test data and predicted values gotten from the model. The mathematical formula for MAE is given by:

𝑀𝐴𝐸 = (1/𝑛)∑ ∣𝑦− 𝑦𝑖∣

Where:

* n is the total number of instances.
* 𝑦 is the export value and
* yi is the predicted export value.

​Mean Squared Error (MSE): The MSE is calculated as the average of the squared differences between the actual and predicted values. The mathematical formula for MSE is given by:

𝑀S𝐸 = (1/𝑛)∑(𝑦− 𝑦𝑖)2

Root Mean Squared Error (RMSE): The RMSE is the square root of the MSE value. It provides a measure of the model's prediction error in the same units as the target variable. The formula for RMSE is given by:

𝑅𝑀𝑆𝐸 = √MSE

R-squared (R²) Score: The R² score, which is also known as the coefficient of determination, measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It is a measure of how well the model fits the data. The formula for R² is:

𝑅2 = 1 − 𝑆𝑆𝑟𝑒𝑠idual/𝑆𝑆𝑡𝑜𝑡al

Where:

S𝑆𝑟𝑒𝑠idual = is the sum of squares of residuals (predicted values minus actual values).

𝑆𝑆𝑡𝑜𝑡al = is the total sum of squares (variance of the actual values).

### Model Performance Evaluation

The MLP model sachieved commendable performance in predicting the export value of agricultural products three years into the future. The following metrics were gotten after hyperparameter tuning;

**MAE**

Mean: 919240192.0

Median: 1171806976.0

**MSE**

Mean: 4.631e+18

Median: 4.711e+18

**RMSE**

Mean: 1852330240.0

Median: 2170509824.0

**R-squared (R2)**

Mean: 0.309

Median: 0.318

### Model Training and Testing Sets

First off, for the data merge, each of the 12 different datasets we enriched using the “Elements” and “Items” columns or both depending on the case, in such a way that information for years between the ranges 2000 and 2019 was preserved. This was to ensure uniformity across the dataset values and reduce the amount of imputation to be done on missing values later on.

After all the individual datasets were merged, the missing values were filled using time series interpolation to ensure that the interpolation procedure took note of the `Year` and `Area` columns. This was to maintain the time-series integrity of the data and retain it’s robustness.

The data numerical values were also scaled using a standard scaler to prevent some of the features dominating the model’s predictive ability. This to ensure that all the features had equal chances to give weighted contribution to labels prediction rather than some features being skewed due to their dominance larger numerical values.

For the training and test methodology, the training and test data was split in such a way that the values for the last three years was split to the test data, while the rest was split into the training data for every country data. This is to be able to accurately evaluate the ability of the model to predict values up to 3 years into the future. Also, the Pandas, `get\_dummies()` method was used to convert the only categorical column, `Area` into a individual columns for each country in such a way that each column contains a Boolean that indicates of the data was from a particular country. Also, the tensorflow model architecture and activation function of `relu` used across each neural layer link ensured a very detailed and robust model was used on the training and test datasets.

In conclusion, the MLP model demonstrated robust performance in forecasting the export value of crop products three years into the future, as shown by [the calculated performance metrics](#metrics) and the clear distinction between training and testing sets for enough evaluation.

# Chapter 2

## **MLP Model**

### Description of Multilayer Perceptron (MLP) Model

The multilayer perceptron (MLP) model used in this project for predicting the export value of agricultural products three years into the future is a type of artificial neural network (ANN) consisting of multiple layers of interconnected neurons. The architecture of the MLP model comprises an input layer, 8 hidden layers, and an output layer. The neural network was developed in such a way that 5 different models were created. Each of these models has an input layer of 287, but an output layer of one. This was done in this fashion because there were 287 columns in the features data but 5 columns in the target data. Each neural network will train on the 287 columns, but make prediction on just one of the target columns. Then, all the predictions are concatenated together before evaluation is then carried out.

This method was employed to ensure that each of the neural networks were able to learn independently and robustly from the training features. This method also gave the opportunity to combine the predictive ability of 5 different neural network models instead of just using the predictive ability of one.

Activation Function for Output Layer: The activation function used for all the 10 layers of the MLP model was the activation function “RELU”. The “RELU” activation function was employed due to it’s robustness, simplicity, computational efficiency and effectiveness, especially in deep learning architectures. Its mathematical statement can be expressed as:

f(x)=max(0,x)

where;

x is positive, and 0 otherwise. This means that any negative input is transformed to 0, while positive inputs remain unchanged. The main advantages of ReLU can also help to mitigate the vanishing gradient problem, which can occur in deeper neural networks with other activation functions like sigmoid or tanh.which allows the model to output continuous values without constraints.

Loss Function: The loss function used to train this MLP model is the MSE loss function. This function was used because of it’s ability to optimize model's parameters and minimize prediction errors. The MSE loss function can be calculated as:

𝑀𝑆𝐸 = (1/𝑛)∑(𝑦 − 𝑦𝑖)2

Where:

𝑛 = n is the total number of instances.

𝑦 = is the actual export value.

𝑦I = is the predicted export value.

Minimizing the MSE loss function during training can help the model learn to make accurate predictions by adjusting its weights and biases.

Number of Units in Output Layer: The number of units in the output layer of the MLP model corresponds to the number of output nodes, which is one in this case since the model predicts a single value (export value) for each instance.

### Steps to Prevent Overfitting

Overfitting occurs when a model learns the training data too well, capturing noise and irrelevant patterns that do not generalize to unseen data. To prevent overfitting in this MLP model, several techniques and steps were employed including;

* Early Stopping Callbacks: An early stopping callback was created with a patience level of 5 to stop the training process if the
* Early Stopping: Early stopping was employed to stop the training process when the model's performance starts to deteriorate. This prevents the model from overfitting to the training data by avoiding unnecessary training epochs. It usually works on the condition;

𝐿𝑜𝑠𝑠𝑣𝑎𝑙𝑖𝑑𝑎𝑡𝑖𝑜𝑛 ≥ 𝐿𝑜𝑠𝑠𝑝𝑟𝑒𝑣\_𝑒𝑝𝑜𝑐ℎ

* Cross-Validation: Cross-validation is used to assess the model's performance on multiple subsets of the data, helping to validate the model's generalization ability. For this model, a 5-fold cross-validation procedure was used to divide the data into 5 subsets and training the model on different combinations of these subsets in order to improve the ability of the model to accurately learn patterns from the data.
* Epochs: To achieve a well-balanced model that captures patterns without overfitting, the number of training epochs was also very carefully considered. An early stopping callback with a patience level of 4 was implemented, allowing the model to train for up to 50 epochs. This setup ensures that the model is given sufficient time to learn meaningful patterns from the data while preventing it from continuing to train unnecessarily once performance on the validation set plateaus or worsens. This approach optimizes training efficiency and model generalization.

# Chapter Three

## **Derivation of Feature and Labels from Given Data**

The specific step by step methodology on how all the features and labels were selected from the data would be iterated in the data processing part of this report in chapter four.

### Label Derivation

The labels used for the MLP project were gotten from the “Food Trade” dataset. Initially, during data preprocessing, data cleaning and data merge, they were together as columns with the “Final\_merge” dataframe. But before training and testing, the “Fianl\_merge” dataframe was split into features and labels with all of the values relating to export of agricultural products sent to the labels data. It is the labels data that was then split into training and test labels.

The training labels contains all labels of products between year 2000 and 2016 while the test lables contains all labels of products between year 2017 and 2019. There were a total of 5 labels for the training and test label datasets and they include;

* 'Export Value of Fruits and Vegetables (USD)',
* 'Export Value of Non-food Items (USD)',
* 'Export Value of Other food Items (USD)',
* 'Export Value of Sugar and Honey Items (USD)',
* 'Export Value of Tobacco (USD)'

### Selection Rationale for Features

All the features used were features that were robust enough to contain information between years 2000 and 2019 for all the different datsets provides. This was to ensure uniformity and retain the time-series perspective of the dataset. Majority of the features (especially the ones from the `Area` column) came from using the `pandas.get\_dummies()\_` method to transform the categorical features into numericals. The features (287 in number) selected for training and testing includes;

* Year
* Area
* Standard Deviation of Change in Temperature between Dec, Jan and Feb (degree celsius)
* Standard Deviation of Change in Temperature between June, July and August (degree celsius)
* Standard Deviation of Change in Temperature between March, April and May (degree celsius)
* Standard Deviation of Change in Temperature in Meteorological Year (degree celsius)
* Standard Deviation of Change in Temperature between September, October and November (degree celsius)
* Change In Temperature between Dec, Jan and Feb (degree celsius)
* Change In Temperature between June, July and August (degree celsius)
* Change In Temperature between March, April and May (degree celsius)
* Change In Temperature in Meteorological Year (degree celsius)
* Change In Temperature between September, October and November (degree celsius)
* Average Exchange rate (yearly)
* Yearly Yield for Cereals Primary (Tonne/Hectare)
* Yearly Yield for Citrus Fruit Total (Tonne/Hectare)
* Yearly Yield for Fibre Crops Fibre Equivalent (Tonne/Hectare)
* Yearly Yield for Fruit Primary (Tonne/Hectare)
* Yearly Yield for Oilcrops Cake Equivalent (Tonne/Hectare)
* Yearly Yield for Oilcrops Oil Equivalent (Tonne/Hectare)
* Yearly Yield for Pulses Total (Tonne/Hectare)
* Yearly Yield for Roots And Tubers Total (Tonne/Hectare)
* Yearly Yield for Sugar Crops Primary (Tonne/Hectare)
* Yearly Yield for Treenuts Total (Tonne/Hectare)
* Yearly Yield for Vegetables Primary (Tonne/Hectare)
* Country Agricultural Land Mass (Hectares)
* Country Land Mass for Agriculture Only (Hectares)
* Country Arable Land Mass (Hectares)
* Country Area Land Mass (Hectares)
* Country Cropland Mass (Hectares)
* Country Land Area (Hectares)
* Country Land Area Equipped for Irrigation (Hectares)
* Country Permanent Crops Land Area (Hectares)
* Country Meadows and Pastures Total Land Area (Hectares)
* Country Total Land Area for Temporary Crops (Hectares)
* Country Total Land Area for Temporary Fallow (Hectares)
* Country Total Land Area for Temporary Meadows and Pastures (Hectares)
* Area\_Afghanistan
* Area\_Albania
* Area\_Algeria
* Area\_American Samoa
* Area\_Andorra
* Area\_Angola
* Area\_Anguilla
* Area\_Antarctica
* Area\_Antigua and Barbuda
* Area\_Argentina
* Area\_Armenia
* Area\_Aruba
* Area\_Australia
* Area\_Austria
* Area\_Azerbaijan
* Area\_Bahamas
* Area\_Bahrain
* Area\_Bangladesh
* Area\_Barbados
* Area\_Belarus
* Area\_Belgium
* Area\_Belize
* Area\_Benin
* Area\_Bermuda
* Area\_Bhutan
* Area\_Bolivia (Plurinational State of)
* Area\_Bonaire, Sint Eustatius and Saba
* Area\_Bosnia and Herzegovina
* Area\_Botswana
* Area\_Brazil
* Area\_British Virgin Islands
* Area\_Brunei Darussalam
* Area\_Bulgaria
* Area\_Burkina Faso
* Area\_Burundi
* Area\_Cabo Verde
* Area\_Cambodia
* Area\_Cameroon
* Area\_Canada
* Area\_Cayman Islands
* Area\_Central African Republic
* Area\_Chad
* Area\_Channel Islands
* Area\_Chile
* Area\_China
* Area\_China, Hong Kong SAR
* Area\_China, Macao SAR
* Area\_China, Taiwan Province of
* Area\_China, mainland
* Area\_Christmas Island
* Area\_Cocos (Keeling) Islands
* Area\_Colombia
* Area\_Comoros
* Area\_Congo
* Area\_Cook Islands
* Area\_Costa Rica
* Area\_Croatia
* Area\_Cuba
* Area\_Curaçao
* Area\_Cyprus
* Area\_Czechia
* Area\_Côte d'Ivoire
* Area\_Democratic People's Republic of Korea
* Area\_Democratic Republic of the Congo
* Area\_Denmark
* Area\_Djibouti
* Area\_Dominica
* Area\_Dominican Republic
* Area\_Ecuador
* Area\_Egypt
* Area\_El Salvador
* Area\_Equatorial Guinea
* Area\_Eritrea
* Area\_Estonia
* Area\_Eswatini
* Area\_Ethiopia
* Area\_Falkland Islands (Malvinas)
* Area\_Faroe Islands
* Area\_Fiji
* Area\_Finland
* Area\_France
* Area\_French Guiana
* Area\_French Polynesia
* Area\_French Southern Territories
* Area\_Gabon
* Area\_Gambia
* Area\_Georgia
* Area\_Germany
* Area\_Ghana
* Area\_Gibraltar
* Area\_Greece
* Area\_Greenland
* Area\_Grenada
* Area\_Guadeloupe
* Area\_Guam
* Area\_Guatemala
* Area\_Guernsey
* Area\_Guinea
* Area\_Guinea-Bissau
* Area\_Guyana
* Area\_Haiti
* Area\_Holy See
* Area\_Honduras
* Area\_Hungary
* Area\_Iceland
* Area\_India
* Area\_Indonesia
* Area\_Iran (Islamic Republic of)
* Area\_Iraq
* Area\_Ireland
* Area\_Isle of Man
* Area\_Israel
* Area\_Italy
* Area\_Jamaica
* Area\_Japan
* Area\_Jersey
* Area\_Jordan
* Area\_Kazakhstan
* Area\_Kenya
* Area\_Kiribati
* Area\_Kuwait
* Area\_Kyrgyzstan
* Area\_Lao People's Democratic Republic
* Area\_Latvia
* Area\_Lebanon
* Area\_Lesotho
* Area\_Liberia
* Area\_Libya
* Area\_Liechtenstein
* Area\_Lithuania
* Area\_Luxembourg
* Area\_Madagascar
* Area\_Malawi
* Area\_Malaysia
* Area\_Maldives
* Area\_Mali
* Area\_Malta
* Area\_Marshall Islands
* Area\_Martinique
* Area\_Mauritania
* Area\_Mauritius
* Area\_Mayotte
* Area\_Mexico
* Area\_Micronesia (Federated States of)
* Area\_Midway Island
* Area\_Monaco
* Area\_Mongolia
* Area\_Montenegro
* Area\_Montserrat
* Area\_Morocco
* Area\_Mozambique
* Area\_Myanmar
* Area\_Namibia
* Area\_Nauru
* Area\_Nepal
* Area\_Netherlands (Kingdom of the)
* Area\_Netherlands
* Antilles (former)
* Area\_New Caledonia
* Area\_New Zealand
* Area\_Nicaragua
* Area\_Niger
* Area\_Nigeria
* Area\_Niue
* Area\_Norfolk Island
* Area\_North Macedonia
* Area\_Northern Mariana Islands
* Area\_Norway
* Area\_Oman
* Area\_Pakistan
* Area\_Palau
* Area\_Palestine
* Area\_Panama
* Area\_Papua New Guinea
* Area\_Paraguay
* Area\_Peru
* Area\_Philippines
* Area\_Pitcairn
* Area\_Poland
* Area\_Portugal
* Area\_Puerto Rico
* Area\_Qatar
* Area\_Republic of Korea
* Area\_Republic of Moldova
* Area\_Romania
* Area\_Russian Federation
* Area\_Rwanda
* Area\_Réunion
* Area\_Saint Barthélemy
* Area\_Saint Helena, Ascension and Tristan da Cunha
* Area\_Saint Kitts and Nevis
* Area\_Saint Lucia
* Area\_Saint Martin (French part)
* Area\_Saint Pierre and Miquelon
* Area\_Saint Vincent and the Grenadines
* Area\_Samoa
* Area\_San Marino
* Area\_Sao Tome and Principe
* Area\_Saudi Arabia
* Area\_Senegal
* Area\_Serbia
* Area\_Serbia and Montenegro
* Area\_Seychelles
* Area\_Sierra Leone
* Area\_Singapore
* Area\_Sint Maarten (Dutch part)
* Area\_Slovakia
* Area\_Slovenia
* Area\_Solomon Islands
* Area\_Somalia
* Area\_South Africa
* Area\_South Georgia and the South Sandwich Islands
* Area\_South Sudan
* Area\_Spain
* Area\_Sri Lanka
* Area\_Sudan
* Area\_Sudan (former)
* Area\_Suriname
* Area\_Svalbard and Jan Mayen Islands
* Area\_Sweden
* Area\_Switzerland
* Area\_Syrian
* Arab Republic
* Area\_Tajikistan
* Area\_Thailand
* Area\_Timor-Leste
* Area\_Togo
* Area\_Tokelau
* Area\_Tonga
* Area\_Trinidad and Tobago
* Area\_Tunisia
* Area\_Turkmenistan
* Area\_Turks and Caicos Islands
* Area\_Tuvalu
* Area\_Türkiye
* Area\_Uganda
* Area\_Ukraine
* Area\_United Arab Emirates
* Area\_United Kingdom of Great Britain and Northern Ireland
* Area\_United Republic of Tanzania
* Area\_United States Virgin Islands
* Area\_United States of America
* Area\_Uruguay
* Area\_Uzbekistan
* Area\_Vanuatu
* Area\_Venezuela (Bolivarian Republic of)
* Area\_Viet Nam
* Area\_Wake Island
* Area\_Wallis and Futuna Islands
* Area\_Western Sahara
* Area\_Yemen
* Area\_Zambia
* Area\_Zimbabwe
* Area\_Åland Islands

Values between 2000 and 2016 were chosen as features because they provide a time-series perspective on export performance, allowing the model to capture seasonality, trends, and cyclical patterns in agricultural product exports. This information is important for forecasting future export values accurately. This also enables the model to learn from past fluctuations and dynamics in the values, enabling it to make predictions that consider the historical context and

## Chapter Four

## **Preprocessing of Features for Model Building**

### Feature Engineering and Data Merge

These were the Feature engineering and data merge tasks done on each datasets;

**`consumer\_price\_indicators` data preparation**

* `Note:` `consumer\_pi` = `consumer\_price\_indicators`
* - Drop the `Domain Code` and `Domain` columns.
* - Use the `Item` column to split the data into 2 different datasets.
* - `Consumer Price, Food indices (2015=100)`
* - `Food price inflation`
* - Drop the `Item Code` column.
* - Drop the `Element Code` and `Element` columns.
* - Drop the `Unit` column.
* - Drop the `Flag` and `Flag Description` columns.
* - Drop the `Note` column.
* - For each of these datasets drop the `Item` column and change the `Values` column into;
* - `Consumer Price for the Year`
* - `Avg Food price inflation(%) for the Year`
* - For the `Months` column, aggregate the values by finding the total of all the months for each year for the `Consumer Price for the Year` and aggregate by average for the `Food price inflation(%) for the Year`. Drop the `Months` column after aggregating.
* - merge the datasets back on common columns i.e (`Area Code`, `Area`, `Year Code`, `Year`)
* - To get the final `consumer\_pi` dataset with columns (`Area Code`, `Area`, `Year Code`, `Year`, `Consumer Price for the Year`, `Avg Food price inflation for the Year`)

**`crop\_production\_indicators` data preparation**

* Note: `crop\_pi` = `crop\_production\_indicators`
* - Drop the `Domain Code` and `Domain` columns.
* - Drop the `Item Code (CPC)` column.
* - Drop the `Element Code` and `Element` columns.
* - Drop the `Unit` column.
* - Drop the `Flag` and `Flag Description` columns.
* - Use the `Item` column to split the data into 11 different datasets.
* - `Cereals, primary`
* - `Citrus fruit, Total`
* - `Fibre crops, Fibre Equivalent`
* - `Fruit, Primary`
* - `Oil crops, Cake Equivalent`
* - `Oil crops, Oil Equivalent`
* - `Pulses, Total`
* - `Roots and Tubers, Total`
* - `Sugar Crops Primary`
* - `Treenuts, Total`
* - `Vegetables, Primary`
* - For each of these datasets drop the `Item` column and change the `Values` column into;
* - `Yearly Yield for Cereals, primary`
* - `Yearly Yield for Citrus fruit, Total`
* - `Yearly Yield for Fibre crops, Fibre Equivalent`
* - `Yearly Yield for Fruit, Primary`
* - `Yearly Yield for Oil crops, Cake Equivalent`
* - `Yearly Yield for Oil crops, Oil Equivalent`
* - `Yearly Yield for Pulses, Total`
* - `Yearly Yield for Roots and Tubers, Total`
* - `Yearly Yield for Sugar Crops Primary`
* - `Yearly Yield for Treenuts, Total`
* - `Yearly Yield for Vegetables, Primary`
* - Merge the datasets back on common columns i.e (`Area Code`, `Area`, `Year Code`, `Year`)
* ### `Emissons` data preparation
* - Split data into two `Emissions from crops` and `Emissions from drained organic soils`.
* #### `crop\_emissions` data preparation
* - Drop the `Domain` and `Domain Code` columns.
* - Use the `Element` column to split into two, `crop\_ch4\_emissions` and `crop\_n2o\_emissions`.
* ##### `crop\_ch4\_emissions` data preparation
* - Drop the `Domain` and `Domain Code` columns.
* - Drop the `Element` and `Element Code` columns.
* - Drop the `Item` and `Item code` columns.
* - Drop the `Source` and `Source Code` columns.
* - Change the `Value` column to `Crop ch4 emissions (Tonnes) for the year` and multiply all values by 1000.
* - Drop the `Unit` column.
* - Drop the `Flag Description` and `Note` column.
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Crop ch4 emissions (Tonnes) for the Year`
* ##### `crop\_n20\_emissions` data preparation
* - Drop the `Domain` and `Domain Code` columns.
* - Drop the `Element` and `Element Code` columns.
* - Drop the `Item` and `Item code` columns.
* - Drop the `Source` and `Source Code` columns.
* - Change the `Value` column to `Crop n2o emissions (Tonnes) for the year` and multiply all values by 1000.
* - Drop the `Unit` column.
* - Drop the `Flag Description` and `Note` column.
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Crop n2o emissions (Tonnes) for the Year`

**`crop\_emissions` data preparation**

* - merge the `crop\_ch4\_emissions` and `crop\_n2o\_emissions` data together on common columns i.e (`Area Code`, `Area`, `Year Code`, `Year`)
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Crop n2o emissions (Tonnes) for the Year`, `Crop ch4 emissions (Tonnes) for the Year`
* #### `organic\_emissions` data preparation
* - Drop the `Domain Code` and `Domain` columns.
* - Split into two `co2\_organic\_emissions` and second `n2o\_organic\_emissions`
* ##### `co2\_organic\_emissions` data preparation
* - Drop the `Domain` and `Domain Code` columns.
* - Drop the `Element` and `Element Code` columns.
* - Drop the `Item` and `Item code` columns.
* - Drop the `Source` and `Source Code` columns.
* - Change the `Value` column to `Organic co2 emissions (Tonnes) for the year` and multiply all values by 1000.
* - Drop the `Unit` column.
* - Drop the `Flag Description` and `Note` column.
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Organic co2 emissions (Tonnes) for the year`
* ##### `n2o\_organic\_emissions` data preparation
* - Drop the `Domain` and `Domain Code` columns.
* - Drop the `Element` and `Element Code` columns.
* - Drop the `Item` and `Item code` columns.
* - Drop the `Source` and `Source Code` columns.
* - Change the `Value` column to `Organic n2o emissions (Tonnes) for the year` and multiply all values by 1000.
* - Drop the `Unit` column.
* - Drop the `Flag Description` and `Note` column.
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Organic n2o emissions (Tonnes) for the year`
* #### `organic\_emissions` data preparation complete
* - merge the `ch4\_organic\_emissions` and `n2o\_organic\_emissions` data together on common columns i.e (`Area Code`, `Area`, `Year Code`, `Year`)
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Organic n2o emissions (Tonnes) for the year`, `Organic co2 emissions (Tonnes) for the year`
* ### `Emissions` data preparation complete
* - merge the `crop\_emissions` and `organic\_emissions` data together on common columns i.e (`Area Code`, `Area`, `Year Code`, `Year`)
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Crop n2o emissions (Tonnes) for the Year`, `Crop ch4 emissions (Tonnes) for the Year`, `Organic n2o emissions (Tonnes) for the year`, `Organic co2 emissions (Tonnes) for the year`

**`Employment` data preparation**

* - Drop the `Domain Code` and `Domain` columns.
* - Drop the `Indicator Code` column.
* - Drop the `Sex Code` and `Sex` columns.
* - Drop the `Element Code` and `Element` columns.
* - Drop the `Source Code` and `Source` columns.
* - Drop the `Flag` and `Flag Description` column.
* - Drop the `Note` column.
* - Use the `Indicator` column to split the `Employment` field into two datasets `weekly\_hours` and `agric\_employment` on the values;
* - `Mean weekly hours worked per employed person in agriculture, forestry and fishing`.
* - `Employment in agriculture, forestry and fishing`. Proceed to drop the `Indicator` columns for both.
* - Incorporate the `Unit` column to the `Value` column for both. (`No` for normal number, `1000 No` for thousand number, multiply all the entries in the `Value` column for the `agric\_employment` dataset). Rename the `Values` column for both to `Mean weekly hours worked per employed person in agriculture, forestry and fishing` and `Employment in agriculture, forestry and fishing` respectively.
* - Merge on similar columns (`Area Code`, `Area`, `Year Code`, `Year`)
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Mean weekly hours worked per employed person in agriculture, forestry and fishing` and `Employment in agriculture, forestry and fishing`.

**`Exchange rates` data preparation**

* - Drop the `Domain Code` and `Domain` column.
* - Drop the `ISO Currency Code (FAO)` column.
* - Drop the `Currency` column.
* - Drop the `Element Code` and`Element` column.
* - Drop the `Months Code` column.
* - Drop the `Flag` and `Flag Description` column.
* - Drop the `Unit` column.
* - Change the `Value` column to `Average Exchage rate (yearly)`.
* - Aggregate the `Average Exchage rate (yearly)` column to average yearly value by using the `Months` column and drop the `Months` column.
* - Remaining - `Area Code`, `Area`, `Year Code`, `Year`, `Average Exchage rate (yearly)`.

**`Fertilizers Use` data preparation**

* - Drop the `Domain Code` and `Domain` columns.
* - Drop the `Element Code` and `Element` columns.
* - Drop the `Year Code` column.
* - Drop the `Item Code` column.
* - Drop the `Flag` and `Flag Description` columns.
* - Change the `Item` column to `Fertilizer Type Used`.
* - Drop the `Unit` column.
* - Use the `Item` column to split the dataset into 24 different datasets on.
* - `Ammonia, anhydrous`
* - `Ammonium nitrate (AN)`
* - `Ammonium sulphate`
* - `Calcium ammonium nitrate (CAN) and other mixtures with calcium carbonate`
* - `Diammonium phosphate (DAP)`
* - `Fertilizers n.e.c.`
* - `Monoammonium phosphate (MAP))`
* - `NPK fertilizers`
* -`Other nitrogenous fertilizers, n.e.c.``
* - `Other NK compounds`
* - `Other NP compounds`
* - `Other phosphatic fertilizers, n.e.c`
* - `Other potassic fertilizers, n.e.c`
* - `Phosphate rock`
* - `PK compounds`
* - `Potassium chloride (muriate of potash) (MOP)`
* - `Potassium nitrate`
* - `Potassium sulphate (sulphate of potash) (SOP)`
* - `Sodium nitrate`
* - `Superphosphates above 35%`
* - `Superphosphates, other`
* - `Urea`
* - `Urea and ammonium nitrate solutions (UAN)`
* - Drop the `Item` column and rename the `Value` column to `Amnt of fertilizers used (Tonnes)`.
* - `Amnt of Ammonia, anhydrous fertilizers used (Tonnes)`
* - `Amnt of Ammonium nitrate (AN) fertilizers used (Tonnes)`
* - `Amnt of Ammonium sulphate fertilizers used (Tonnes)`
* - `Amnt of Calcium ammonium nitrate (CAN) and other mixtures with calcium carbonate fertilizers used (Tonnes)`
* - `Amnt of Diammonium phosphate (DAP) fertilizers used (Tonnes)`
* - `Amnt of ertilizers n.e.c. fertilizers used (Tonnes)`
* - `Amnt of Monnammonium phosphate (MAP) fertilizers used (Tonnes)`
* - `Amnt of NPK fertilizers used (Tonnes)`
* - `Amnt of Other nitrogenous fertilizer, n.e.c. fertilizers used (Tonnes)`
* - `Amnt of Other NK compounds fertilizers used (Tonnes)`
* - `Amnt of Other NP compounds fertilizers used (Tonnes)`
* - `Amnt of Other phosphatic fertilizers, n.e.c fertilizers used (Tonnes)`
* - `Amnt of Other potassic fertilizers, n.e.c fertilizers used (Tonnes)`
* - `Amnt of Phosphate rock fertilizers used (Tonnes)`
* - `Amnt of PK compounds fertilizers used (Tonnes)`
* - `Amnt of Potassium chloride (muriate of potash) (MOP) fertilizers used (Tonnes)`
* - `Amnt of Potassium nitrate fertilizers used (Tonnes)`
* - `Amnt of Potassium sulphate (sulphate of potash) (SOP) fertilizers used (Tonnes)`
* - `Amnt of Sodium nitrate fertilizers used (Tonnes)`
* - `Amnt of Superphosphates above 35% fertilizers used (Tonnes)`
* - `Amnt of Superphosphates, other fertilizers used (Tonnes)`
* - `Amnt of Urea fertilizers used (Tonnes)`
* - `Amnt of Urea and ammonium nitrate solutions (UAN) fertilizers used (Tonnes)`
* - Aggregate the `Average Exchage rate (yearly)` column to average yearly value by using the `Months` column and drop the `Months` column.

**`food balances indicators` data preparation**

* - Drop the `Domain Code`, `Domain`, `Area Code (M49)`, `Element Code`, `Item Code (FBS)`, columns
* - Use the element column to split the data into 5 different datasets (`export\_qty`, `import\_qty`, `food`, `losses`, `non\_food\_uses`) based on unique values;
* - `Export Quantity`
* - `Import Quantity`
* - `Food`
* - `Losses`
* - `Other uses (non-food)`.

**`Food Security Indicators` data preparation**

* - Drop the `Domain Code`, `Domain`, `Area Code (M49)`, `Element`, `Element Code`, `Item Code`, `Flag`, and `Flag Description` columns,
* - Use the `Item` column to split the dataframe into 4 dataframes (`food\_prod\_variability`, `food\_supply\_variability`, `political\_stability`, `anemia\_prevalence`) on the unique values;
* - `Per capita food production variability (constant 2014-2016 thousand int$ per capita)`
* - for this `food\_prod\_variability` dataframe, drop the `Item` and `Unit` column.
* - Multiply the `Value` column by 1000.
* - Rename the `Value` column to `Per Capita Food Production Value (International Dollar)`
* - `Per capita food supply variability (kcal/cap/day)`
* - for this `food\_supply\_variability` dataframe, , drop the `Item` and `Unit` column.
* - Multiply the `Value` column by 1000.
* - Rename the `Value` column to `Per Capita Food Supply Value (Calories per Capita Per Day)`
* - `Political stability and absence of violence/terrorism (index)`
* - for this`political\_stability` dataframe, , drop the `Item` and `Unit` column.
* - Rename the `Value` column to `Political stability and absence of violence/terrorism (index)`
* - `Prevalence of anemia among women of reproductive age (15-49 years)`
* - for this `anemia\_prevalence`) dataframe, , drop the `Item` and `Unit` column.
* - Rename the `Value` column to `Percentage Prevalence of anemia among women of reproductive age (15-49 years)`
* - Merge the four dataframes (`food\_prod\_variability`, `food\_supply\_variability`, `political\_stability`, and `anemia\_prevalence`) on common columns `Year` and `Area` to get the `fsi\_final\_merge` dataframe.
* - Export this dataframe to `cleaned\_fsi` csv file.

**`Food Trade Indicators` data preparation**

* - Drop the `Domain Code`, `Domain`, `Area Code (M49)`, `Year Code`, `Item Code (CPC)`, `Element Code`, `Unit`, `Flag`, `Flag Description`, `Note`
* - Multiply all values in the `Value` column by `1000`
* - Use the `Element` column to split it into 2 different dataframes(`export\_value` and `import\_value`) based on unqiue values;
* - `Export Value`
* - `Import Value`
* - For the `export\_value` dataframe, drop the `Element` column.
* - Use the `Item` column to split into 5 dataframes (`fruits\_and\_veggies`, `non\_food`, `other\_food`, `sugar\_and\_honey`, `tobacco`) based on information-rich values;
* - `Fruit and Vegetables`
* - `Non-food`
* - `Other food`
* - `Sugar and Honey`
* - `Tobacco`
* - Drop the `Item` column for all 5 dataframes.
* - Rename the `Values` column for each of the 5 dataframes as follows;
* - `Export Value of Fruits and Vegetables`
* - `Export Value of Non-food Items`
* - `Export Value of Other food Items (USD)`
* - `Export Value of Sugar and Honey Items (USD)`
* - `Export Value of Tobacco (USD)`
* - merge them back on common columns `Year` and `Area` to get `export\_merged` dataframe.
* - For the `import\_value` dataframe, drop the `Element` column.
* - Use the `Item` column to split the dataframe into 12 dataframes(`import\_alcoholic\_beverages`, `import\_cereals\_and\_prep`, `import\_dairy`, `import\_fats\_and\_oils`, `import\_fruits\_and\_veggies`, `import\_meats`, `import\_non\_alcohols`, `import\_non\_edible\_fats`, `import\_non\_food`, `import\_other\_food`, `import\_sugar\_and\_honey`, `import\_tobacco`);
* - `Alcoholic Beverages`
* - `Cereals and Preparations`
* - `Dairy Products and Eggs`
* - `Fats and Oils (excluding Butter)`
* - `Fruit and Vegetables`
* - `Meat and Meat Preparations`
* - `Non-alcoholic Beverages`
* - `Non-edible Fats and Oils`
* - `Non-food`
* - `Other food`
* - `Sugar and Honey`
* - `Tobacco` respectively
* - Drop the `Item` column for all 12 dataframes.
* - Rename the `Values` column for each of the 12 dataframes to;
* - `Import Value of Alcoholic Beverages (USD)`
* - `Import Value of Cereals and Preparations (USD)`
* - `Import Value of Dairy Products and Eggs (USD)`
* - `Import Value of Fats and Oils (excluding Butter) (USD)`
* - `Import Value of Fruit and Vegetables (USD)`
* - `Import Value of Meat and Meat Preparations (USD)`
* - `Import Value of Non-alcoholic Beverages (USD)`
* - `Import Value of Non-edible Fats and Oils (USD)`
* - `Import Value of Non-food (USD)`
* - `Import Value of Other food (USD)`
* - `Import Value of Sugar and Honey (USD)`
* - `Import Value of Tobacco (USD)`
* - merge the 12 dataframes back on common columns `Year` and `Area` to get `import\_merged` dataframe.
* - merge the `import\_merged` and `export\_merged` dataframes back on common columns `Year` and `Area` to get `final\_fti\_merged` dataframe.
* - Export to csv file as `cleaned\_fti`.

**`Foreign Direct Investment` data preparation**

* - Drop the `Domain`, `Domain Code`, `Area Code (M49)`, `Element Code`, `Element`, `Item Code`, `Year Code`, `Unit`, `Flag`, `Flag Description`, `Note`
* - Multiply each value in the `Value` column by 1000000
* - The `Item` column has 6 different unique values;
* - `FDI inflows to Agriculture, Forestry and Fishing`
* - `FDI inflows to Food, Beverages and Tobacco`
* - `FDI outflows to Agriculture, Forestry and Fishing`
* - `FDI outflows to Food, Beverages and Tobacco`
* - `Total FDI inflows`
* - `Total FDI outflows`
* - Due to inconsistencies, only the `Total FDI inflows` and `Total FDI outflows` values in the `Item` column will be considered.
* - Use these two values `Total FDI inflows` and `Total FDI outflows` to get two dataframes (total\_fdi\_inflows) and (total\_fdi\_outflows).
* - Dorop the `Item` column for both and rename the `Value` column for each to;
* - `Total FDI Inflows (Dollars)`
* - `Total FDI Outflows (Dollars)`
* - Merge the `total\_fdi\_inflows` and `total\_fdi\_outflows` on common columns `Area` and `Year` to get final dataframe `fdi\_merged`.
* - Export to csv called `cleaned\_fdi`.

**`Land Temperature Change` data preparation**

* - Drop the `Domain Code`, `Domain`, `Area Code (M49)`, `Element Code`, `Months Code`, `Year Code`, `Unit`, `Flag`, `Flag Description` columns.
* - Use the `Element` to split into two dataframes (`standard\_dev` and `temp\_change`) due to unique values;
* - `Standard Deviation`
* - `Temperature change`
* - For the `standard\_dev` dataframe, use the `Months` column to split it into 5 dataframes (`dec\_jan\_feb`, `jun\_jul\_aug`, `mar\_apr\_may`, `meterorlogical\_year`, `sep\_oct\_nov`) based on unique values;
* - `Dec-Jan-Feb`
* - `Jun-Jul-Aug`
* - `Mar-Apr-May`
* - `Meteorological year`
* - `Sep-Oct-Nov`
* - For these five dataframes (`sd\_dec\_jan\_feb`, `sd\_jun\_jul\_aug`, `sd\_mar\_apr\_may`, `sd\_meterorlogical\_year`, `sd\_sep\_oct\_nov`) drop the `Months` column and rename the `Values` column to;
* - `Standard Deviation of Change In Temperature between Dec, Jan and Feb (℃)`
* - `Standard Deviation of Change In Temperature between June, July and August (℃)`
* - `Standard Deviation of Change In Temperature between March, April and May (℃)`
* - `Standard Deviation of Change In Temperature in Metereological Year (℃)`
* - `Standard Deviation of Change In Temperature between September, October and December (℃)` repectively.
* - Merge them all back on the `Area` and `Year` columns to form the `sd\_merged` dataframe.
* - For the `temp\_change` dataframe, use the `Months` column to split it into 5 dataframes (`dec\_jan\_feb`, `jun\_jul\_aug`, `mar\_apr\_may`, `metereological\_year`, `sep\_oct\_nov`) based on unique values;
* - `Dec-Jan-Feb`
* - `Jun-Jul-Aug`
* - `Mar-Apr-May`
* - `Metereological year`
* - `Sep-Oct-Nov`
* - For these five dataframes (`temp\_dec\_jan\_feb`, `temp\_jun\_jul\_aug`, `temp\_mar\_apr\_may`, `temp\_metereological\_year`, `temp\_sep\_oct\_nov`) drop the `Months` column and rename the `Values` column to;
* - `Change In Temperature between Dec, Jan and Feb (℃)`
* - `Change In Temperature between June, July and August (℃)`
* - `Change In Temperature between March, April and May (℃)`
* - `Change In Temperature in Meterological Year (℃)`
* - `Change In Temperature between September, October and December (℃)` repectively.
* - Merge them all back on the `Area` and `Year` columns to form the `temp\_merged` dataframe.
* - Merge the `sd\_merged` and `temp\_merged` dataframes on `Year` and `Area` columns to form the `Land\_use\_merged` dataframe.
* - Export this to a dataframe called `cleaned\_land\_use` dataframe.

**`Land Use` data preparation**

* - Drop the `Domain Code`, `Domain`, `Area Code`, `Element Code`, `Element`, `Item Code`, `Year Code`, `Flag`, `Flag Description`, `Note` columns.
* - Drop the `Unit` column and multiply each value in the `Value` colun by 1000
* - Use the `Item` column to split the dataframe into 12 Different dataframes (`agric\_land`, `agric`, `irrigated\_agric\_area`, `arable\_land`, `country\_area`, `cropland`, `irrigated\_cropland\_area`, `farm\_buildings`,`irrigated\_forestry\_area`, `land\_area`, `irrigated\_land\_area`, `equipped\_irrigated\_land\_area`, `cultivated \_perm`, `natural\_perm`, `irrigated\_perm`, `permanent\_crops`, `permanent\_meadows`, `temp\_crops`, `temp\_fallow`, `temp\_meadows`) based on the following unique values respectively;
* - `Agricultural land`
* - `Agriculture`
* - `Arable land`
* - `Country Area`
* - `Cropland`
* - `Land area`
* - `Land area actually irrigated`
* - `Land area equipped for irrigation`
* - `Permanent crops`
* - `Permanent meadows and pastures`
* - `Temporary crops`
* - `Temporary fallow`
* - `Temporary meadows and pastures`
* - For each of these 12 dataframes, rename the `Value` columns to;
* - `Country Agricultural Land Mass (Hectares)`
* - `Country Land Mass for Agriculture Only (Hectares)`
* - `Country Arable Land Mass (Hectares)`
* - `Country Area Land Mass (Hectares)`
* - `Country Cropland Mass (Hectares)`
* - `Country Land Area (Hectares)`
* - `Country Land Area Actually Irrigates (Hectares)`
* - `Country Land Area Equipped for Irrigation (Hectares)`
* - `Country Permanent Crops Land Area (Hectares)`
* - `Country Meadows and Pastures Total Land Area (Hectares)`
* - `Country Total Land Area for Temporary Crops (Hectares)`
* - `Country Total Land Area for Temporary Fallow (Hectares)`
* - `Country Total Land Area for Temporary Meadows and Pastures (Hectares)`
* - Then merge the 20 dataframes on the `Year` and `Area` columns to get the final merge dataframe called `land\_use\_final\_merged`
* - Export to a csv file called `land\_use\_cleaned`.

**`Pesticides Use` data preparation**

* - Drop the `Domain Code`, `Domain`, `Area Code (M49)`, `Element Code`, `Item Code`, `Year Code`, `Flag`, `Flag Description` and `Note` columns.
* - Use the `Element` columns to split it into three dataframes (`agricultural\_use`, `per\_area cropland`, and `per\_agric\_value`) based on the unique values;
* - `Agricultural Use`,
* - `Use per area of cropland`,
* - `Use per value of agricultural production`.
* - For the `agricultural\_use` dataframe, use the `Item` column to split it into 7 different dataframes (`pesticides`, `insecticides`, `herbicides`, `fungicides\_bactericides`, `fungicides\_seed\_treatments`, `insecticides\_seed\_treatments`, `rodenticides`) based on the values;
* - `Pesticides (Total)`
* - `Insecticides`
* - `Herbicides`
* - `Fungicides and Bactericides`
* - `Fungicides – Seed treatments`
* - `Insecticides – Seed Treatments`
* - `Rodenticides`, respectively.
* - For each of these 7 dataframes, drop the `Item` and `Element` columns, rename the `Value` column for each to;
* - `Total Amount of Pesticides (Total) Used for Agriculture`
* - `Total Amount of Insecticides Used (Tonnes) for Agriculture`
* - `Total Amount of Herbicides Used (Tonnes) for Agriculture`
* - `Total Amount of Fungicides and Bactericides Used (Tonnes) for Agriculture`
* - `Total Amount of Fungicides – Seed treatments Used (Tonnes) for Agriculture`
* - `Total Amount of Insecticides – Seed Treatments Used (Tonnes) for Agriculture`
* - `Total Amount of Rodenticides Used (Tonnes) for Agriculture`, respectively.
* - Merge back all the 7 dataframes on common columns `Year` and `Area` to get the `agricultural\_use\_merged` dataframe.
* - For the `per\_area cropland` dataframe, drop the `Item` and `Element` columns. Then divide all the values in the `Value` column by 1000 (to convert from kg/hectare to tonne/hectare). Then rename the `Value` column to `Total Amount of Pesticides(Tonnes) Used per Hectare of Cropland`.
* - For the `per\_agric\_value` dataframe, drop the `Item` and `Element` columns. Then divide all the values in the `Value` column by 1000000 (to convert from gram/dollars to Tonne/dollars). Then rename the `Value` column to `Total Amount of Pesticides(Tonnes) Used per per Value of Agricultural Production`.
* - Merge the `agricultural\_use\_merged`, `per\_area cropland`, and `per\_agric\_value` dataframes on common columns `Year` and `Area` to get the `pesticides\_merged` dataframe.
* - Export to `pesticides\_cleaned` csv file.

**Final Merge of All Datasets**

* ### Merge all the cleaned dataframes
* - `cleaned\_land\_temp`
* - `cleaned\_exchange\_rate`
* - `cleaned\_employment`
* - `cleaned\_emissions`
* - `cleaned\_crop\_pi`
* - `cleaned\_consumer\_pi`
* - `cleaned\_fsi`
* - `cleaned\_fti`
* - `cleaned\_fdi`
* - `cleaned\_land\_use`
* - `cleaned\_pesticides`
* on common columns `Area` and `Year`.

### Handling Missing Values

After the final merge, there were a lot of missing values most especially due to discrepancies in the `Year` and `Area` columns for the datasets. Some columns had year values far back to 1980 for some countries while a lot of other columns has values just starting from 2000. For this reason, year values between 2000 and 2019 only were selected to reduce the number of missing values.

Even after this filter, some columns still had missing values up to 2000 for a total row count of 4000+. Due to this, columns with missing values above 1000 rows was filtered out to reduce inaccuracy in values coming from imputation. After this second filter, the missing values were then filled using time-series based interpolation method. This was done to maintain the date-time series integrity of the data.

### Encoding Categorical Variables

After all missing values have been imputed, the categorical `Area` column was transformed to numerical by using the pandas `pd.get\_dummies()` method`. This was what added additional features to our model data.

### Feature Scaling

After categorical transformation, feature scaling was done on all numerical columns (excluding the Boolean columns added by the ‘pd.get\_dummies()’ method) using the scikit-learn standard scaler. This was done to combat presence of any outliers in the datasets and also ensure that the model training pattern was not skewed towards features with higher values only.

# Chapter Five

## **Conclusion**

### Feature Importance

The important features that were indispensable for the model to perform well on the data was gotten using randomforestregressor models. These were the features and their weighted contributions to model performance;

* Yearly Yield for Fruit Primary (Tonne/Hectare) : 0.238
* Area\_Brazil : 0.093
* Year\_numeric : 0.078
* Country Total Land Area for Temporary Meadows and Pastures (Hectares) : 0.055
* Area\_Germany : 0.042
* Area\_Netherlands (Kingdom of the) : 0.038
* Country Land Area Equipped for Irrigation (Hectares) : 0.032
* Yearly Yield for Treenuts Total (Tonne/Hectare) : 0.032
* Yearly Yield for Roots And Tubers Total (Tonne/Hectare) : 0.029
* Yearly Yield for Vegetables Primary (Tonne/Hectare) : 0.027

The top 10 most important features, as determined by feature importance scores, provide valuable insights into the factors driving the model's predictions for export values of agricultural products.

Yearly Yield for Fruit Primary (Tonne/Hectare) emerges as the most critical feature, indicating that the production yield of primary fruits per hectare significantly influences export values.

Other important features include specific geographical areas such as Brazil and Germany, along with agricultural yield metrics for various crops.

### Conclusion and Recommendation

Due to these metrics gotten:

MAE: 919240192.0, MSE: 4.631e+18, RMSE: 1852330240.0 and R-squared (R2): 0.309,

we can see that the model's performance is not ideal, as indicated by the relatively high errors (MAE, MSE, RMSE) and the moderate R-squared score value. While it can capture some patterns in the data, it struggles to make accurate predictions consistently. Some further approach that can be taken to improve the model’s performance includes;

* Getting more robust, consistent and informative data. Due to the fact that the values across the different data files were not consistent, only data values between year 2000 and 2019 were able to be considered. This lead to loss of valuable data. Also even after selecting data across only 19 years, there were still numerous missing data which had to be imputed. The large amount of imputed data (especially in the target columns) can greatly affect the time-series quality of the data.
* More informative features, addition of more informative features and datasets to the dataset files would also greatly improve the ability of the neural networks model to learn patterns from the data.
* More time: Due to the time deadline nature of the project, enough robust training methodologies and tuning algorithms were not able to be implemented. With more time and more in-depth training and tuning strategies, the performance of the models in capturing patterns from the dataset would improve.
* Domain Knowledge Integration: Consulting a domain expert on international exports can also give more insights into how best feature engineering can be done to improve model scores on the data.

By implementing these strategies, the model can be refined to better capture the underlying patterns and dynamics in agricultural export values, ultimately leading to improved performance and more accurate predictions in future tests and real-world applications.