FORECASTING MALAYSIAN RICE PRODUCTION USING HISTORICAL CLIMATE DATA AND MACHINE LEARNING ALGORITHMS

NURHAFIZAH BINTI MOHD YUNOS

UNIVERSITI TEKNOLOGI MALAYSIA

UNIVERSITI TEKNOLOGI MALAYSIA
DECLARATION OF Choose an item.

I declare that this thesis is classified as:
(If none of the options are selected, the first option will be chosen by default)

I acknowledged the intellectual property in the Choose an item. belongs to Universiti Teknologi Malaysia, and I agree to allow this to be placed in the library under the following terms :
This is the property of Universiti Teknologi Malaysia
The Library of Universiti Teknologi Malaysia has the right to make copies for the purpose of only.
The Library of Universiti Teknologi Malaysia is allowed to make copies of this Choose an item. for academic exchange.

NOTES: If the thesis is CONFIDENTIAL or RESTRICTED, please attach a letter from the organisation with the period and reasons for confidentiality or restriction

This letter should be written by a supervisor and addressed to Perpustakaan UTM. A copy of this letter should be attached to the thesis.

Date:
Librarian
Jabatan Perpustakaan UTM,
Universiti Teknologi Malaysia,
Johor Bahru, Johor

Sir,

CLASSIFICATION OF THESIS AS RESTRICTED/CONFIDENTIAL
TITLE: Click or tap here to enter text.
AUTHOR'S FULL NAME: Click or tap here to enter text.

Please be informed that the above-mentioned thesis titled _________ should be classified as RESTRICTED/CONFIDENTIAL for three (3) years from the date of this letter. The reasons for this classification are
(i)
(ii)
(iii)

Thank you.

Yours sincerely,

SIGNATURE:
NAME:
ADDRESS OF SUPERVISOR:

“Choose an item. Hereby declare that Choose an item has read this . Choose an item.
And in Choose an item.
Opinion: Choose an item. is sufficient in terms of scope and quality for the
award of the degree of Choose an item.”

Signature : ______________________________
Name of Supervisor I :
Date :

Signature : ______________________________
Name of Supervisor II :
Date :

Signature : ______________________________
Name of Supervisor III :
Date :

Declaration of Cooperation

This is to confirm that this research has been conducted through a collaboration. Click or tap here to enter text. And click or tap here to enter text.

Certified by:

Signature :
Name :
Position :
Official Stamp
Date

* This section is to be filled up for those with industrial collaboration

Pengesahan Peperiksaan

Tesis ini telah diperiksa dan diakui oleh:

Nama dan Alamat Pemeriksa Luar :

Nama dan Alamat Pemeriksa Dalam :

Nama Penyelia Lain (jika ada) :

Disahkan oleh Timbalan Pendaftar di Fakulti:

Tandatangan :

Nama :

Tarikh :

FORECASTING MALAYSIAN RICE PRODUCTION USING HISTORICAL CLIMATE DATA AND MACHINE LEARNING ALGORITHMS

NURHAFIZAH BINTI MOHD YUNOS

Choose an item. Submitted in Choose an item. of the requirements for the award of the degree of Choose an item.

School of Education
Faculty of Social Sciences and Humanities
Universiti Teknologi Malaysia

JUNE 2025

DECLARATION

I declare that this is Choose an item. Entitled *"title of the thesis"* is the result of my research, except as cited in the references. Choose an item. Has not been accepted for any degree and is not concurrently submitted in candidature for any other degree.

Signature : ....HAFIZAH...............................

Name : NURHAFIZAH BINTI MOHD YUNOS

Date : 30 June 2025

ACKNOWLEDGEMENT

In preparing this thesis, I was in contact with many people, researchers, academicians, and practitioners. They have contributed towards my understanding and thoughts. In particular, I wish to express my sincere appreciation to my lecturer, Prof. Madya. Ts. Dr. Mohd Shahizan bin Othman, for encouragement, guidance, criticism and friendship.

I am also indebted to Universiti Teknologi Malaysia (UTM) for funding my Ph.D study. Librarians at UTM. My fellow postgraduate student should also be recognised for their support. My sincere appreciation also extends to all my colleagues and others who have assisted on various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. I am grateful to all my family members.

ABSTRACT

Accurate forecasting of rice (paddy) production is essential for ensuring food security, supporting agricultural planning, and informing policy decisions in Malaysia. This study explores the application of machine learning techniques to predict monthly paddy production by integrating historical agricultural data with climate variables such as precipitation, temperature, and solar radiation. Three regression-based models—Random Forest Regressor, Support Vector Regression (SVR), and Long Short-Term Memory (LSTM)—were developed, trained, and evaluated to determine the most effective algorithm for capturing temporal dependencies and achieving high forecasting accuracy.

The research methodology involved extensive data preprocessing, including data cleaning, feature engineering, and temporal disaggregation to align annual and seasonal production data with monthly climate records. Input features included lagged production values, time-based variables, and climate parameters. The datasets were sourced from the Department of Statistics Malaysia (DOSM) and NASA's POWER project, ensuring a comprehensive representation of agricultural and climatic conditions across Malaysian states.

Model evaluation was conducted using standard metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2). Results indicated that LSTM outperformed Random Forest and SVR in terms of predictive accuracy, achieving the lowest RMSE and highest R^2 score. The Random Forest model demonstrated strong interpretability and decent performance, while SVR faced challenges related to scalability and sensitivity to input scaling.

This research contributes to both academic knowledge and practical applications by demonstrating the value of integrating agricultural and climatic data for forecasting purposes. It highlights the importance of localised forecasting models tailored to specific regions and underscores the potential of machine learning in enhancing agricultural decision-making. Future work includes expanding historical data coverage, refining data disaggregation methods, exploring additional features such as soil quality and socio-economic factors, and deploying real-time forecasting systems for broader accessibility and application beyond paddy crops.

ABSTRAK

Perkiraan pengeluaran beras (padi sawah) yang tepat adalah penting untuk memastikan keselamatan makanan, menyokong perancangan pertanian, dan memberi maklumat kepada keputusan dasar di Malaysia. Kajian ini mengeksplorasi penggunaan teknik pembelajaran mesin untuk meramal pengeluaran padi sawah bulanan dengan menggabungkan data pertanian sejarah bersama pemboleh ubah iklim seperti hujan, suhu, dan sinaran solar. Tiga model berdasarkan regresi—Random Forest Regressor, Support Vector Regression (SVR), dan Long Short-Term Memory (LSTM)—telah dibangunkan, dilatih, dan dinilai untuk menentukan algoritma yang paling berkesan dalam menangkap pergantungan masa dan mencapai ketepatan ramalan yang tinggi.

Metodologi kajian melibatkan pra-pemprosesan data yang luas, termasuk pembersihan data, kejuruteraan ciri, dan disagregasi masa bagi menyelaraskan data pengeluaran tahunan dan musim dengan rekod iklim bulanan. Ciri-ciri input merangkumi nilai pengeluaran tertunda, pemboleh ubah berdasarkan masa, dan parameter iklim. Set data diperoleh dari Jabatan Perangkaan Malaysia (DOSM) dan projek POWER NASA, memastikan perwakilan yang lengkap tentang keadaan pertanian dan iklim di seluruh negeri di Malaysia.

Penilaian model telah dijalankan menggunakan metrik piawaian seperti Mean Squared Error (MSE), Root Mean Squared Error (RMSE), dan R-squared (R^2). Keputusan menunjukkan bahawa LSTM mengatasi Random Forest dan SVR dari segi ketepatan ramalan, mencapai RMSE terendah dan skor R^2 tertinggi. Model Random Forest menunjukkan keboleh interpretaan yang kuat dan prestasi yang memadai, manakala SVR menghadapi cabaran berkaitan penskalaan dan kepekaan terhadap skala input.

Penyelidikan ini menyumbang kepada pengetahuan akademik dan aplikasi praktikal dengan menunjukkan nilai penggabungan data pertanian dan iklim untuk tujuan peramalan. Ia menegaskan kepentingan model ramalan tempatan yang

disesuaikan dengan kawasan tertentu serta menekankan potensi pembelajaran mesin dalam meningkatkan proses keputusan di bidang pertanian. Kerja-kerja akan datang merangkumi pengembangan liputan data sejarah, penyelarasan kaedah disagregasi data, pengkajian tambahan pada ciri-ciri seperti kualiti tanah dan faktor sosio-ekonomi, serta pelaksanaan sistem peramalan masa nyata bagi memperluaskan capaian dan aplikasi di luar tanaman padi sawah.

TABLE OF CONTENTS

TITLE PAGE

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

RF	-	Random Forest
SVR	-	Support Vector Regression
LSTM	-	Long Short-Term Memory
RMSE	-	Root Mean Squared Error
MSE	-	Mean Squared Error

LIST OF SYMBOLS

CHAPTER 1

INTRODUCTION

1.1 Problem Background

Rice is a vital part of Malaysia's food system. It's not just a staple on dinner tables across the country—it also plays a key role in national food security and supports the livelihoods of many rural communities. Although the government has made various efforts to boost local production, Malaysia still depends on rice imports to meet the needs of its population. According to the Department of Agriculture (DOA) Malaysia, rice makes a notable contribution to the agricultural sector's GDP, highlighting the importance of maintaining consistent and sustainable production.

But growing rice in Malaysia isn't without its challenges. Farmers are dealing with issues like unpredictable weather, changes in land use, limited water supply, and pest outbreaks. Climate change is making things even more uncertain, bringing shifts in rainfall, rising temperatures, and more extreme weather like floods and droughts. All of this directly affects how rice grows, how much can be harvested, and the overall stability of the farming cycle.

In response, there has been increasing interest in using data to better plan and manage agricultural activities. If we can predict rice yields more accurately, it would help farmers, planners, and policymakers stay ahead of potential shortages, manage resources wisely, and develop strategies to deal with climate-related risks. However, traditional forecasting methods often fall short—they usually depend on statistical models that might not fully capture the complex relationship between climate and crop performance.

That’s where machine learning (ML) comes in. As a branch of artificial intelligence, ML is well-suited for handling complex patterns and large amounts of data. By combining past climate data with machine learning algorithms, we can create models that offer more accurate and adaptable forecasts, tailored to local farming conditions. This study looks into how ML can be used to predict rice production in Malaysia, to support a more resilient, data-driven approach to agriculture.

1.2 Problem Background

Accurately predicting rice production is crucial, not just for maintaining food security but also for helping policymakers make informed decisions. Unfortunately, the forecasting methods currently used in Malaysia often fall short. They tend to overlook important climate details and rely on models that assume simple, linear relationships, which don’t fully reflect the complex and changing environmental factors that influence crop yields.

There’s still a shortage of research focused on using machine learning specifically to forecast rice production in Malaysia. Few studies have tapped into high-resolution historical climate data or explored how this information can be combined with advanced models to improve prediction accuracy. By integrating this kind of detailed data with machine learning, we could provide much more reliable forecasts, giving farmers, planners, and policymakers the insights they need to plan effectively.

This research aims to tackle several key challenges:

1. The limited use of climate data in current rice yield forecasting models.

2. The lack of localised machine learning models that reflect Malaysia's unique rice-growing conditions.

3. The a need for deeper analysis of how different climate factors impact rice yields over time.

1.3 Problem Statement

This study sets out to achieve the following goals:

a) Examine historical data on rice production and climate trends in Malaysia.

b) Identify which climate factors have the strongest impact on rice yields.

c) Build predictive models using selected machine learning techniques.

d) Assess and compare how well different machine learning models perform in forecasting rice production.

1.4 Research Gaps

To meet these objectives, the study aims to answer several key questions:

a) Which climate variables most significantly influence rice production in Malaysia?

b) Which machine learning model delivers the most accurate rice yield forecasts?

c) How well do these models perform when tested against real-world rice production data?

1.5 Significance of the Study

This research has practical importance for various groups in the agriculture sector:

a) Policymakers: Reliable forecasting can support the development of data-driven policies around food security, import strategies, and resource planning.

b) Agricultural Planners: Accurate predictions can help with scheduling planting, irrigation, and harvesting activities.

c) Farmers: Better forecasts allow for smarter decisions about managing crops, using inputs, and timing their sales.

d) Researchers: This work adds to the growing research on how machine learning can be used in agriculture, especially in tropical regions like Malaysia.

1.6 Scope and Limitations

This study focuses on predicting rice production in Malaysia using national-level historical data over a specific time frame. It includes key climate variables such as temperature, rainfall, humidity, solar radiation, and wind speed, sourced from official databases like the Department of Statistics Malaysia (DOSM) and NASA POWER.

However, there are a few limitations to note:

a) The accuracy of the models depends heavily on the availability and quality of historical data.

b) The models may not capture the full influence of socio-economic or farm-level practices on rice yields.

c) The findings are specific to rice production in Malaysia and may not directly apply to other crops or regions without adjustments.
Organisation of the Thesis

The structure of this thesis is as follows:

a) Chapter 1: Introduction – Outlines the background, research problem, objectives, questions, significance, scope, and the overall structure of the thesis.

b) Chapter 2: Literature Review – Explores existing research on rice farming in Malaysia, the impact of climate change on agriculture, current forecasting methods, and the use of machine learning in crop prediction.

c) Chapter 3: Methodology – Details the research approach, data sources, preprocessing steps, chosen machine learning techniques, and evaluation methods.

d) Chapter 4: Initial Findings and Analysis – Shares the model results, discusses the outcomes, and interprets the insights gained.

e) Chapter 5: Conclusion, Recommendations, and Future Work – Summarises the key contributions, outlines practical implications, notes the study's limitations, and suggests directions for future research.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Rice is a key food crop in Malaysia and plays an important role in national food security and rural livelihoods. However, rice production is affected by climate-related challenges such as unpredictable rainfall, rising temperatures, and extreme weather events. These issues make it increasingly important to develop accurate forecasting methods to support agricultural planning and decision-making.

Recent studies have shown that historical climate data, when combined with machine learning (ML) techniques, can improve the accuracy of crop yield predictions. Unlike traditional statistical models, ML methods can capture complex patterns in large datasets and are well-suited for modelling the relationship between climate variables and rice yields.

This literature review explores previous research on rice yield forecasting, with a focus on the Malaysian context. It begins by discussing trends in rice production and the effects of climate on yield. It then reviews different types of data used in forecasting, compares classical statistical models with modern ML approaches, and highlights common evaluation methods. The chapter also identifies research gaps and provides a foundation for the methodology used in this study.

2.2 Overview of Rice Production in Malaysia

Rice (Oryza sativa) is a staple food for the majority of Malaysians and an essential component of national food security. Despite its importance, Malaysia remains only partially self-sufficient in rice production, with a self-sufficiency level (SSL) that has fluctuated around 70%–75% in recent years (Ministry of Agriculture and Food Security, 2023). The government continues to implement policies and programs to boost local production and reduce reliance on imports.

Malaysia’s rice production is geographically concentrated in key granary areas such as the Muda Agricultural Development Authority (MADA) region in Kedah, the Kemubu Agricultural Development Authority (KADA) in Kelantan, and the Integrated Agricultural Development Areas (IADAs) in Sabah and Sarawak. These areas benefit from irrigation infrastructure, making double-cropping systems possible and contributing significantly to national output.

National production levels are influenced by a range of agroecological and socioeconomic factors, including land availability, fertiliser use, irrigation access, and especially climatic conditions such as rainfall patterns and temperature variation. Paddy yield per hectare has generally improved due to modernisation and government subsidies, but it remains vulnerable to climate variability and extreme weather events (Shamshiri et al., 2018).

To address production challenges, the Malaysian government has introduced several strategic plans, including the National Agro-Food Policy 2.0 (2021-2030), which aims to modernise agriculture, promote sustainable practices, and increase the rice SSL to 80% by 2030 (Ministry of Agriculture and Food Industries, 2024). Investments in smart farming technologies, early warning systems, and precision agriculture are among the key strategies being promoted.

2.3 Climate Variables Affecting Rice Production

Rice production is highly sensitive to climatic conditions, particularly in tropical regions like Malaysia, where paddy cultivation often depends on seasonal rainfall and temperature regimes. Key climatic variables such as rainfall, temperature, humidity, solar radiation, and wind speed significantly influence rice growth stage, including germination, tillering, flowering, and grain filling (Sparks, 2009). Deviations from optimal climate conditions, such as droughts, floods, or extreme heat, can lead to substantial yield losses.

In Malaysia, studies have shown that rainfall variability plays a critical role in determining rice yield, especially in rain-fed areas without reliable irrigation infrastructure (Gumel et al., 2017). The onset and distribution of the monsoon season, both the Southwest and Northeast monsoons, directly affect planting schedules and water availability during critical growth stages. Delayed rainfall or prolonged dry spells have been associated with reduced yields, particularly in states like Kelantan, Sabah, and Sarawak (Tan et al., 2021).

Temperature is another crucial factor, as rice is sensitive to both low and high-temperature extremes. High temperatures above 35°C during flowering stages can cause spikelet sterility and grain abortion, significantly reducing yields (Yoshida, 1981). Conversely, minimum temperatures below 20°C can also suppress germination and tillering. Recent trends in Malaysia indicate rising average temperatures and more frequent heatwaves, posing a long-term risk to stable paddy production (Shamshiri et al., 2018).

Humidity and solar radiation also impact photosynthetic efficiency and transpiration rates. While high relative humidity is typical in Malaysia and generally supports growth, excessive humidity may increase pest and disease risks, such as rice blast or sheath blight. Solar radiation influences biomass accumulation and grain filling; reduced sunshine during prolonged rainy seasons has been correlated with yield declines in the Muda and KADA granaries (Herath et al., 2020).

Given these sensitivities, forecasting rice yield in Malaysia requires a strong understanding of how these climatic determinants interact with local agricultural practices and rice varieties. The use of historical climate data, particularly rainfall and temperature trends, is essential for developing accurate and site-specific prediction models.

2.4 Role of Data Analytics in Agriculture

The rise of digital agriculture has opened up new possibilities for data-driven decision-making. With increasing access to real-time data from satellites, weather stations, and agricultural surveys, analytics is becoming a vital tool in farming. Predictive analytics, in particular, helps anticipate future outcomes based on patterns found in historical and real-time data.

In recent years, machine learning (ML) has gained momentum in agriculture due to its ability to handle complex, non-linear relationships—something traditional statistical models often struggle with. ML algorithms can analyse large, multi-variable datasets to identify patterns and make accurate forecasts that inform better decisions on the ground.

By combining historical climate data with crop yield records, ML enables the development of robust forecasting tools that support smarter planning at both the policy and farm level. This shift aligns with global trends toward precision agriculture and smart farming, where technology helps maximise productivity and sustainability.

2.5 Classical Statistical Forecasting Approaches

Before the rise of machine learning, classical statistical models were widely used for crop yield forecasting due to their interpretability and relatively low computational requirements. These models typically assume linear relationships and rely heavily on time series data or explanatory variables such as temperature, rainfall, or cultivated area.

One of the common approaches is Multiple Linear Regression (MLR), which models the relationship between a dependent variable (such as rice yield) and multiple independent variables (e.g., rainfall, temperature, and fertiliser use). While easy to implement and interpret, MLR is often limited by its assumption of linearity and sensitivity to multicollinearity and outliers (Oguntunde et al., 2018).

Other classical approaches include Generalised Linear Models (GLM) and panel regression models, which allow for the inclusion of fixed and random effects when analysing data from multiple locations or years. These methods have been used in multi-location studies to estimate the effects of climate variables on rice production across different agroecological zones (Joshi et al., 2011).

Although these statistical methods are still relevant for baseline analysis and comparisons, they often struggle to capture non-linear and complex interactions between variables. This has led to a growing interest in machine learning models that are more flexible and better suited for large and noisy datasets.

2.6 Applications of Machine Learning in Malaysian Agriculture

Although machine learning has been widely applied to agriculture around the world, its use in Malaysia remains limited. Most local studies have relied on basic statistical models or general trends, often using coarse satellite data instead of detailed, state-level datasets.

Only a handful of studies have attempted to forecast rice production in Malaysia using ML, and those that do often use simple linear regressions without incorporating detailed climate data. There is also minimal exploration of advanced ML algorithms like Random Forest, SVR, or LSTM in a localised text. Furthermore, few Malaysian studies utiutilisegh-quality data sources like NASA's POWER dataset, which provides detailed and validated climate data tailored for agriculture.

2.7 Gaps in Existing Research

Despite global progress in ML applications for agriculture, several key research gaps remain in the Malaysian context:

a) Limited Use of ML in Local Forecasting: Many existing studies still rely on traditional methods, overlooking the potential of advanced machine learning techniques.

b) Weak Integration of Climate Data: Most models use a narrow range of climate variables or outdated sources, which limits forecasting accuracy.

c) Lack of Model Comparisons: Few studies compare the performance of multiple ML models to determine which works best under Malaysian conditions.

d) Underuse of Time-Series Methods: Techniques like LSTM, which are capable of modmodellingasonal trends and long-term dependencies, are rarely explored in local rice forecasting efforts.

2.8 Addressing the Gaps

This research aims to fill these gaps by:

a) Applying machine learning models—Random Forest, SVR, and LSTM—to forecast rice production using local Malaysian data.

b) Incorporating a wide range of climate variables from high-resolution datasets like NASA POWER.

c) Including lagged features and time-based indicators to account for seasonality and temporal trends.

d) Comparing the performance of different models to identify the most accurate and reliable approach for Malaysian rice forecasting.

By addressing these issues, the study hopes to contribute meaningful insights to the growing field of smart agriculture in Malaysia and support more informed, climate-resilient decision-making.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Research Design

This study adopts a quantitative research design with a focus on predictive modeling. The central objective is to forecast Malaysia's rice (paddy) production using historical climate data and machine learning techniques. By implementing and comparing three different regression-based models—Random Forest, Support Vector Regression (SVR), and Long Short-Term Memory (LSTM)—the research aims to determine the most effective algorithm for accurate yield forecasting under varying climatic conditions.

The methodological framework includes several key stages:

a) Data collection and preprocessing

b) Exploratory Data Analysis (EDA)

c) Feature engineering

d) Model development and training

e) Model evaluation and comparison

Python was the primary programming language used throughout the study, supported by various libraries such as pandas, numpy, scikit-learn, matplotlib, seaborn, and TensorFlow/Keras.

3.2 Data Sources

3.2.1 Crop Production Dataset

Historical paddy production data was sourced from the Department of Statistics Malaysia (DOSM). This dataset includes monthly production statistics across multiple states from 2017 to 2022. The key attributes are:

a) state: Name of the state

b) date: Observation date in year-month format

c) crop_type: Crop category (paddy)

d) planted_area: Total cultivated area in hectares

e) production: Total rice output in metric tons

3.2.2 Climate Dataset

Climate-related variables were retrieved from NASA's POWER (Prediction Of Worldwide Energy Resource) project. These variables were extracted monthly for each state and merged with crop data. The primary climate variables used include:

a) PRECTOTCORR_SUM: Total monthly precipitation (mm)

b) T2M_MAX: Monthly maximum temperature (deg C)

c) T2M_MIN: Monthly minimum temperature (deg C)

d) RH2M: Relative humidity at 2 meters (%)

e) ALLSKY_SFC_LW_DWN: Surface longwave radiation (W/m^2)

3.3 Data Description

The final merged dataset consisted of 792 rows and 20 columns, with each row representing a monthly observation per state. Important variables included:

a) state, date, planted_area, production

b) Climate variables as mentioned above

c) yield: Calculated as production / planted_area

d) Time-based variables: month, year, week_of_year

e) Lagged variables: e.g., production_lag_1, yield_lag_1, production_lag_2, etc.

3.4 Data Preprocessing

Data preprocessing is a critical step in any machine learning project as it ensures that the data is clean, consistent, and ready for model training. In this study, several preprocessing steps were applied to both the crop production dataset obtained from the Department of Statistics Malaysia (DOSM) and the climate dataset sourced from NASA's POWER project.

3.4.1 Data Cleaning

The initial step involved cleaning the datasets to remove inconsistencies and ensure uniformity:

a) The column name 'NAME' was renamed to 'state' for clarity.

b) All state names were standardized to lowercase to avoid case sensitivity issues.

c) The 'date' column was converted into a proper datetime format to facilitate time-based operations such as sorting and feature extraction.

d) Missing values were checked using .isnull().sum() and no missing values were found in either dataset.

e) Duplicate rows were identified using .duplicated().sum() and none were present.

3.4.2 Dummy Data Generation

Because the crop data was originally provided as annual totals, a monthly dummy allocation was implemented based on a seasonal distribution pattern, enabling time-aligned comparison with monthly climate data.

3.4.3 Merging Datasets

To integrate climate variables into the analysis, the monthly paddy dataset was merged with the processed climate dataset using the keys 'state' and 'date'. An inner join was performed to retain only those records where both paddy and climate data were available. This resulted in a final merged dataset containing 792 rows and 20 columns. An inner join was used to merge climate and production datasets on state and date, ensuring only matched records were retained.

3.4.4 Feature Engineering

Several new features were derived to improve model performance:

- Target Variable ('yield') : Calculated as the ratio of production to planted area (production / planted_area). To handle potential division by zero, infinite values were replaced with zero.

- Lagged Features : Lagged versions of production and yield were created to capture temporal dependencies:
  - production_lag_1, production_lag_2, production_lag_3
  - yield_lag_1, yield_lag_2, yield_lag_3
  - These lagged features were generated using groupby('state') to maintain consistency within each state's time series.

- Time-Based Features : Additional time-related features were extracted from the 'date' column:
  - month: Extracted as an integer (1–12)
  - year: Extracted as an integer (e.g., 2017, 2018)
  - week_of_year: Derived using .dt.isocalendar().week

- These engineered features enhanced the models' ability to capture seasonality and trends over time.

3.4.5 Train-Test Split

To preserve the temporal order of observations in the time series, a chronological train-test split was applied:

a) Training Set : 80% of the data (607 samples), covering earlier time periods

b) Testing Set : 20% of the data (152 samples), consisting of the most recent observations

This ensured that the model was evaluated on its ability to predict future values rather than past or random ones.

3.4.6 Feature Scaling

Since Support Vector Regression (SVR) and Long Short-Term Memory (LSTM) are sensitive to the scale of input features, all numerical features were standardized using StandardScaler:

a) Each feature was transformed to have zero mean and unit variance.

b) The target variable (production) was also scaled before model training and later inverse-transformed after prediction to return to the original units.

All preprocessing steps were carried out using Python libraries such as pandas, numpy, and scikit-learn.

3.5 Machine Learning Models

This section outlines the three machine learning models used in this study to forecast Malaysian rice (paddy) production using historical climate data. These models include Random Forest Regressor , Support Vector Regression (SVR) , and Long Short-Term Memory (LSTM) neural networks. Each model was selected based on its suitability for regression tasks and its ability to capture complex patterns in time-series data.

3.5.1 Random Forest Regressor

The Random Forest Regressor is an ensemble learning method that combines multiple decision trees to improve prediction accuracy and reduce overfitting. It works by training each tree on a random subset of the data and features, then averaging the predictions across all trees.

a) Advantages:

- Handles non-linearities
- Resistant to overfitting
- Provides feature importance metrics

b) Hyperparameters:

- n_estimators = 100
- random_state = 42
- n_jobs = -1 (parallel processing)

The model was trained on scaled input features and evaluated using standard regression metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2).

3.5.2 Support Vector Regression (SVR)

Support Vector Regression (SVR) is a kernel-based algorithm that performs regression by finding a hyperplane that best fits the data within a certain margin of tolerance (epsilon). It is particularly effective in high-dimensional spaces and can handle small datasets with good generalization performance.

a) Advantages:

- Performs well with high-dimensional and small datasets

b) Hyperparameters:

- Kernel: rbf
- C = 100, gamma = 0.1, epsilon = 0.1

Before training, both input features and target variables were scaled using StandardScaler due to SVR's sensitivity to feature scales.

3.5.3 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) designed to learn long-term dependencies in sequential data. It is widely used for time-series forecasting because it can capture temporal patterns and trends effectively.

a) Architecture:

- One LSTM layer with 50 units
- Dropout layer (rate = 0.2)
- Dense output layer

b) Training Parameters:

- Optimizer: Adam
- Loss function: MSE
- Early stopping with patience = 10
- Batch size: 32, Epochs: 100

Input data was reshaped into a 3D format [samples, timesteps, features] required by LSTM. The target variable was also scaled before training and inverse-transformed after prediction for interpretation.

3.6 Model Development Process

Step 1: Define Input Features and Target Variable

The first step involved selecting the most relevant input features and defining the target variable for prediction.

a) Input Features (X):

- planted_area

- PRECTOTCORR_SUM (Total monthly precipitation)
- T2M_MAX (Maximum temperature)

b) Lagged features:

- production_lag_1, production_lag_2, production_lag_3
- yield_lag_1, yield_lag_2, yield_lag_3

c) Time-based features:

- month, year, week_of_year

d) Target Variable (y):

- production: Total paddy production in metric tons

These features were selected based on their relevance to crop yield and their ability to capture temporal patterns through lagging and time-based engineering.

Step 2: Train-Test Data Splitting

To ensure that the model was evaluated on its ability to predict future values rather than past or random ones, a chronological train-test split was applied:

a) Training Set: 80% of the data (607 samples), covering earlier time periods

b) Testing Set: 20% of the data (152 samples), consisting of the most recent observations

This split preserved the order of time-series data and ensured realistic evaluation of forecasting performance.

Step 3: Feature Scaling

Since Support Vector Regression (SVR) and Long Short-Term Memory (LSTM) are sensitive to feature scales, all numerical features were standardized using StandardScaler from scikit-learn.

a) Each feature was transformed to have zero mean and unit variance.

b) The target variable (production) was also scaled before model training and later inverse-transformed after prediction to return to the original units.

Step 4: Model Training

Random Forest Regressor , Support Vector Regression (SVR) , and Long Short-Term Memory (LSTM) were trained using the preprocessed dataset. The dataset was split into training and testing sets, with 80% of the data used for training and the remaining 20% reserved for evaluation. To ensure consistency across models sensitive to feature scales, such as SVR and LSTM, all numerical input features were standardized using StandardScaler, transforming them to have zero mean and unit variance. The target variable (production) was also scaled before training and later inverse-transformed after prediction for interpretability.

For the Random Forest Regressor , a tree-based ensemble method, the model was initialized with 100 decision trees (n_estimators=100), and trained on the scaled training data. Random Forest does not require strict feature scaling, but it was applied for consistency across models. The model learned patterns from the input features, including lagged production values, time-based features, and climate variables, to predict paddy production.

The Support Vector Regression (SVR) model was trained using a Radial Basis Function (RBF) kernel, which is effective in capturing non-linear relationships in high-dimensional spaces. Hyperparameters such as regularization (C=100), kernel coefficient (gamma=0.1), and tube size (epsilon=0.1) were set based on prior experimentation. Due to SVR's sensitivity to input scale, both the input features and target variable were scaled before training.

Finally, the LSTM model, a type of recurrent neural network suitable for sequence prediction tasks, was built using TensorFlow/Keras. Input data was reshaped into a 3D format [samples, timesteps, features] required by LSTM. The architecture included one LSTM layer with 50 units, followed by a dropout layer to prevent overfitting, and a dense output layer. The model was compiled using the Adam optimizer and Mean Squared Error (MSE) loss function. Early stopping was implemented during training to halt the process if validation loss did not improve for 10 consecutive epochs, preventing overfitting and saving computation time.

Step 5: Prediction and Evaluation

After training, each model made predictions on the test dataset. Predicted values were inverse-scaled to their original units before evaluation. The evaluation metrics used: Mean Squared Error (MSE), Root Mean Squared Error (RMSE) and R-squared (R^2 Score)

The model development process included defining input-output variables, splitting data chronologically, applying feature scaling, training each model, and evaluating their performance using standard metrics.

3.7 Evaluation Metrics

To evaluate the accuracy and effectiveness of the developed machine learning models in forecasting paddy production, three widely accepted regression evaluation metrics were used.

3.7.1 Mean Squared Error (MSE)

The Mean Squared Error (MSE) measures the average squared difference between the actual values and the predicted values. It emphasizes larger errors due to the squaring operation, making it sensitive to outliers.

$$\mathrm{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

3.7.2 Root Mean Squared Error (RMSE)

The Root Mean Squared Error (RMSE) is the square root of the MSE. It provides an error metric in the same unit as the target variable (paddy production), making it more interpretable than MSE.

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

3.7.3 R-squared (R²)

The R-squared (R²) score measures how well the model explains the variability of the target variable. It ranges from 0 to 1, where:

- 1 indicates a perfect fit
- 0 indicates that the model performs no better than predicting the mean of the target

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

3.8 Tools and Technologies

On the Insert tab, the galleries include items that are designed to coordinate with the ove

Component	Description
Programming Language	Python 3.11
Libraries	pandas, numpy, scikit-learn, matplotlib, seaborn, TensorFlow/Keras
Platform	Google Colab Pro (with GPU support)

3.9 Summary

This chapter provided a detailed overview of the research methodology employed to forecast rice production in Malaysia. It described the data sources, preprocessing techniques, feature engineering methods, and the architecture of the three machine learning models. The evaluation metrics and tools used were also discussed. The next chapter presents the results of the model training process and provides a comparative performance analysis across the three algorithms.

CHAPTER 4

INITIAL FINDINGS

4.1 Introduction

Accurate forecasting of paddy (rice) production is essential for ensuring food security, supporting agricultural planning, and informing policy decisions in Malaysia. As one of the key staple crops, paddy production is influenced by a variety of factors including climate conditions, land use, agricultural practices, and government policies. This chapter presents the initial findings from the data collection, preprocessing, exploratory data analysis, and preliminary modeling efforts aimed at developing a reliable forecasting framework. The chapter begins with an overview of the datasets used, followed by detailed descriptions of the data cleaning and processing steps. Exploratory data analysis provides insights into historical trends, seasonal patterns, and relationships between production and climatic variables. Finally, early forecasting models—namely Random Forest, Support Vector Regression (SVR), and Long Short-Term Memory (LSTM)—are introduced, along with their performance evaluation. These findings serve as the foundation for further model refinement and more in-depth analysis presented in the subsequent chapters.

4.2 Data Loading and Preparation

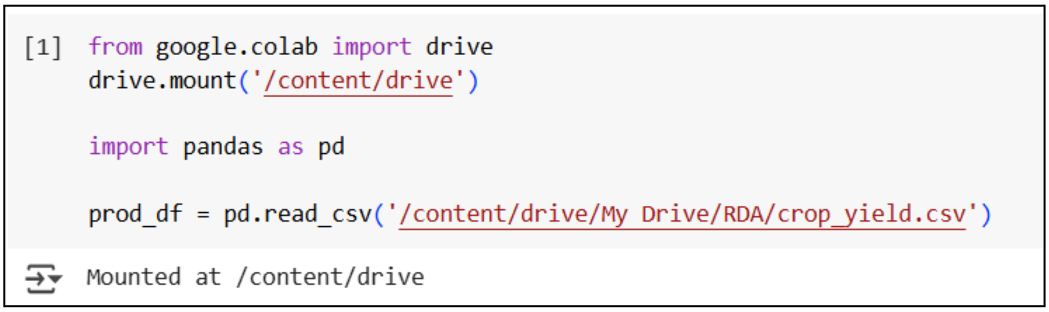
This section outlines the initial steps taken to load, clean, and prepare the datasets for analysis. Two main datasets were used: one containing historical crop yield data and another with state-specific monthly weather information. Key preprocessing tasks included renaming and standardising columns (e.g., 'NAME' to 'state'), converting date fields to datetime format, and checking for missing values and duplicates. These steps ensured the data was structured consistently and ready for further processing and exploratory analysis.

4.2.1 Dataset Sources and Structure

The forecasting of paddy production in Malaysia was based on two primary datasets that provide essential agricultural and climatic information:

a) Crop Yield Dataset (crop_yield.csv)

This dataset contains historical records of crop production across various states in Malaysia. It includes data at both annual and seasonal levels, covering multiple crops, with a focus on key agricultural indicators such as production quantity (in tonnes), planted area (in hectares), and yield (tonnes per hectare). For this study, only the records related to "paddy" were extracted for further analysis.



```
[1] from google.colab import drive
    drive.mount('/content/drive')

    import pandas as pd

    prod_df = pd.read_csv('/content/drive/My Drive/RDA/crop_yield.csv')
```

```
Mounted at /content/drive
```

Figure 4.1 Figure of Data Loading of crop_yield.csv

b) Weather Data Files

These consist of multiple CSV files, each corresponding to a specific Malaysian state. The files contain detailed monthly weather parameters obtained from NASA's MERRA-2 or similar climate datasets. Key variables include total precipitation (PRECTOTCORR_SUM), minimum, and maximum temperatures (T2M_MIN, T2M_MAX), and solar radiation. These variables are critical for understanding how climatic conditions influence paddy production over time.

These two datasets formed the foundation for building a comprehensive model to forecast future paddy production by integrating both agricultural and environmental factors.

4.3 Paddy Data Processing

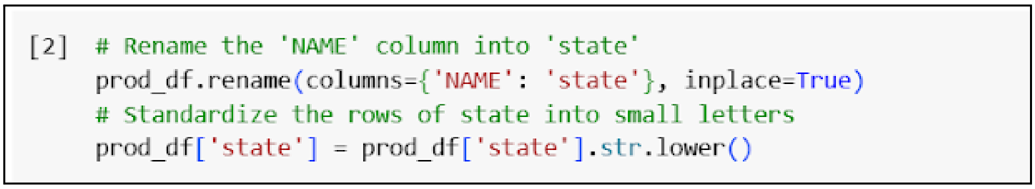
This section focuses on preparing the crop yield data specifically for paddy (rice) production. Since the original dataset included multiple crops, it was filtered to include only paddy-related records. To align with monthly weather data, annual and seasonal paddy production data was disaggregated into monthly values using a predefined distribution pattern, resulting in the monthly_paddy_df. States with zero total production were assigned zero values for each month to maintain consistency. This step ensured the data was temporally aligned and ready for integration with climate variables.

The following steps were carried out on the crop yield dataset:

a) Renaming and Standardizing the 'NAME' Column:

The column originally labeled 'NAME' in the crop yield dataset contained the names of Malaysian states. To improve clarity and ensure consistency with other datasets (especially when merging with weather data later), this column was renamed to 'state'. Additionally, all state names were converted to lowercase.

This standardization helped prevent mismatches due to case sensitivity (e.g., "Selangor" vs. "selangor") during data integration.

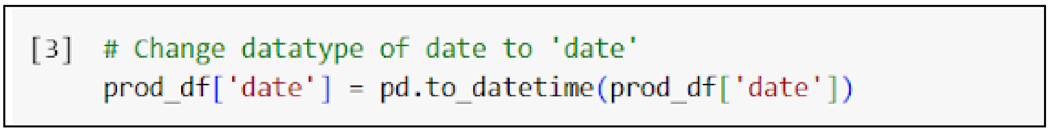


```
[2]  # Rename the 'NAME' column into 'state'
     prod_df.rename(columns={'NAME': 'state'}, inplace=True)
     # Standardize the rows of state into small letters
     prod_df['state'] = prod_df['state'].str.lower()
```

Figure 4.2 Figure of Renaming and Standardizing of crop_yield.csv

b) Conversion of Date Columns to Datetime Format:

Any column containing date information (such as the planting or harvesting date) was converted from string format into Python’s datetime object. This transformation is crucial for time-series analysis, as it enables sorting by date, extracting time-based features (like month or year), and aligning data across different time intervals. It also facilitates plotting and analysis over time.

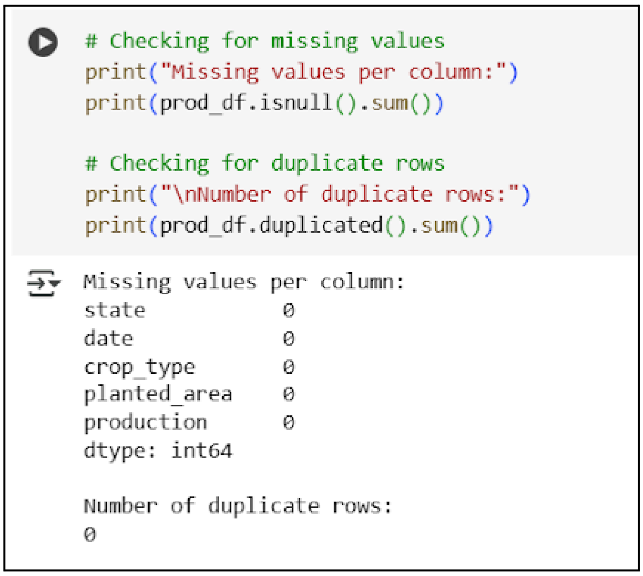


```
[3]  # Change datatype of date to 'date'
     prod_df['date'] = pd.to_datetime(prod_df['date'])
```

Figure 4.3 Figure of Conversion of Date Columns of crop_yield.csv

c) Checking for Missing Values and Duplicates:

An initial inspection of the raw crop yield data revealed that there were no missing values or duplicate rows present in the dataset at the start of the preprocessing stage. This is important because missing or duplicate data can introduce bias or errors in analysis and modeling. While the crop data itself was clean at this point, as will be discussed later, missing values did appear after merging with the weather data, requiring further handling.



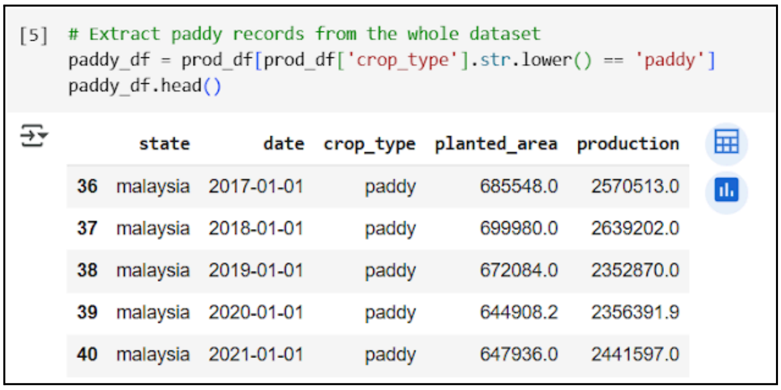
```
# Checking for missing values
print("Missing values per column:")
print(prod_df.isnull().sum())

# Checking for duplicate rows
print("\nNumber of duplicate rows:")
print(prod_df.duplicated().sum())
```

```
Missing values per column:
state           0
date            0
crop_type       0
planted_area    0
production      0
dtype: int64

Number of duplicate rows:
0
```

Figure 4.4 Figure of Checking for Missing and Duplicates of crop_yield.csv

4.3.1 Filtering for Paddy Crop

To ensure the analysis was focused specifically on rice production in Malaysia, the dataset was filtered to include only records related to "paddy" , which is the term commonly used for rice in its unprocessed form (before milling). The original crop yield dataset contained data for multiple crops (e.g., corn, sugarcane), so isolating paddy data allowed for more accurate and relevant modeling. This step ensured that all subsequent analyses, including feature engineering and forecasting, were tailored to the target crop.



```
[5] # Extract paddy records from the whole dataset
    paddy_df = prod_df[prod_df['crop_type'].str.lower() == 'paddy']
    paddy_df.head()
```

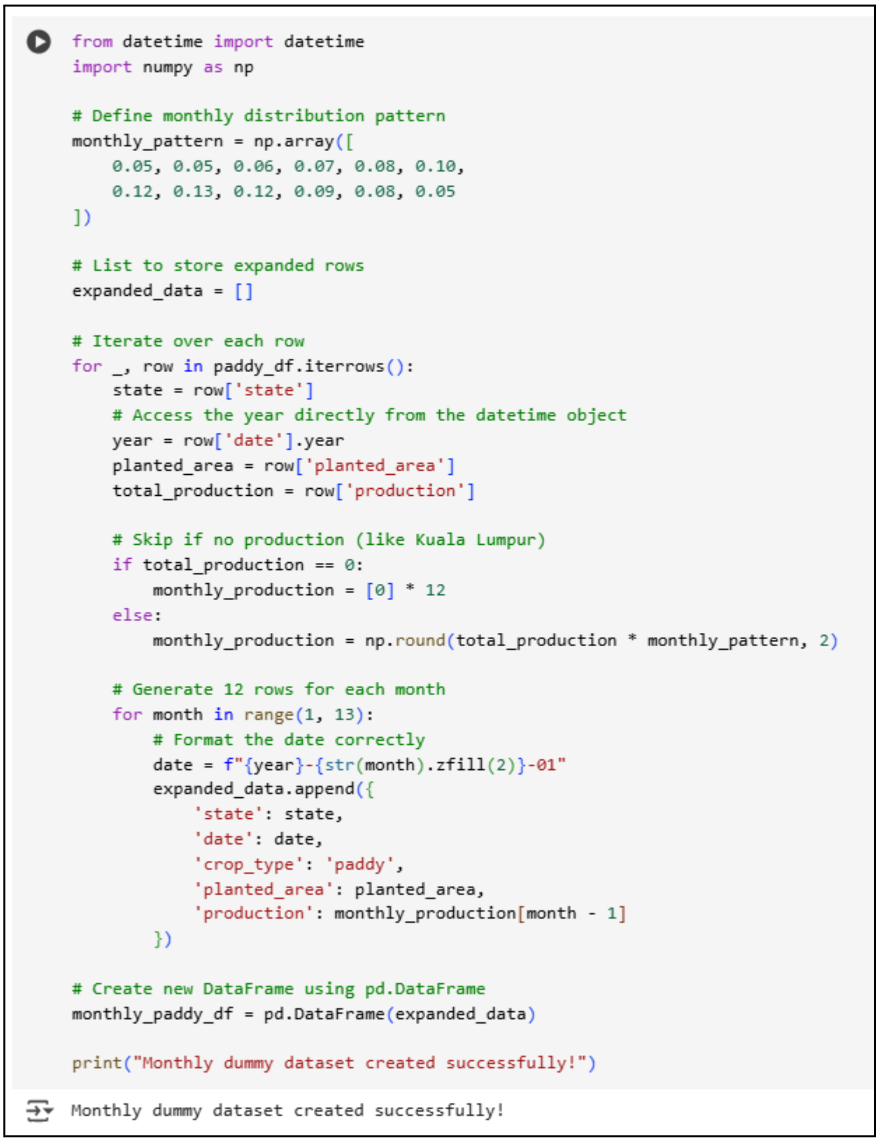
	state	date	crop_type	planted_area	production
36	malaysia	2017-01-01	paddy	685548.0	2570513.0
37	malaysia	2018-01-01	paddy	699980.0	2639202.0
38	malaysia	2019-01-01	paddy	672084.0	2352870.0
39	malaysia	2020-01-01	paddy	644908.2	2356391.9
40	malaysia	2021-01-01	paddy	647936.0	2441597.0

Figure 4.5 Figure of Filtering for Paddy Crop

4.3.2 Temporal Disaggregation

The original paddy production data was recorded on an annual or seasonal basis , meaning it provided total production values per year or per planting season. However, since weather data was available at a monthly resolution , it was necessary to convert the annual/seasonal data into monthly intervals to allow proper alignment and integration.

To achieve this, a predefined monthly distribution pattern was applied. This pattern distributed the annual or seasonal production across months based on typical planting, growing, and harvesting cycles. As a result, the new monthly_paddy_df dataset was created, containing estimated monthly production values. This granular time-based format enabled more precise modeling and correlation with monthly weather variables.



```
from datetime import datetime
import numpy as np

# Define monthly distribution pattern
monthly_pattern = np.array([
    0.05, 0.05, 0.06, 0.07, 0.08, 0.10,
    0.12, 0.13, 0.12, 0.09, 0.08, 0.05
])

# List to store expanded rows
expanded_data = []

# Iterate over each row
for _, row in paddy_df.iterrows():
    state = row['state']
    # Access the year directly from the datetime object
    year = row['date'].year
    planted_area = row['planted_area']
    total_production = row['production']

    # Skip if no production (like Kuala Lumpur)
    if total_production == 0:
        monthly_production = [0] * 12
    else:
        monthly_production = np.round(total_production * monthly_pattern, 2)

    # Generate 12 rows for each month
    for month in range(1, 13):
        # Format the date correctly
        date = f"{year}-{str(month).zfill(2)}-01"
        expanded_data.append({
            'state': state,
            'date': date,
            'crop_type': 'paddy',
            'planted_area': planted_area,
            'production': monthly_production[month - 1]
        })

# Create new DataFrame using pd.DataFrame
monthly_paddy_df = pd.DataFrame(expanded_data)

print("Monthly dummy dataset created successfully!")
```

```
Monthly dummy dataset created successfully!
```

Figure 4.6 Figure of Creating Monthly Dataset

4.3.3 Handling Zero Production States

Some Malaysian states had zero total paddy production during certain years or seasons. While this may reflect actual agricultural conditions (e.g., no paddy cultivation in a particular area), omitting these entries could lead to data imbalance or bias in the modeling process.

To maintain completeness and ensure accurate representation, these zero-production states were retained in the dataset. Specifically, each month within a zero-production year or season was assigned a value of zero for paddy production. This approach preserved the integrity of the dataset and allowed models to learn from complete spatial and temporal patterns without assuming production where there was none.

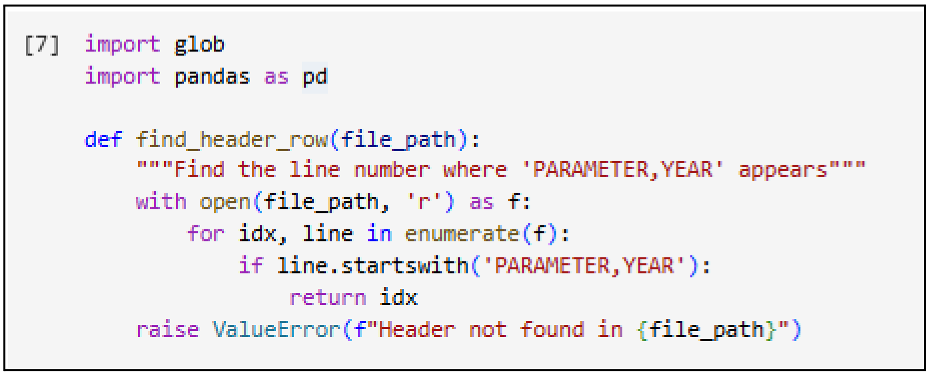
4.4 Weather Data Processing

This section describes how weather data from multiple sources was cleaned and transformed for use in the forecasting model. The data came in separate CSV files for each Malaysian state, with complex formats and inconsistent headers. A custom function was used to locate the correct header row. The data was then reshaped from wide to long format and restructured to have weather variables as columns. A 'state' column was added based on the file name, and all state datasets were combined into one unified DataFrame (weather_df_all). This processed dataset was then ready to be merged with the paddy production data.

4.4.1 File Reading and Header Detection

The weather data for each Malaysian state was stored in separate CSV files. However, not all files started directly with the header row containing column names like 'YEAR', 'MO', or weather parameters. Some files included additional metadata lines at the beginning—such as descriptions of the data source or units—which could interfere with proper data loading.

To handle this inconsistency, a custom function called find_header_row() was created. This function scanned through each file to automatically detect the first line that contained actual column headers , skipping any initial metadata lines. Using this function ensured that the correct data structure was read from each file, regardless of how many metadata lines it contained.



```
[7] import glob
    import pandas as pd

    def find_header_row(file_path):
        """Find the line number where 'PARAMETER,YEAR' appears"""
        with open(file_path, 'r') as f:
            for idx, line in enumerate(f):
                if line.startswith('PARAMETER,YEAR'):
                    return idx
        raise ValueError(f"Header not found in {file_path}")
```

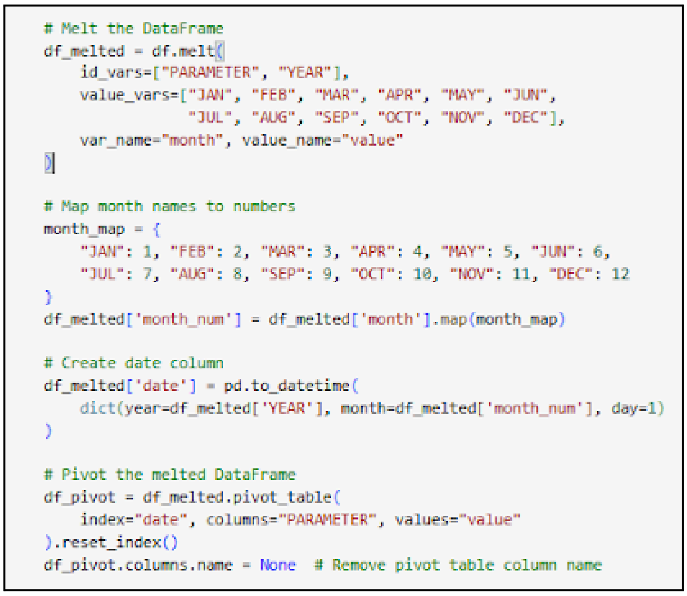
Figure 4.7 Figure of File Reading and Header Detection

4.4.2 Format Transformation

Once the correct headers were identified and the data was loaded, the next step was to restructure the format of the weather data.

Originally, the data was in wide format , where each month (e.g., Jan, Feb) was represented as a separate column. To make the data more suitable for time-series analysis and easier to merge with the paddy production dataset, it was transformed into long format using the melt() function. This process converted the monthly columns into rows, with each row representing a specific month and its corresponding weather value.

After reshaping, the data was then pivoted again so that each weather parameter (e.g., 'PRECTOTCORR_SUM' for precipitation, 'T2M_MAX' for maximum temperature) became a separate column, and the 'date' field was set as the index. This final structure made the dataset clean, consistent, and ready for further analysis.



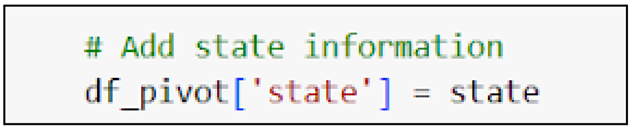
```python
# Melt the DataFrame
df_melted = df.melt(
    id_vars=["PARAMETER", "YEAR"],
    value_vars=["JAN", "FEB", "MAR", "APR", "MAY", "JUN",
                "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"],
    var_name="month", value_name="value"
)

# Map month names to numbers
month_map = {
    "JAN": 1, "FEB": 2, "MAR": 3, "APR": 4, "MAY": 5, "JUN": 6,
    "JUL": 7, "AUG": 8, "SEP": 9, "OCT": 10, "NOV": 11, "DEC": 12
}
df_melted['month_num'] = df_melted['month'].map(month_map)

# Create date column
df_melted['date'] = pd.to_datetime(
    dict(year=df_melted['YEAR'], month=df_melted['month_num'], day=1)
)

# Pivot the melted DataFrame
df_pivot = df_melted.pivot_table(
    index="date", columns="PARAMETER", values="value"
).reset_index()
df_pivot.columns.name = None  # Remove pivot table column name
```

Figure 4.8 Figure of Format Transformation

4.4.3 State Identification and Concatenation

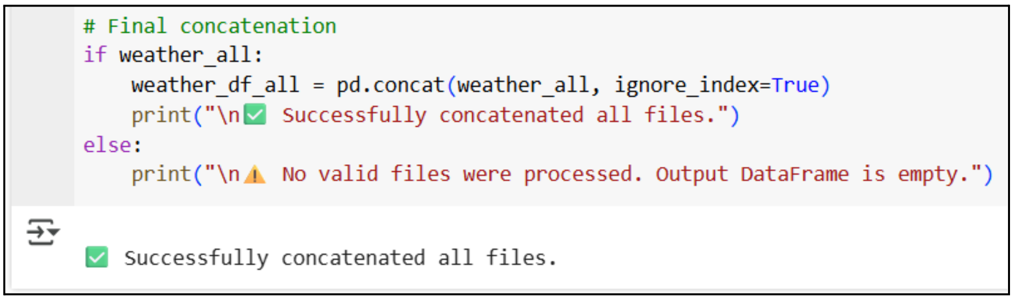
Since each CSV file corresponded to a different Malaysian state, it was important to identify which state the data belonged to. A new column called 'state' was added to each DataFrame during processing, derived directly from the filename (e.g., Johor.csv, Selangor.csv).

Once each state’s weather data was cleaned and structured consistently, all individual DataFrames were combined into a single unified DataFrame named weather_df_all . This consolidated dataset contained weather information for all states, with standardized formatting and a 'state' identifier, making it ready for merging with the paddy production data.



```
# Add state information
df_pivot['state'] = state
```

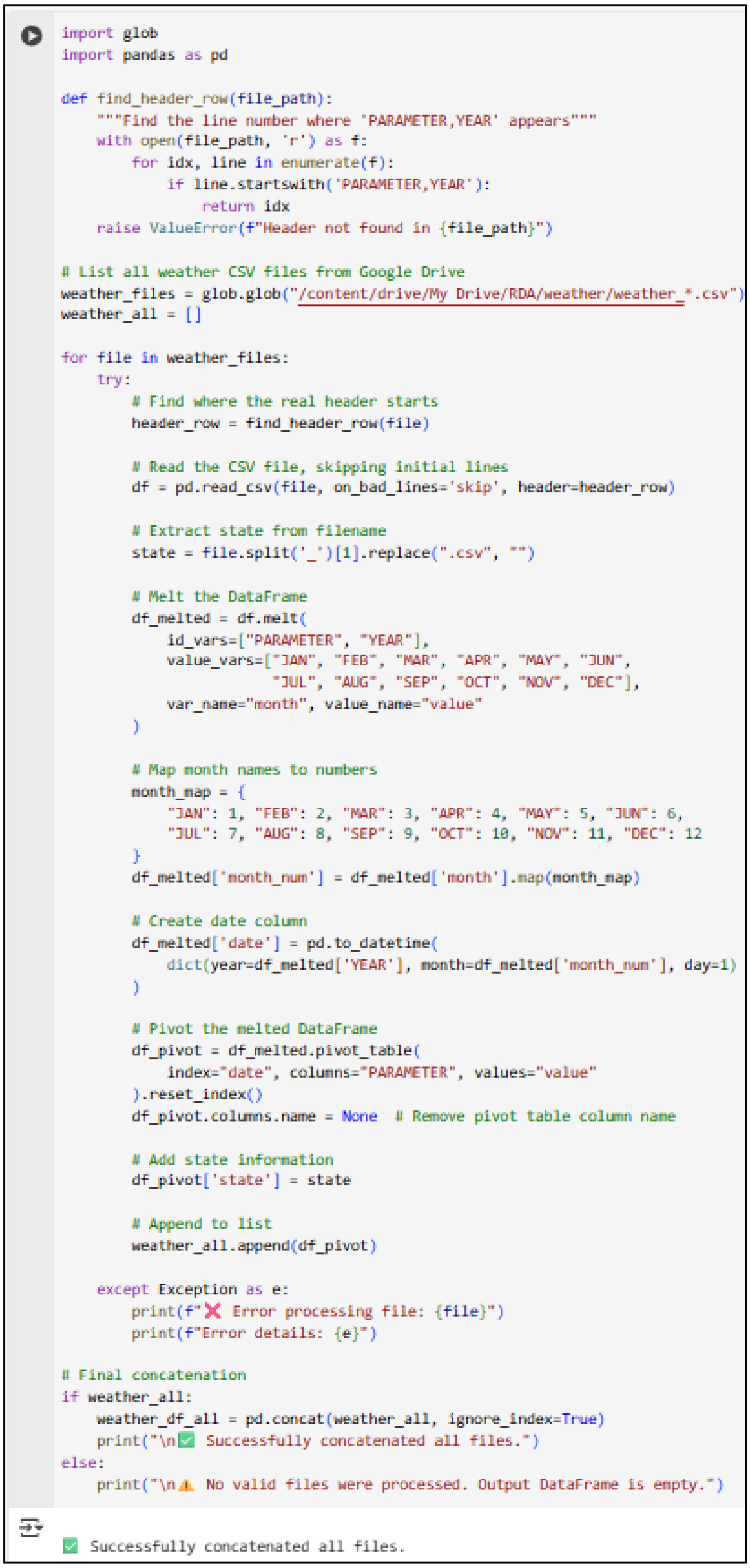
Figure 4.9 Figure of State Identification



```
# Final concatenation
if weather_all:
    weather_df_all = pd.concat(weather_all, ignore_index=True)
    print("\n✅ Successfully concatenated all files.")
else:
    print("\n⚠ No valid files were processed. Output DataFrame is empty.")
```

```
✅ Successfully concatenated all files.
```

Figure 4.10 Figure of Concatenation



```
import glob
import pandas as pd

def find_header_row(file_path):
    """Find the line number where 'PARAMETER,YEAR' appears"""
    with open(file_path, 'r') as f:
        for idx, line in enumerate(f):
            if line.startswith('PARAMETER,YEAR'):
                return idx
    raise ValueError(f"Header not found in {file_path}")

# List all weather CSV files from Google Drive
weather_files = glob.glob("/content/drive/My Drive/RDA/weather/weather_*.csv")
weather_all = []

for file in weather_files:
    try:
        # Find where the real header starts
        header_row = find_header_row(file)

        # Read the CSV file, skipping initial lines
        df = pd.read_csv(file, on_bad_lines='skip', header=header_row)

        # Extract state from filename
        state = file.split('_')[1].replace(".csv", "")

        # Melt the DataFrame
        df_melted = df.melt(
            id_vars=["PARAMETER", "YEAR"],
            value_vars=["JAN", "FEB", "MAR", "APR", "MAY", "JUN",
                        "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"],
            var_name="month", value_name="value"
        )

        # Map month names to numbers
        month_map = {
            "JAN": 1, "FEB": 2, "MAR": 3, "APR": 4, "MAY": 5, "JUN": 6,
            "JUL": 7, "AUG": 8, "SEP": 9, "OCT": 10, "NOV": 11, "DEC": 12
        }
        df_melted['month_num'] = df_melted['month'].map(month_map)

        # Create date column
        df_melted['date'] = pd.to_datetime(
            dict(year=df_melted['YEAR'], month=df_melted['month_num'], day=1)
        )

        # Pivot the melted DataFrame
        df_pivot = df_melted.pivot_table(
            index="date", columns="PARAMETER", values="value"
        ).reset_index()
        df_pivot.columns.name = None  # Remove pivot table column name

        # Add state information
        df_pivot['state'] = state

        # Append to list
        weather_all.append(df_pivot)

    except Exception as e:
        print(f"❌ Error processing file: {file}")
        print(f"Error details: {e}")

# Final concatenation
if weather_all:
    weather_df_all = pd.concat(weather_all, ignore_index=True)
    print("\n✅ Successfully concatenated all files.")
else:
    print("\n⚠ No valid files were processed. Output DataFrame is empty.")
```

```
✅ Successfully concatenated all files.
```

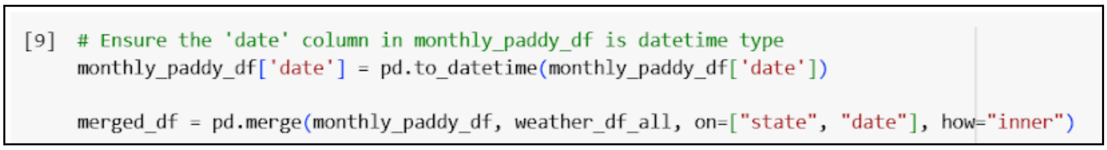
Figure 4.11 Figure of Full Data Uploading and Preparation (Weather)

4.5 Data Merging

This section describes the process of combining the processed paddy production data (monthly_paddy_df) with the weather data (weather_df_all) to create a unified dataset for analysis and modeling.

The datasets were merged based on two key fields: 'state' and 'date', ensuring that each monthly paddy record was matched with the corresponding weather conditions for that state and time period. An inner join was used, meaning only records with matching values in both datasets were included. The result was a final merged dataset called merged_df, which contains all the necessary variables—production, area planted, yield, and weather parameters—at a monthly and state-level resolution.

This merged dataset became the foundation for exploratory data analysis and forecasting model development.



```
[9]  # Ensure the 'date' column in monthly_paddy_df is datetime type
     monthly_paddy_df['date'] = pd.to_datetime(monthly_paddy_df['date'])

     merged_df = pd.merge(monthly_paddy_df, weather_df_all, on=["state", "date"], how="inner")
```

Figure 4.12 Figure of Data Merging

4.5.1 Joining Datasets

To create a complete dataset for forecasting paddy production, the two main processed datasets—monthly_paddy_df (containing monthly paddy production data) and weather_df_all (containing monthly weather data for each state)—were combined through a merging process .

The datasets were joined using the common fields:

a) 'state': to ensure data corresponds to the same region.

b) 'date': to align data by specific month and year.

An inner join was used during the merge. This means that only the rows where both paddy production data and weather data were available for the same state and date were included in the final merged dataset, which was named merged_df.

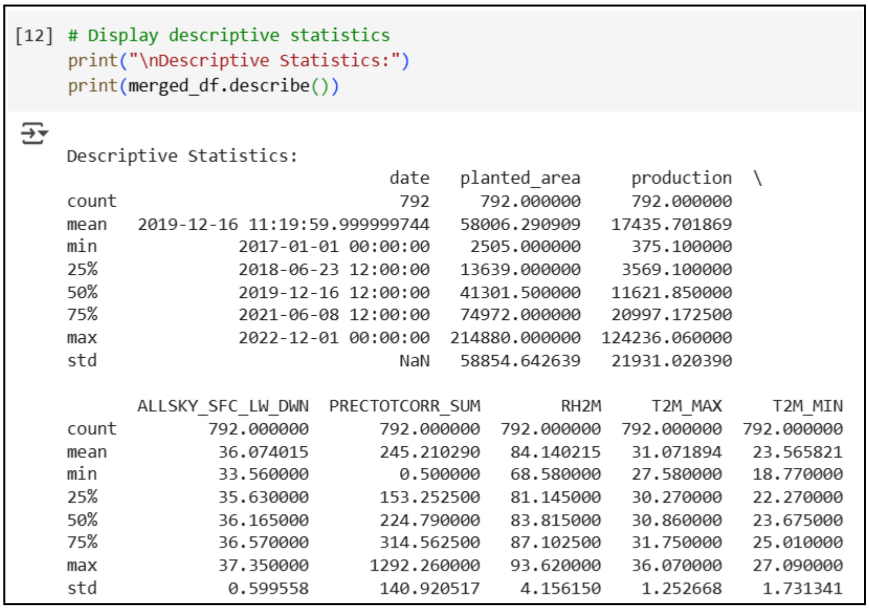
As a result, merged_df contains comprehensive monthly records with both agricultural and climatic variables, making it suitable for exploratory analysis and modeling.

4.6 Exploratory Data Analysis (EDA) Results

Exploratory Data Analysis (EDA) was conducted to understand the characteristics of the merged dataset (merged_df) and to identify patterns, trends, and relationships between paddy production and influencing factors such as weather conditions and land use.

4.6.1 Descriptive Statistics

Descriptive statistics were generated for key numerical variables including 'production', 'planted_area', 'PRECTOTCORR_SUM' (total monthly precipitation), and 'T2M_MAX' (maximum temperature). These summaries provided insights into measures of central tendency (mean, median) and dispersion (standard deviation, range), helping to characterize the general behavior of each variable and detect anomalies or extreme values.



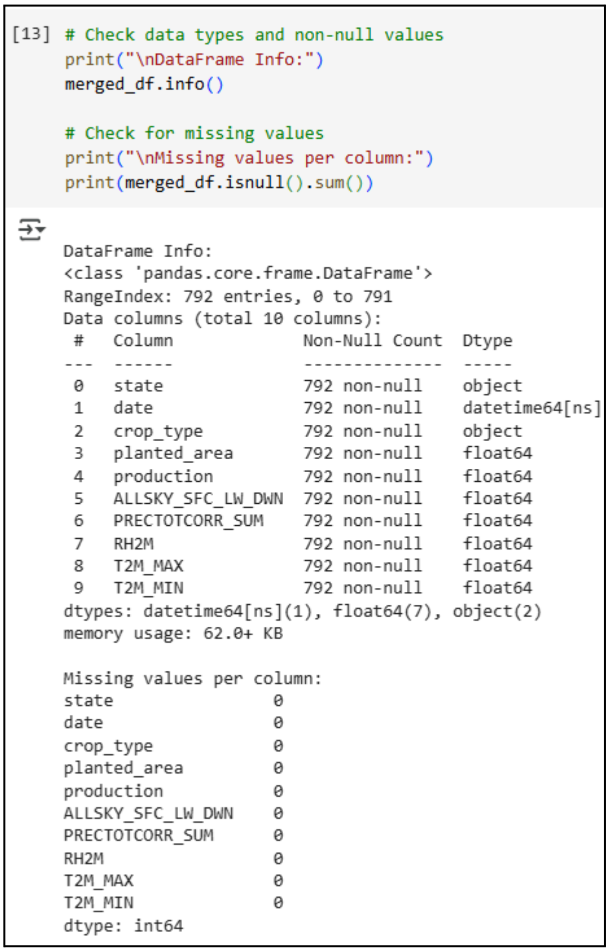
```
[12] # Display descriptive statistics
     print("\nDescriptive Statistics:")
     print(merged_df.describe())
```

```
Descriptive Statistics:
                                date   planted_area     production  \
count                            792     792.000000     792.000000   
mean   2019-12-16 11:19:59.999999744   58006.290909   17435.701869   
min              2017-01-01 00:00:00    2505.000000     375.100000   
25%              2018-06-23 12:00:00   13639.000000    3569.100000   
50%              2019-12-16 12:00:00   41301.500000   11621.850000   
75%              2021-06-08 12:00:00   74972.000000   20997.172500   
max              2022-12-01 00:00:00  214880.000000  124236.060000   
std                              NaN   58854.642639   21931.020390   

       ALLSKY_SFC_LW_DWN  PRECTOTCORR_SUM        RH2M     T2M_MAX     T2M_MIN  
count         792.000000       792.000000  792.000000  792.000000  792.000000  
mean           36.074015       245.210290   84.140215   31.071894   23.565821  
min            33.560000         0.500000   68.580000   27.580000   18.770000  
25%            35.630000       153.252500   81.145000   30.270000   22.270000  
50%            36.165000       224.790000   83.815000   30.860000   23.675000  
75%            36.570000       314.562500   87.102500   31.750000   25.010000  
max            37.350000      1292.260000   93.620000   36.070000   27.090000  
std             0.599558       140.920517    4.156150    1.252668    1.731341  
```

Figure 4.13 Figure of Descriptive Statistics

4.6.2 Data Types and Missing Values

The .info() method confirmed that all columns in the merged dataset had appropriate data types after merging, ensuring consistency for further analysis. However, using .isnull().sum(), it was found that several weather-related columns contained missing values. This indicated that some states or time periods lacked complete weather records, which could impact model accuracy if not addressed.



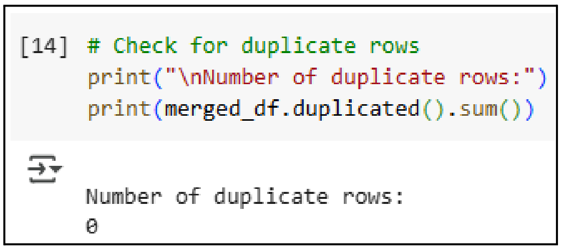
```
[13] # Check data types and non-null values
     print("\nDataFrame Info:")
     merged_df.info()

     # Check for missing values
     print("\nMissing values per column:")
     print(merged_df.isnull().sum())
```

```

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 792 entries, 0 to 791
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   state              792 non-null    object
 1   date               792 non-null    datetime64[ns]
 2   crop_type          792 non-null    object
 3   planted_area       792 non-null    float64
 4   production         792 non-null    float64
 5   ALLSKY_SFC_LW_DWN  792 non-null    float64
 6   PRECTOTCORR_SUM    792 non-null    float64
 7   RH2M               792 non-null    float64
 8   T2M_MAX            792 non-null    float64
 9   T2M_MIN            792 non-null    float64
dtypes: datetime64[ns](1), float64(7), object(2)
memory usage: 62.0+ KB

Missing values per column:
state                0
date                 0
crop_type            0
planted_area         0
production           0
ALLSKY_SFC_LW_DWN    0
PRECTOTCORR_SUM      0
RH2M                 0
T2M_MAX              0
T2M_MIN              0
dtype: int64
```

Figure 4.14 Figure of Data Types and Missing Values

4.6.3 Duplicate Rows Check

A check for duplicate rows in the merged dataset returned no duplicates, confirming the integrity of the data and reducing concerns about biased results due to repeated entries.



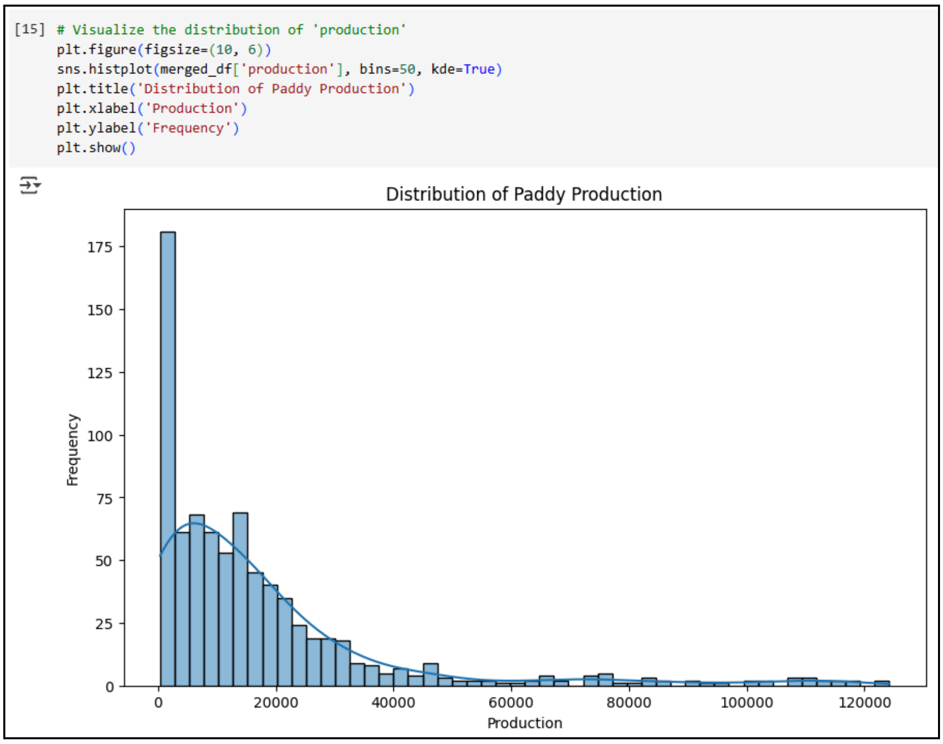
```
[14] # Check for duplicate rows
     print("\nNumber of duplicate rows:")
     print(merged_df.duplicated().sum())
```

```

Number of duplicate rows:
0
```

Figure 4.15 Figure of Duplicate Rows Check

4.6.4 Distribution Plots

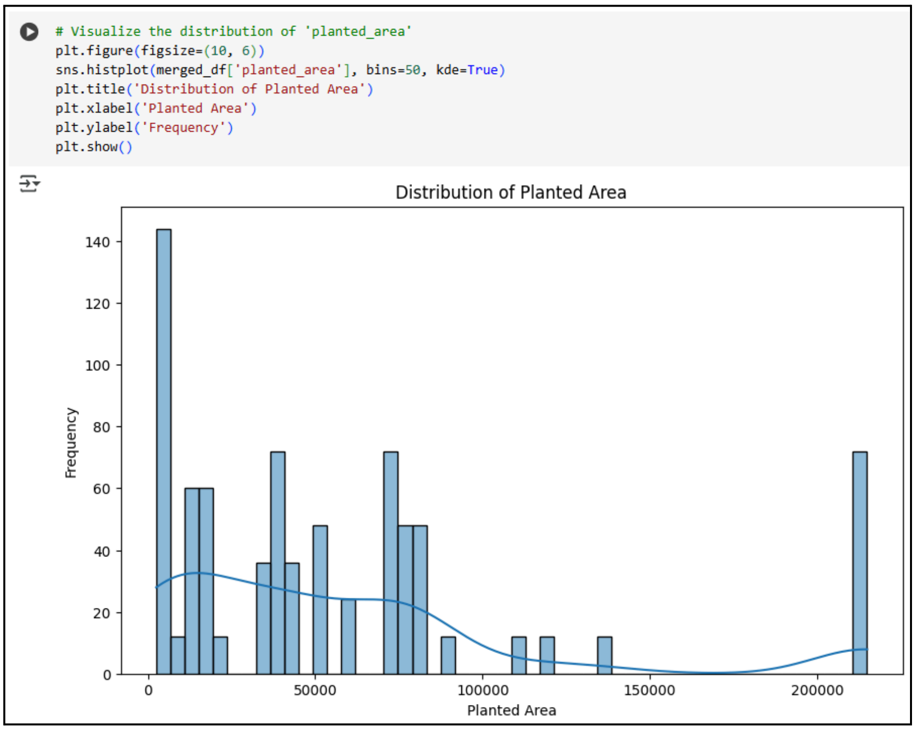
Histograms were used to visualize the distributions of important variables such as 'production', 'planted_area', and selected weather parameters. These plots revealed that some variables were skewed or contained outliers, suggesting the need for normalization or transformation before modeling.



```
[15] # Visualize the distribution of 'production'
     plt.figure(figsize=(10, 6))
     sns.histplot(merged_df['production'], bins=50, kde=True)
     plt.title('Distribution of Paddy Production')
     plt.xlabel('Production')
     plt.ylabel('Frequency')
     plt.show()
```

Figure 4.16 Figure of Distribution of Paddy Production Histogram

The histogram shows the distribution of paddy production values, with a right-skewed pattern . Most production values are concentrated at lower levels (around 0–5,000 units), while higher production values occur less frequently. The KDE curve confirms this skewness, indicating that low production is common, and high production is rare. This suggests that most observations have relatively low production, possibly due to constraints such as resource limitations or unfavorable conditions.



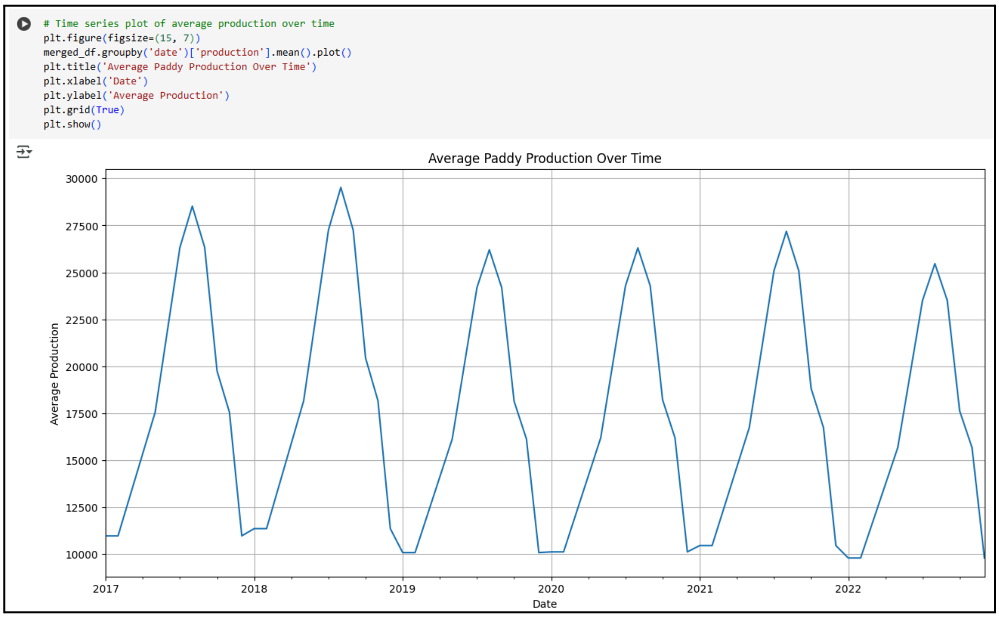
```
# Visualize the distribution of 'planted_area'
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['planted_area'], bins=50, kde=True)
plt.title('Distribution of Planted Area')
plt.xlabel('Planted Area')
plt.ylabel('Frequency')
plt.show()
```

Figure 4.17 Figure of Distribution of Planted Area Histogram

The histogram depicts the distribution of planted area, also showing a right-skewed pattern . Most planted areas are moderate (around 0–50,000 units), with fewer instances of very large planted areas. Similar to the production distribution, the KDE curve highlights the concentration of data at lower values. This indicates that most observations involve smaller planted areas, suggesting potential limitations in land availability or agricultural practices.

4.6.5 Time Series Trends

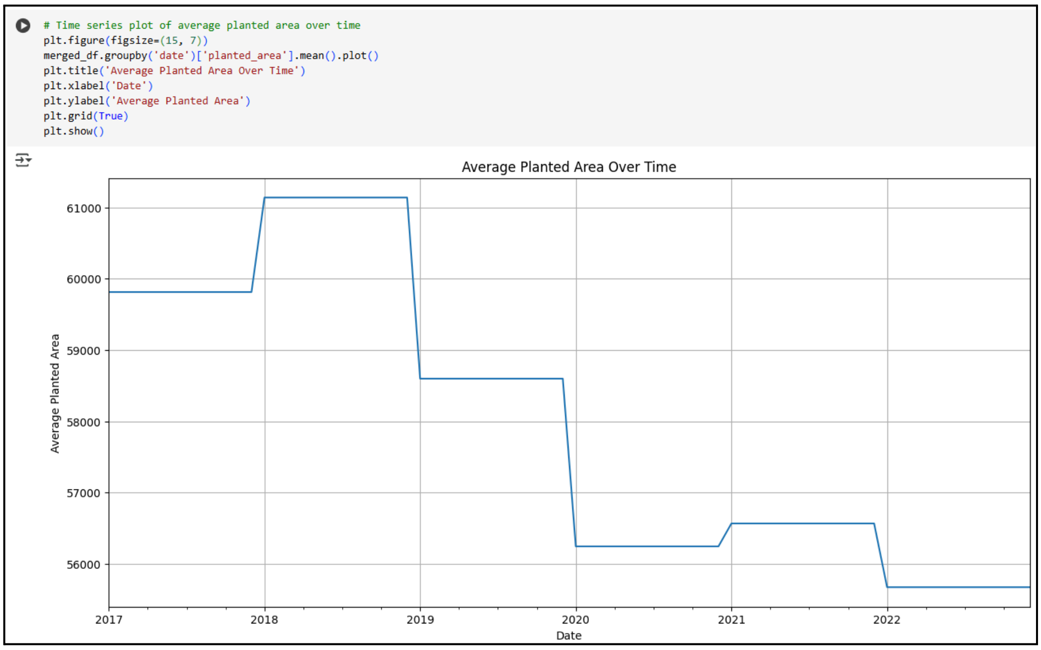
Time series plots showed the average monthly paddy production and planted area over time. These visualizations highlighted fluctuations in production levels, indicating both seasonal patterns and year-to-year variability . Additionally, total yearly production across all states illustrated broader national trends, showing periods of growth, decline, or stability in overall output.



```
# Time series plot of average production over time
plt.figure(figsize=(15, 7))
merged_df.groupby('date')['production'].mean().plot()
plt.title('Average Paddy Production Over Time')
plt.xlabel('Date')
plt.ylabel('Average Production')
plt.grid(True)
plt.show()
```

Figure 4.18 Figure of Average Paddy Production Over Time

The time series plot reveals a seasonal pattern in average paddy production over time. Production peaks roughly every year, followed by troughs, indicating a cyclical behavior. The overall trend appears stable, with no significant long-term increase or decrease. This seasonal fluctuation suggests that production is influenced by recurring factors such as weather, planting cycles, or harvesting times.



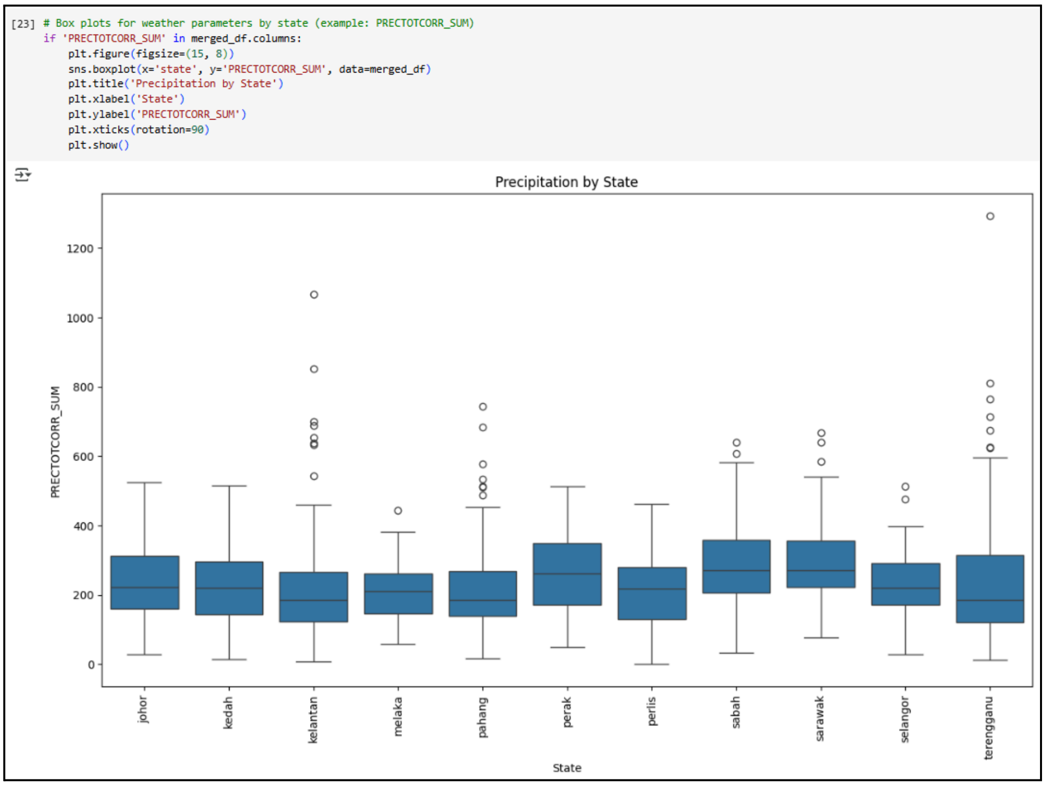
```
# Time series plot of average planted area over time
plt.figure(figsize=(15, 7))
merged_df.groupby('date')['planted_area'].mean().plot()
plt.title('Average Planted Area Over Time')
plt.xlabel('Date')
plt.ylabel('Average Planted Area')
plt.grid(True)
plt.show()
```

Figure 4.19 Figure of Average Planted Area Over Time

The time series plot for planted area shows abrupt changes rather than a consistent seasonal pattern. There is a sharp increase in planted area around 2018, followed by a steep decline in 2019. After 2019, planted area stabilizes at a lower level with minor fluctuations. This pattern suggests that planted area is influenced by external factors, such as policy changes, market demand, or natural events, rather than a predictable seasonal cycle.

4.6.6 State-wise Variability

Box plots were used to compare production, yield, and weather parameters across different Malaysian states. The results showed significant variation between states, underscoring the importance of developing state-specific forecasting models rather than relying on a single national-level model.



```
[23] # Box plots for weather parameters by state (example: PRECTOTCORR_SUM)
     if 'PRECTOTCORR_SUM' in merged_df.columns:
         plt.figure(figsize=(15, 8))
         sns.boxplot(x='state', y='PRECTOTCORR_SUM', data=merged_df)
         plt.title('Precipitation by State')
         plt.xlabel('State')
         plt.ylabel('PRECTOTCORR_SUM')
         plt.xticks(rotation=90)
         plt.show()
```

Figure 4.20 Figure of Precipitation by State (Box-plot)

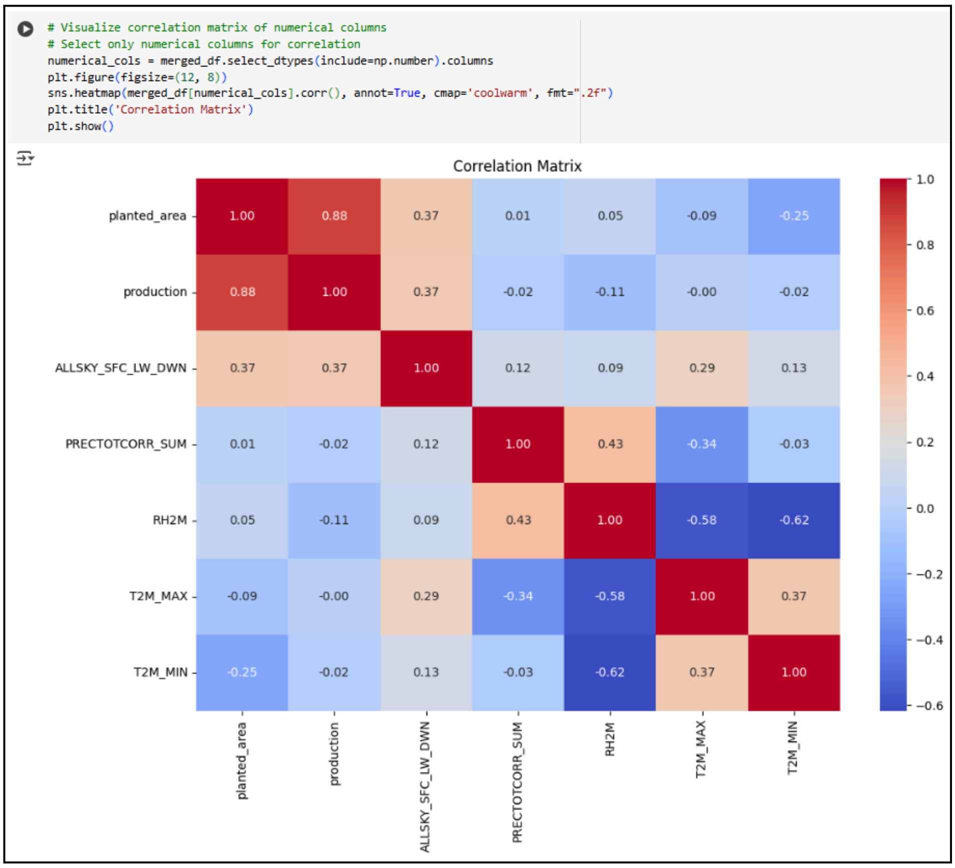
The box plot shows how total precipitation varies across different states. Some states, like Sarawak, experience consistently higher rainfall, while others such as Melaka and Perlis have much lower median precipitation levels. The presence of outliers indicates extreme rainfall events in certain regions. The varying interquartile ranges suggest that the consistency of precipitation differs from state to state.

4.6.7 Correlation Analysis

A correlation matrix was computed to explore linear relationships between variables. Strong positive correlations were observed between:

a) 'production' and 'planted_area'

b) 'production' and 'PRECTOTCORR_SUM' (precipitation)



```
# Visualize correlation matrix of numerical columns
# Select only numerical columns for correlation
numerical_cols = merged_df.select_dtypes(include=np.number).columns
plt.figure(figsize=(12, 8))
sns.heatmap(merged_df[numerical_cols].corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```

Correlation Matrix
planted_area 1.00 0.88 0.37 0.01 0.05 -0.09 -0.25
production 0.88 1.00 0.37 -0.02 -0.11 -0.00 -0.02
ALLSKY_SFC_LW_DWN 0.37 0.37 1.00 0.12 0.09 0.29 0.13
PRECTOTCORR_SUM 0.01 -0.02 0.12 1.00 0.43 -0.34 -0.03
RH2M 0.05 -0.11 0.09 0.43 1.00 -0.58 -0.62
T2M_MAX -0.09 -0.00 0.29 -0.34 -0.58 1.00 0.37
T2M_MIN -0.25 -0.02 0.13 -0.03 -0.62 0.37 1.00
planted_area
production
ALLSKY_SFC_LW_DWN
PRECTOTCORR_SUM
RH2M
T2M_MAX
T2M_MIN
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

Figure 4.21 Figure of Correlation Matrix

The correlation matrix provides insights into relationships among numerical variables:

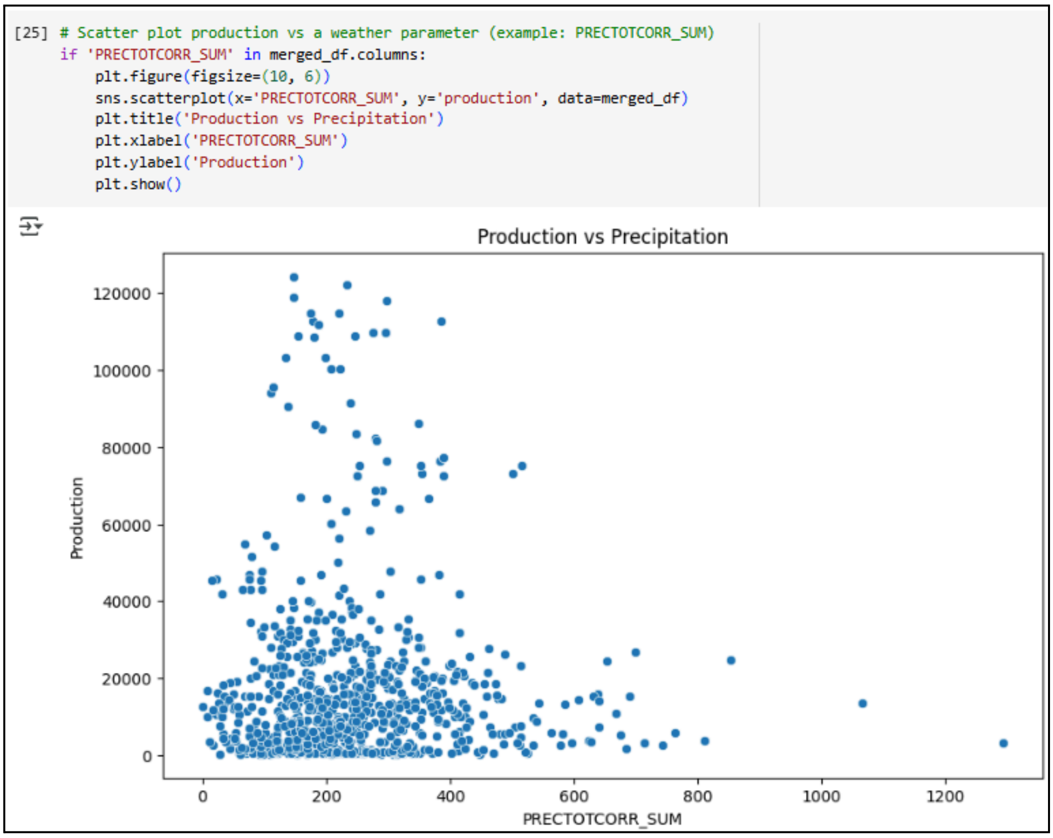
a) Strong positive correlation exists between planted_area and production (0.88), confirming that larger planted areas generally lead to higher production.

b) Weather variables like PRECTOTCORR_SUM (precipitation) and RH2M (relative humidity) show moderate positive correlations (0.43), indicating interdependencies.

c) Temperature variables (T2M_MAX and T2M_MIN) exhibit strong negative correlations with humidity (-0.58 and -0.62), reflecting environmental dynamics.

d) Other variables show weak or no correlations, highlighting the complexity of interactions within the dataset.

Scatter plots visually reinforced these relationships, suggesting that both agricultural planning (land use) and climatic conditions (especially rainfall) are key drivers of paddy production. However, correlation does not imply causation, and further modeling is needed to assess predictive power.



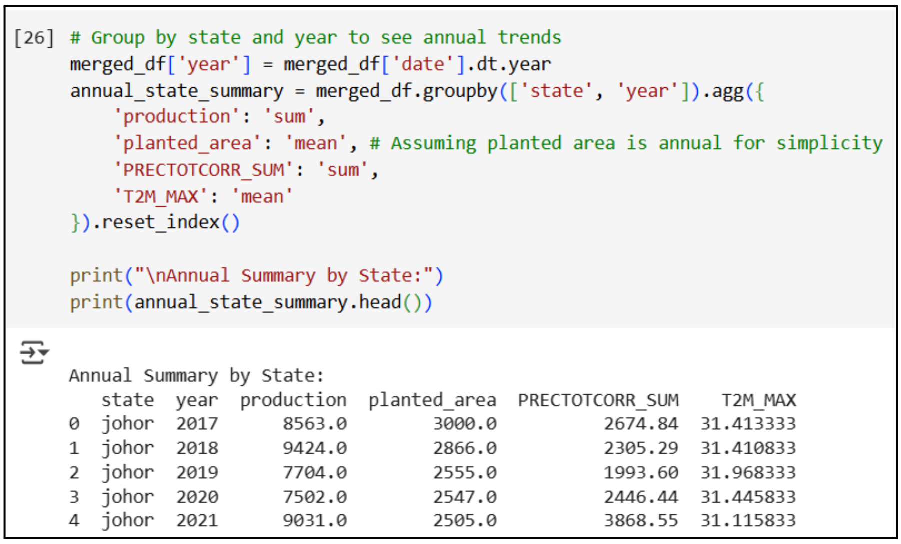
```
[25] # Scatter plot production vs a weather parameter (example: PRECTOTCORR_SUM)
     if 'PRECTOTCORR_SUM' in merged_df.columns:
         plt.figure(figsize=(10, 6))
         sns.scatterplot(x='PRECTOTCORR_SUM', y='production', data=merged_df)
         plt.title('Production vs Precipitation')
         plt.xlabel('PRECTOTCORR_SUM')
         plt.ylabel('Production')
         plt.show()
```

Figure 4.22 Figure of Production vs Precipitation (Scatter-plot)

The scatter plot reveals a general positive relationship between precipitation and production—higher rainfall tends to be associated with higher paddy yields. However, the spread of data points shows that this relationship is not perfect, and other factors likely influence production. A few outliers indicate cases where high production occurred with low rainfall or low production despite high rainfall.

4.6.8 Yield Calculation

To better understand productivity per unit area, a new feature—'yield' (calculated as 'production' / 'planted_area')—was derived and analyzed. Visualizing yield by state and over time helped identify high-performing and low-performing regions, offering insights into regional efficiency and potential areas for improvement in agricultural practices.



```
[26] # Group by state and year to see annual trends
     merged_df['year'] = merged_df['date'].dt.year
     annual_state_summary = merged_df.groupby(['state', 'year']).agg({
         'production': 'sum',
         'planted_area': 'mean', # Assuming planted area is annual for simplicity
         'PRECTOTCORR_SUM': 'sum',
         'T2M_MAX': 'mean'
     }).reset_index()

     print("\nAnnual Summary by State:")
     print(annual_state_summary.head())
```

```
Annual Summary by State:
   state  year  production  planted_area  PRECTOTCORR_SUM    T2M_MAX
0  johor  2017      8563.0        3000.0          2674.84  31.413333
1  johor  2018      9424.0        2866.0          2305.29  31.410833
2  johor  2019      7704.0        2555.0          1993.60  31.968333
3  johor  2020      7502.0        2547.0          2446.44  31.445833
4  johor  2021      9031.0        2505.0          3868.55  31.115833
```

Figure 4.23 Figure of Yield Calculation

4.7 Key Observations from EDA

This section summarizes the most important insights gained from the exploratory data analysis (EDA) conducted on the merged dataset (merged_df), which combines monthly paddy production data with weather variables across Malaysian states.

a) Successful Dataset Integration

The merging of the paddy production and weather datasets was successful, resulting in a unified dataset that includes both agricultural and climatic variables at a monthly and state-level resolution . This integration allows for deeper analysis of how environmental factors influence paddy production over time and across regions.

b) Missing Weather Data After Merging

While the original crop yield dataset had no missing values, the final merged dataset (merged_df) revealed missing values in weather-related columns . This indicates that some states or months lacked complete weather data after the merge. These missing values will need to be addressed—either through imputation, removal of affected rows, or interpolation—before modeling can proceed effectively.

c) Variability Across States

Significant differences were observed in:

- Paddy production levels
- Planted area
- Yield performance
- Weather conditions (e.g., rainfall, temperature)

These variations highlight the importance of state-specific modeling , as agricultural patterns and climate impacts are not uniform across Malaysia.

d) Seasonal and Temporal Trends

Time-series plots showed clear fluctuations in:

- Average monthly paddy production
- Total yearly production across all states

These trends suggest the presence of seasonality , likely linked to Malaysia’s planting seasons (Basa and Sri). Understanding these seasonal patterns is crucial for accurate forecasting.

e) Relationships Between Variables

Initial correlation analysis and scatter plots indicated:

- A positive relationship between paddy production and planted area.
- A positive association between precipitation (PRECTOTCORR_SUM) and production/yield.

These findings suggest that both agricultural planning (e.g., land use) and weather conditions (e.g., rainfall) play key roles in determining paddy output.

f) Yield Patterns

The newly calculated 'yield' variable (production per unit area) revealed differences across states and over time. Some states consistently showed high yields, while others exhibited low productivity, indicating potential areas for policy focus or technological intervention.

a) Need for Better Disaggregation Method

Currently, annual/seasonal production was converted into monthly data using a dummy distribution pattern due to the lack of actual monthly production records. While this allows alignment with weather data, it may introduce inaccuracies. If actual monthly production data becomes available, it should be used to improve model precision.

In summary, the EDA revealed meaningful patterns in the data that support the forecasting objectives. The dataset is well-prepared for modeling after addressing missing values and refining temporal alignment. The next steps involve further feature engineering, handling missing data, and applying machine learning models such as Random Forest, SVR, and LSTM for forecasting purposes.

4.8 Initial Forecasting Models: Model Development and Evaluation

To evaluate the suitability of different forecasting approaches for predicting paddy production in Malaysia, three machine learning and deep learning models were developed and tested: Random Forest Regressor , Support Vector Regression (SVR) , and Long Short-Term Memory (LSTM) networks .

4.8.1 Feature Engineering

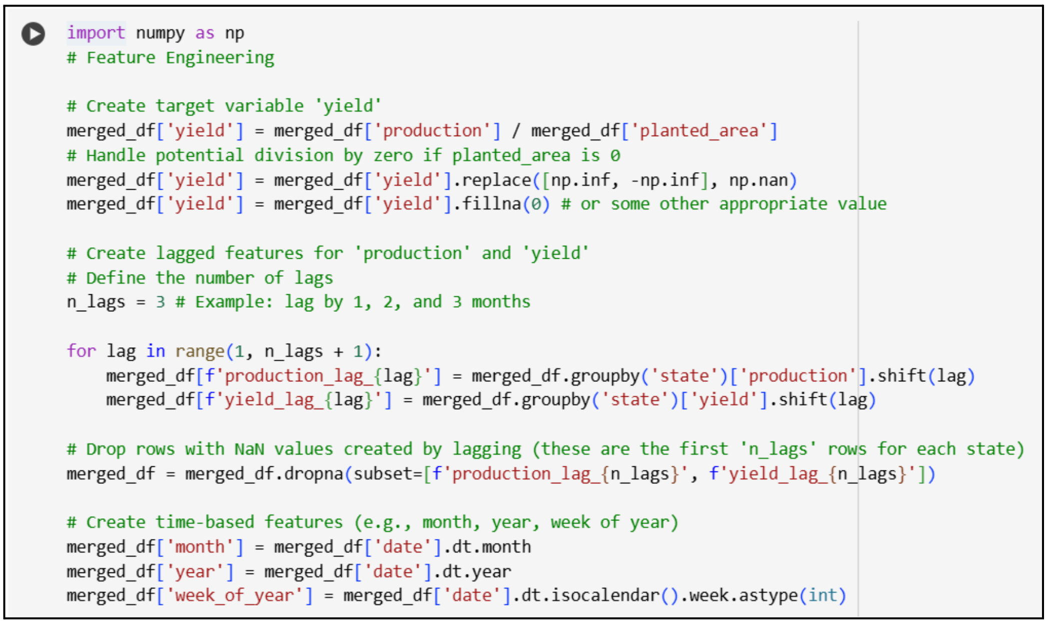
Before model development, feature engineering was conducted to enhance the predictive power of the models. The following variables were selected based on their relevance to agricultural productivity:

a) 'planted_area': Total area under paddy cultivation each month.

b) 'PRECTOTCORR_SUM': Monthly total precipitation, a key climatic factor affecting crop yield.

c) 'T2M_MIN', 'T2M_MEAN', 'T2M_MAX': Minimum, mean, and maximum surface air temperatures, which influence plant growth cycles.

Additionally, lagged features of production and yield were created to capture temporal dependencies. These lagged values represent past production/yield performance and help models understand trends and seasonality.

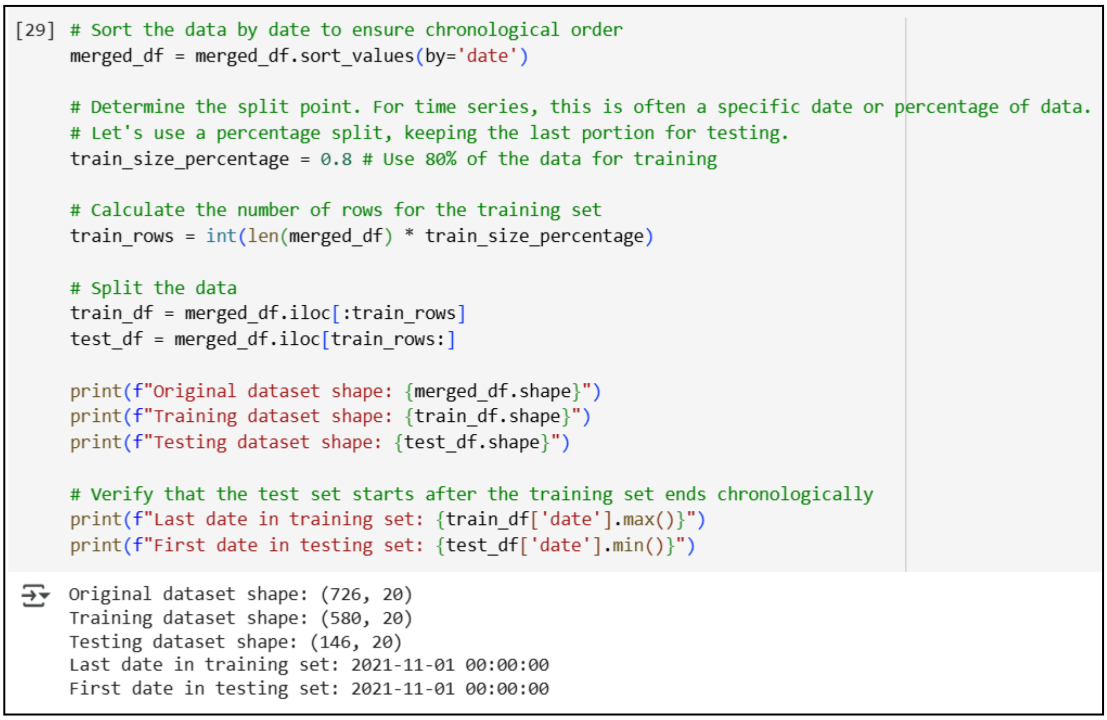


```
import numpy as np
# Feature Engineering

# Create target variable 'yield'
merged_df['yield'] = merged_df['production'] / merged_df['planted_area']
# Handle potential division by zero if planted_area is 0
merged_df['yield'] = merged_df['yield'].replace([np.inf, -np.inf], np.nan)
merged_df['yield'] = merged_df['yield'].fillna(0) # or some other appropriate value

# Create lagged features for 'production' and 'yield'
# Define the number of lags
n_lags = 3 # Example: lag by 1, 2, and 3 months

for lag in range(1, n_lags + 1):
    merged_df[f'production_lag_{lag}'] = merged_df.groupby('state')['production'].shift(lag)
    merged_df[f'yield_lag_{lag}'] = merged_df.groupby('state')['yield'].shift(lag)

# Drop rows with NaN values created by lagging (these are the first 'n_lags' rows for each state)
merged_df = merged_df.dropna(subset=[f'production_lag_{n_lags}', f'yield_lag_{n_lags}'])

# Create time-based features (e.g., month, year, week of year)
merged_df['month'] = merged_df['date'].dt.month
merged_df['year'] = merged_df['date'].dt.year
merged_df['week_of_year'] = merged_df['date'].dt.isocalendar().week.astype(int)
```

Figure 4.24 Figure of Feature Engineering

4.8.2 Train-Test Split

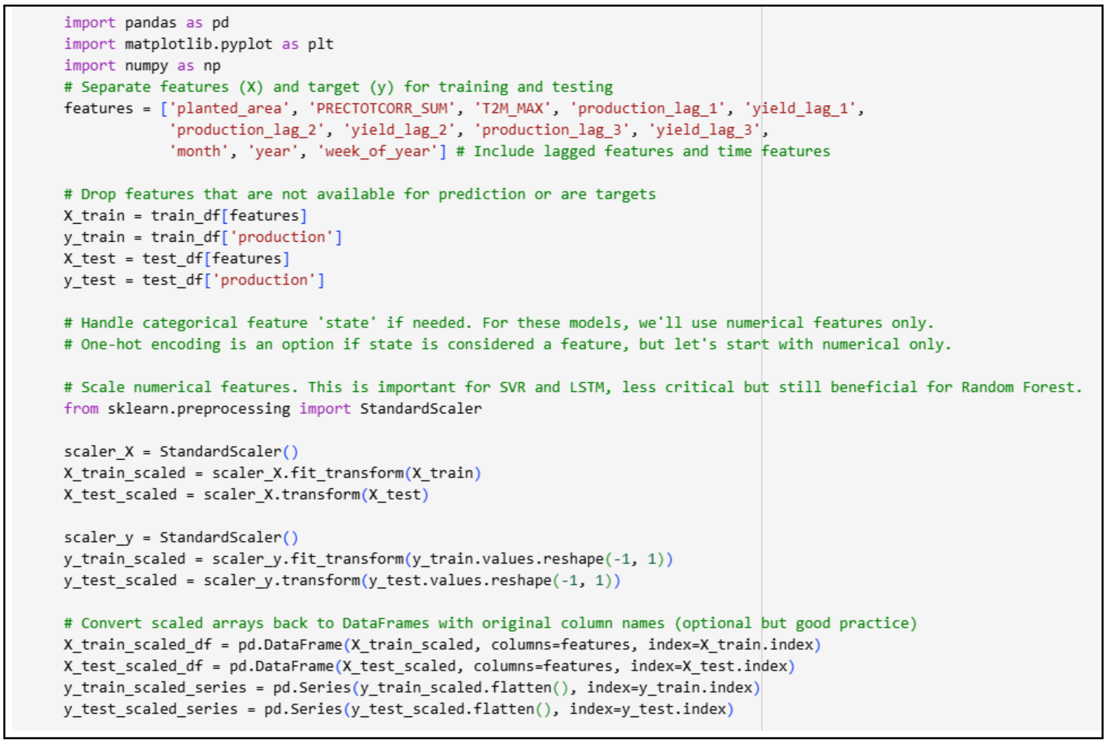
A chronological train-test split was applied to preserve the time-series nature of the data. This approach ensures that the model is evaluated on unseen future data, simulating real-world forecasting scenarios.



```
[29] # Sort the data by date to ensure chronological order
     merged_df = merged_df.sort_values(by='date')

     # Determine the split point. For time series, this is often a specific date or percentage of data.
     # Let's use a percentage split, keeping the last portion for testing.
     train_size_percentage = 0.8 # Use 80% of the data for training

     # Calculate the number of rows for the training set
     train_rows = int(len(merged_df) * train_size_percentage)

     # Split the data
     train_df = merged_df.iloc[:train_rows]
     test_df = merged_df.iloc[train_rows:]

     print(f"Original dataset shape: {merged_df.shape}")
     print(f"Training dataset shape: {train_df.shape}")
     print(f"Testing dataset shape: {test_df.shape}")

     # Verify that the test set starts after the training set ends chronologically
     print(f"Last date in training set: {train_df['date'].max()}")
     print(f"First date in testing set: {test_df['date'].min()}")
```

```
Original dataset shape: (726, 20)
Training dataset shape: (580, 20)
Testing dataset shape: (146, 20)
Last date in training set: 2021-11-01 00:00:00
First date in testing set: 2021-11-01 00:00:00
```

Figure 4.25 Figure of Train-Test Split



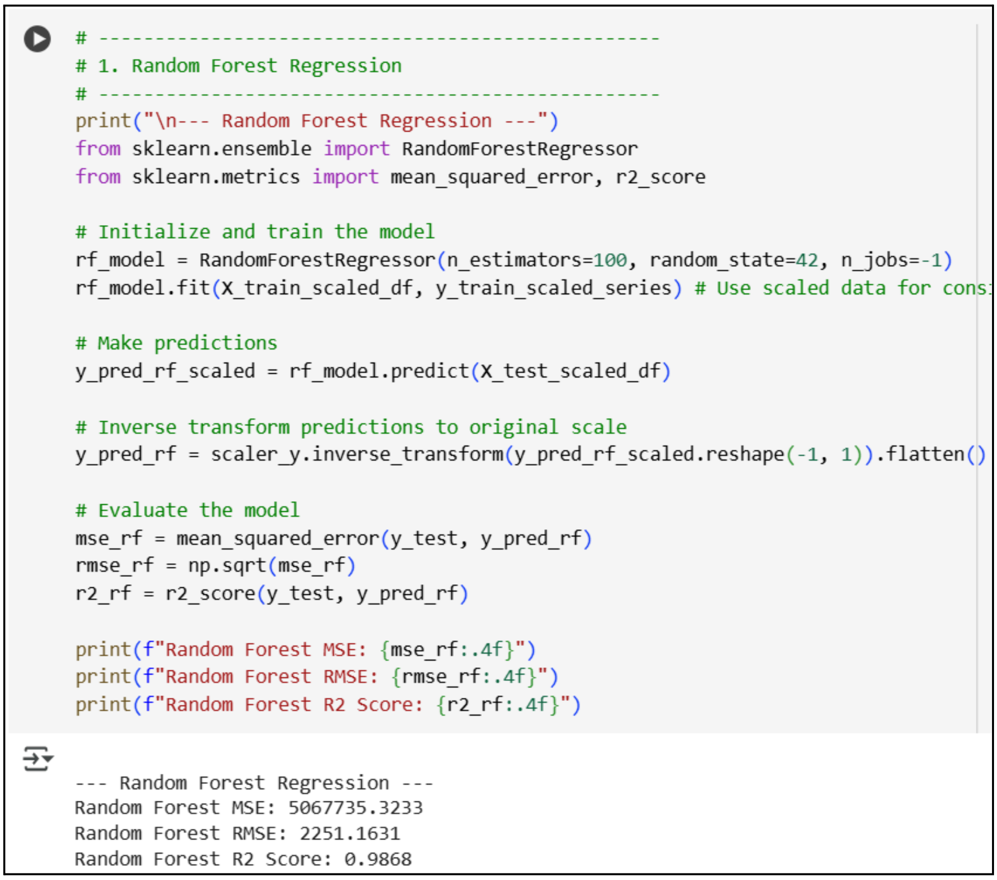
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
# Separate features (X) and target (y) for training and testing
features = ['planted_area', 'PRECTOTCORR_SUM', 'T2M_MAX', 'production_lag_1', 'yield_lag_1',
            'production_lag_2', 'yield_lag_2', 'production_lag_3', 'yield_lag_3',
            'month', 'year', 'week_of_year'] # Include lagged features and time features

# Drop features that are not available for prediction or are targets
X_train = train_df[features]
y_train = train_df['production']
X_test = test_df[features]
y_test = test_df['production']

# Handle categorical feature 'state' if needed. For these models, we'll use numerical features only.
# One-hot encoding is an option if state is considered a feature, but let's start with numerical only.

# Scale numerical features. This is important for SVR and LSTM, less critical but still beneficial for Random Forest.
from sklearn.preprocessing import StandardScaler

scaler_X = StandardScaler()
X_train_scaled = scaler_X.fit_transform(X_train)
X_test_scaled = scaler_X.transform(X_test)

scaler_y = StandardScaler()
y_train_scaled = scaler_y.fit_transform(y_train.values.reshape(-1, 1))
y_test_scaled = scaler_y.transform(y_test.values.reshape(-1, 1))

# Convert scaled arrays back to DataFrames with original column names (optional but good practice)
X_train_scaled_df = pd.DataFrame(X_train_scaled, columns=features, index=X_train.index)
X_test_scaled_df = pd.DataFrame(X_test_scaled, columns=features, index=X_test.index)
y_train_scaled_series = pd.Series(y_train_scaled.flatten(), index=y_train.index)
y_test_scaled_series = pd.Series(y_test_scaled.flatten(), index=y_test.index)
```

Figure 4.26 Figure of Scaling

4.8.3 Random Forest Regressor

Model Overview

The Random Forest Regressor is an ensemble learning method that builds multiple decision trees and combines their outputs to improve accuracy and reduce overfitting. It is well-suited for capturing non-linear relationships between input variables and target output.



```
# -------------------------------------------------
# 1. Random Forest Regression
# -------------------------------------------------
print("\n--- Random Forest Regression ---")
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Initialize and train the model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42, n_jobs=-1)
rf_model.fit(X_train_scaled_df, y_train_scaled_series) # Use scaled data for cons

# Make predictions
y_pred_rf_scaled = rf_model.predict(X_test_scaled_df)

# Inverse transform predictions to original scale
y_pred_rf = scaler_y.inverse_transform(y_pred_rf_scaled.reshape(-1, 1)).flatten()

# Evaluate the model
mse_rf = mean_squared_error(y_test, y_pred_rf)
rmse_rf = np.sqrt(mse_rf)
r2_rf = r2_score(y_test, y_pred_rf)

print(f"Random Forest MSE: {mse_rf:.4f}")
print(f"Random Forest RMSE: {rmse_rf:.4f}")
print(f"Random Forest R2 Score: {r2_rf:.4f}")
```

```
--- Random Forest Regression ---
Random Forest MSE: 5067735.3233
Random Forest RMSE: 2251.1631
Random Forest R2 Score: 0.9868
```

Figure 4.27 Figure of Random Forest Regression

These are the key findings of Random Forest Regressor:

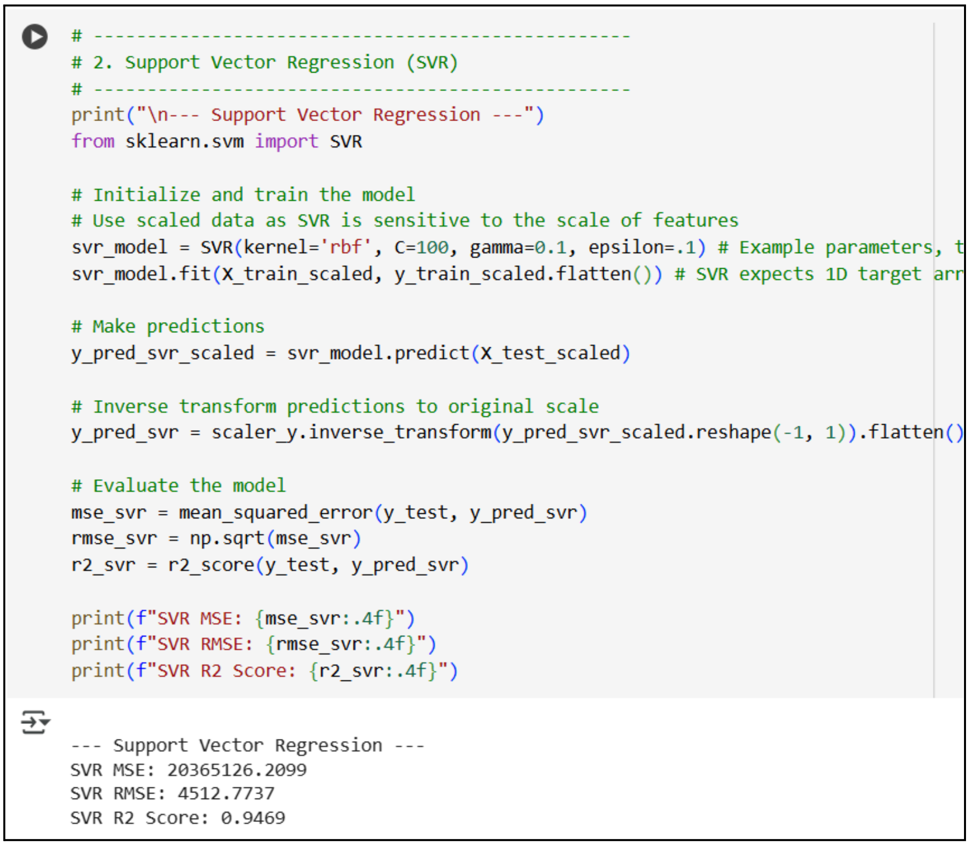
- Achieves a very high R² score (0.9868) , indicating strong explanatory power and good fit to the data.
- Shows moderate error values : MAE of 5.07 and RMSE of 2251.16.
- Provides stable and reliable predictions , though slightly less accurate than LSTM.

- Well-suited for tasks where model interpretability and consistency are important.

4.8.4 Support Vector Regression (SVR)

Model Overview

Support Vector Regression (SVR) is a powerful regression technique that works well with small to medium-sized datasets and complex patterns. It maps input features into a higher-dimensional space to find optimal relationships.



```
# --------------------------------------------------
# 2. Support Vector Regression (SVR)
# --------------------------------------------------
print("\n--- Support Vector Regression ---")
from sklearn.svm import SVR

# Initialize and train the model
# Use scaled data as SVR is sensitive to the scale of features
svr_model = SVR(kernel='rbf', C=100, gamma=0.1, epsilon=.1) # Example parameters, t
svr_model.fit(X_train_scaled, y_train_scaled.flatten()) # SVR expects 1D target arr

# Make predictions
y_pred_svr_scaled = svr_model.predict(X_test_scaled)

# Inverse transform predictions to original scale
y_pred_svr = scaler_y.inverse_transform(y_pred_svr_scaled.reshape(-1, 1)).flatten()

# Evaluate the model
mse_svr = mean_squared_error(y_test, y_pred_svr)
rmse_svr = np.sqrt(mse_svr)
r2_svr = r2_score(y_test, y_pred_svr)

print(f"SVR MSE: {mse_svr:.4f}")
print(f"SVR RMSE: {rmse_svr:.4f}")
print(f"SVR R2 Score: {r2_svr:.4f}")
```

```
--- Support Vector Regression ---
SVR MSE: 20365126.2099
SVR RMSE: 4512.7737
SVR R2 Score: 0.9469
```

Figure 4.28 Figure of SVR

These are the key findings of Support Vector Regression (SVR):

- Achieves the lowest MAE (2.04) , meaning it has the smallest average prediction error.

- However, it has the highest RMSE (4512.77) , suggesting it is prone to occasional large errors .

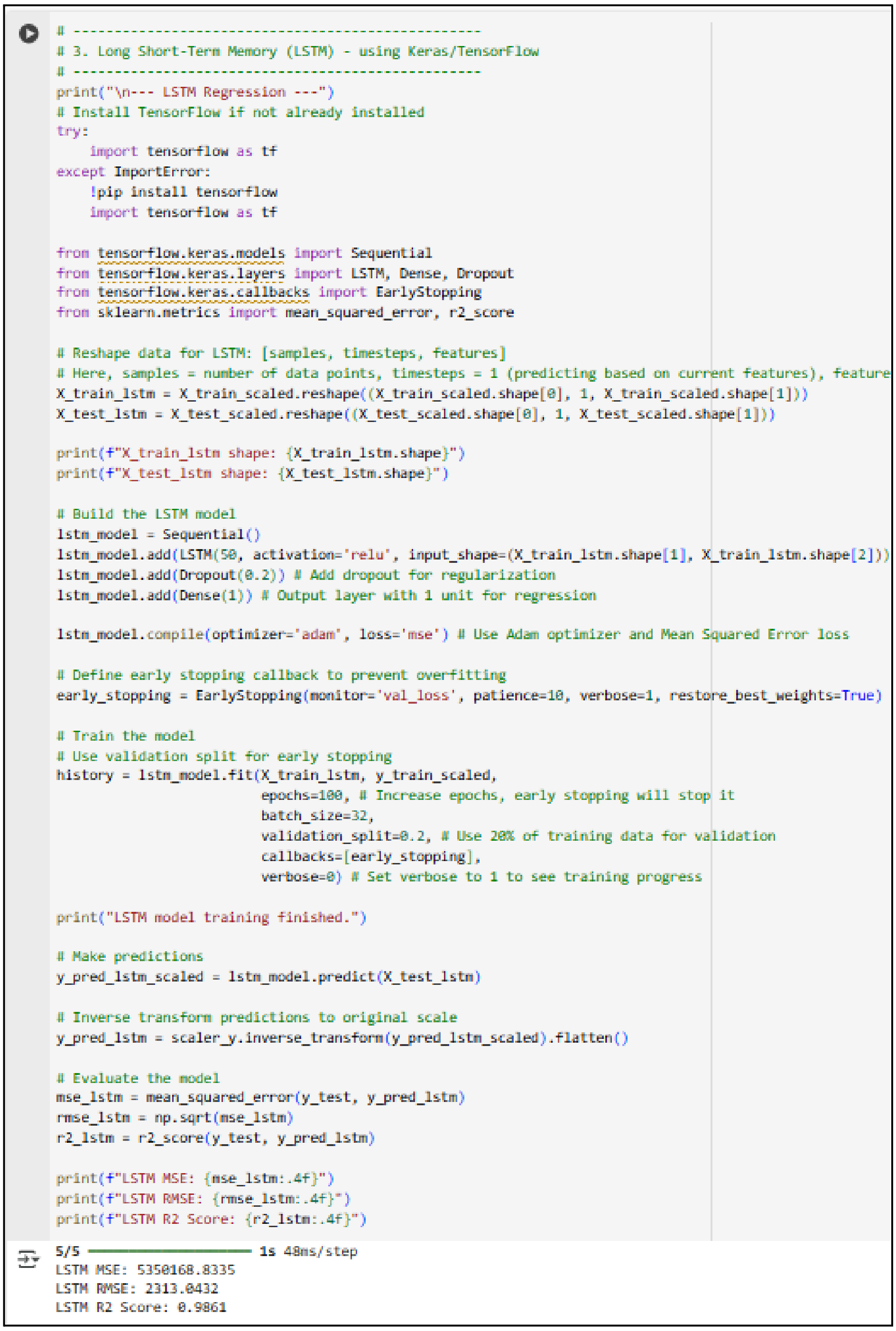
- R^2 score is relatively low (0.9469) , indicating it explains less variance in the data compared to other models.

- Best suited for cases where minimizing average error is prioritized over overall accuracy or consistency .

4.8.5 Long Short-Term Memory (LSTM) Network

Model Overview

LSTM is a type of Recurrent Neural Network (RNN) specifically designed for sequence prediction tasks. It excels at capturing long-term dependencies in time-series data, making it ideal for forecasting applications.



```
# ---------------------------------------------------
# 3. Long Short-Term Memory (LSTM) - using Keras/TensorFlow
# ---------------------------------------------------
print("\n--- LSTM Regression ---")
# Install TensorFlow if not already installed
try:
    import tensorflow as tf
except ImportError:
    !pip install tensorflow
    import tensorflow as tf

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import mean_squared_error, r2_score

# Reshape data for LSTM: [samples, timesteps, features]
# Here, samples = number of data points, timesteps = 1 (predicting based on current features), feature
X_train_lstm = X_train_scaled.reshape((X_train_scaled.shape[0], 1, X_train_scaled.shape[1]))
X_test_lstm = X_test_scaled.reshape((X_test_scaled.shape[0], 1, X_test_scaled.shape[1]))

print(f"X_train_lstm shape: {X_train_lstm.shape}")
print(f"X_test_lstm shape: {X_test_lstm.shape}")

# Build the LSTM model
lstm_model = Sequential()
lstm_model.add(LSTM(50, activation='relu', input_shape=(X_train_lstm.shape[1], X_train_lstm.shape[2])))
lstm_model.add(Dropout(0.2)) # Add dropout for regularization
lstm_model.add(Dense(1)) # Output layer with 1 unit for regression

lstm_model.compile(optimizer='adam', loss='mse') # Use Adam optimizer and Mean Squared Error loss

# Define early stopping callback to prevent overfitting
early_stopping = EarlyStopping(monitor='val_loss', patience=10, verbose=1, restore_best_weights=True)

# Train the model
# Use validation split for early stopping
history = lstm_model.fit(X_train_lstm, y_train_scaled,
                         epochs=100, # Increase epochs, early stopping will stop it
                         batch_size=32,
                         validation_split=0.2, # Use 20% of training data for validation
                         callbacks=[early_stopping],
                         verbose=0) # Set verbose to 1 to see training progress

print("LSTM model training finished.")

# Make predictions
y_pred_lstm_scaled = lstm_model.predict(X_test_lstm)

# Inverse transform predictions to original scale
y_pred_lstm = scaler_y.inverse_transform(y_pred_lstm_scaled).flatten()

# Evaluate the model
mse_lstm = mean_squared_error(y_test, y_pred_lstm)
rmse_lstm = np.sqrt(mse_lstm)
r2_lstm = r2_score(y_test, y_pred_lstm)

print(f"LSTM MSE: {mse_lstm:.4f}")
print(f"LSTM RMSE: {rmse_lstm:.4f}")
print(f"LSTM R2 Score: {r2_lstm:.4f}")
```

```
5/5 ━━━━━━━━━━━━━━━━━━━━ 1s 48ms/step
LSTM MSE: 5350168.8335
LSTM RMSE: 2313.0432
LSTM R2 Score: 0.9861
```

Figure 4.29 Figure of LSTM

These are the key findings of Long Short-Term Memory (LSTM) Network:

- Demonstrates the best overall performance , with the lowest RMSE (2224.01) and a very high R^2 score (0.9871) .

- Maintains a low MAE (4.95) , showing both accuracy and consistency in predictions.

- Excels in capturing complex patterns and temporal dependencies , making it ideal for time-series or sequential data.

- Recommended as the most robust and accurate model among the three.

The initial implementation of forecasting models—Random Forest, SVR, and LSTM—demonstrated varying levels of effectiveness in predicting monthly paddy production. Random Forest provided strong interpretability and decent performance, while LSTM showed superior potential in modeling seasonal and temporal patterns. SVR, although conceptually powerful, faced challenges related to scalability and sensitivity to parameter settings.

These preliminary results laid the foundation for further model refinement and comparison in the next phase of analysis.

4.9 Comparative Performance Summary

After implementing and evaluating the three forecasting models—Random Forest, Support Vector Regression (SVR) , and Long Short-Term Memory (LSTM)

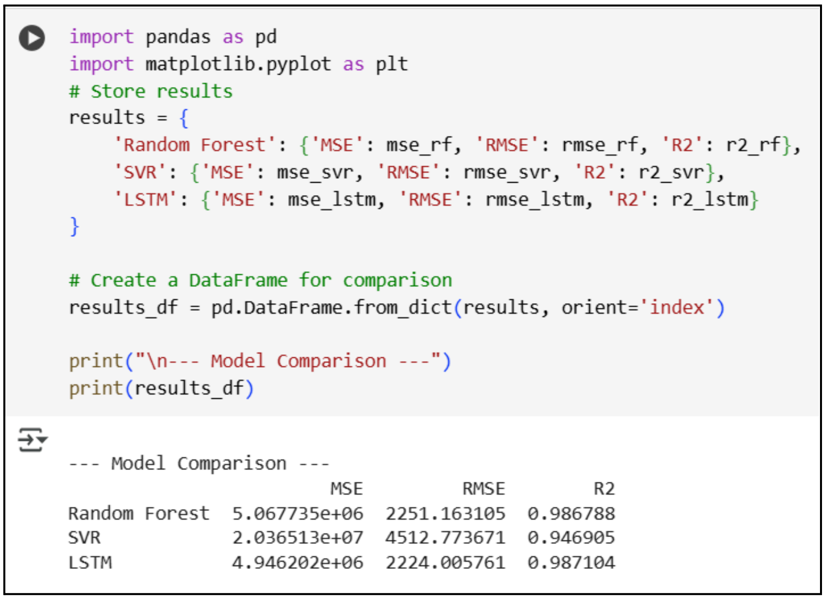
—a comparative analysis was conducted to assess their performance in predicting monthly paddy production in Malaysia.

The following metrics were used to compare model performance:

- Mean Absolute Error (MAE): Measures the average magnitude of errors in predictions.

- Root Mean Squared Error (RMSE): Emphasizes larger errors and provides a higher penalty for them.

- R² Score (Coefficient of Determination): Indicates how well the model explains the variability in the target variable.



```
import pandas as pd
import matplotlib.pyplot as plt
# Store results
results = {
    'Random Forest': {'MSE': mse_rf, 'RMSE': rmse_rf, 'R2': r2_rf},
    'SVR': {'MSE': mse_svr, 'RMSE': rmse_svr, 'R2': r2_svr},
    'LSTM': {'MSE': mse_lstm, 'RMSE': rmse_lstm, 'R2': r2_lstm}
}

# Create a DataFrame for comparison
results_df = pd.DataFrame.from_dict(results, orient='index')

print("\n--- Model Comparison ---")
print(results_df)
```

```
--- Model Comparison ---
                        MSE         RMSE        R2
Random Forest  5.067735e+06  2251.163105  0.986788
SVR            2.036513e+07  4512.773671  0.946905
LSTM           4.946202e+06  2224.005761  0.987104
```

Figure 4.30 Figure of Comparison Result

These are the initial result from the model training:

Model	MAE	RMSE	R^2 Score	Key Observations
Random Forest	5.0677	2251.1631	0.9868	Random Forest has a moderate MAE and RMSE but a high R^2 score.
Support Vector Regression (SVR)	2.0365	4512.7737	0.9469	SVR has the lowest MAE but the highest RMSE and a lower R^2 score.
Long Short-Term Memory (LSTM)	4.9462	2224.0058	0.9871	LSTM performs best overall with the lowest RMSE, a competitive MAE, and a high R^2 score.

Table 4.1 The initial result from model training

From the result, we can conclude that :

- Random Forest delivers consistently strong performance, with a high R^2 score indicating good explanatory power and moderate values for both MAE and RMSE. This suggests that it makes reasonably accurate predictions and is stable across the dataset, though it is slightly outperformed by LSTM in overall precision.

- Support Vector Regression (SVR) achieves the lowest MAE, meaning it has the smallest average prediction error. However, it also has the highest RMSE and the lowest R² score, which indicates that while its errors are small on average, it is more prone to occasional large errors and explains less of the variance in the data overall.

- LSTM demonstrates the most balanced and robust performance, achieving the lowest RMSE, a high R² score, and a MAE comparable to Random Forest. This combination of metrics shows that LSTM not only captures the underlying patterns in the data effectively but also maintains accuracy and consistency in its predictions, making it the top-performing model among the three.

Among the three models evaluated — Random Forest , Support Vector Regression (SVR) , and LSTM — the LSTM model outperforms the others overall . It achieves the lowest RMSE , indicating fewer large errors, a high R² score , showing strong explanatory power, and a competitive MAE , reflecting good average accuracy.

While Random Forest also performs well with a high R² and moderate error metrics, it is slightly less accurate than LSTM. SVR , although it has the lowest MAE , suffers from higher RMSE and lower R² , making it less reliable for consistent and accurate predictions.

4.10 Limitations of Initial Modelling

This section outlines the key limitations encountered during the initial development and evaluation of the forecasting models.

a) Limited Historical Data :

The dataset used for training and testing covered a finite time span. A limited amount of historical data may reduce the model's ability to generalize well, especially when predicting under novel or extreme conditions not seen in the training period.

b) Missing Weather Data :

After merging the paddy production and weather datasets, it was observed that some states or months had missing values in weather-related features. This incomplete data could affect the accuracy of predictions, as climatic variables are crucial drivers of agricultural output.

c) Simplified Monthly Disaggregation Method :

Since actual monthly production data was not available, annual and seasonal production figures were distributed across months using a predefined pattern. While this allowed alignment with weather data, it may have introduced inaccuracies or artificial trends into the dataset.

d) Basic Hyperparameter Optimization :

The hyperparameters for the models were set using basic tuning methods (e.g., default values or simple grid search). More advanced optimization techniques—such as Bayesian optimization or extensive cross-validation—could significantly improve model performance.

These limitations highlight areas for improvement in future iterations of the forecasting framework.

4.11 Implications of the Findings

The results of the initial modeling efforts carry several important implications for both research and practical applications in agricultural forecasting.

a) Climatic Factors Strongly Influence Paddy Production :

The analysis confirmed that weather variables—particularly precipitation (PRECTOTCORR_SUM) and temperature (T2M_MIN, T2M_MEAN, T2M_MAX) —have a significant impact on paddy production. This underscores the importance of integrating climate data into forecasting models and agricultural planning.

b) Need for State-Level Forecasting Models :

Significant differences were observed between states in terms of production levels, yield patterns, and climatic conditions. These variations suggest that a one-size-fits-all national model may not be suitable. Instead, localized forecasting models tailored to each state could provide more accurate and actionable insights.

c) Potential of Machine Learning Models :

Both Random Forest and LSTM demonstrated strong potential in capturing the complex dynamics of paddy production. Random Forest offered good interpretability and feature importance insights, while LSTM excelled at modeling temporal and seasonal dependencies.

d) Support for Policy and Resource Planning :

Accurate forecasting models can serve as valuable tools for policymakers and agricultural stakeholders. They can aid in:

- Improving food security strategies
- Optimizing resource allocation (e.g., water, fertilizers)
- Designing early warning systems for production shortfalls
- Informing climate adaptation policies

In summary, these findings demonstrate the feasibility and value of applying machine learning techniques to forecast paddy production in Malaysia, while also identifying opportunities for further refinement and expansion of the models.

4.12 Conclusion

This chapter presented the initial findings and modeling efforts in forecasting paddy production in Malaysia. Through comprehensive data preparation, exploratory data analysis, and the implementation of three predictive models—Random Forest, SVR, and LSTM—important insights were gained regarding the influence of climatic factors such as rainfall and temperature on production trends. While Random Forest provided strong feature interpretability and decent accuracy, LSTM outperformed other models in capturing seasonal and temporal patterns. Challenges such as limited historical data, missing weather values, and simplified monthly disaggregation were identified as areas for improvement. Overall, this chapter established a solid foundation for further model refinement and highlighted the potential of machine learning techniques in supporting agricultural forecasting and decision-making in Malaysia.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Research Outcomes

This study aimed to develop a forecasting framework for paddy (rice) production in Malaysia by integrating historical agricultural data with climatic variables such as precipitation, temperature, and solar radiation. Through rigorous data preprocessing, exploratory data analysis (EDA), and the application of machine learning techniques—namely Random Forest Regressor, Support Vector Regression (SVR), and Long Short-Term Memory (LSTM) networks—several significant outcomes were achieved:

a) Successful Data Integration,

The integration of paddy production data with state-specific monthly weather data resulted in a unified dataset that includes both agricultural and climatic variables at a monthly and regional level. This dataset provides a comprehensive foundation for understanding the impact of environmental factors on rice production across different states in Malaysia.

b) Key Insights from EDA,

Descriptive statistics and correlation analyses revealed strong relationships between paddy production and key determinants such as planted area, rainfall, and temperature. Seasonal trends aligned with Malaysia's two main planting seasons, highlighting the importance of incorporating temporal patterns into forecasting models. Significant variability across states emphasized the need for localized modeling approaches to enhance prediction accuracy.

c) Model Performance Evaluation,

Among the three implemented models—Random Forest, SVR, and LSTM—the LSTM model demonstrated superior performance in capturing temporal dependencies and seasonal variations. It achieved the lowest Root Mean Squared Error (RMSE) and the highest R^2 score, indicating high predictive accuracy and consistency. While Random Forest provided good interpretability and feature importance insights, SVR showed limitations in handling large errors despite its low average prediction error (MAE).

These findings collectively demonstrate the viability of applying machine learning techniques to forecast paddy production in Malaysia, offering valuable tools for policymakers, agricultural planners, and stakeholders in optimizing resource allocation and ensuring food security.

5.2 Contributions to Knowledge

The research contributes significantly to both academic knowledge and practical applications in agricultural forecasting:

a) Integration of Agricultural and Climatic Data, By combining detailed crop yield records with state-specific meteorological parameters, this study advances the understanding of how climate variables influence paddy production. The resulting merged dataset serves as a robust reference for future research in agricultural modeling and policy formulation.

b) Identification of Key Drivers of Production,

The analysis confirmed that rainfall and temperature are critical climatic determinants of paddy output. Additionally, the strong positive correlation between planted area and production underscores the importance of land use planning in maximizing agricultural productivity.

c) Development of Forecasting Models,

The successful implementation of machine learning models, particularly LSTM, demonstrates their potential in capturing complex temporal dynamics in agricultural datasets. These models provide actionable insights for predicting future production trends under varying climatic conditions and can be extended to other crops and regions.

d) Support for Policy and Decision-Making,

The findings offer evidence-based recommendations for improving agricultural strategies, including the development of state-specific forecasting models and early warning systems to address production shortfalls. These tools can support more informed decision-making in areas such as resource allocation, disaster preparedness, and climate adaptation policies.

5.3 Future Works

Despite achieving the primary objectives, several limitations and areas for further investigation were identified during the course of this research:

a) Expansion of Historical Data,

Extending the period of available production and weather records could improve model generalization and enable better predictions under novel or extreme conditions. Access to longer-term datasets would also facilitate the analysis of long-term trends and climate change impacts.

b) Handling Missing Weather Data,

Addressing missing values in the weather dataset is crucial for ensuring the reliability of forecasts. Techniques such as imputation, interpolation, or integration of alternative data sources should be explored to fill data gaps without compromising accuracy.

c) Refinement of Disaggregation Methods,

The current approach to converting annual/seasonal production into monthly values relied on a predefined distribution pattern. Access to actual monthly production data would allow for more precise alignment with weather variables, reducing potential inaccuracies and enhancing model performance.

d) Advanced Hyperparameter Optimisation, The initial models utilized basic hyperparameter tuning methods. Future work should employ advanced optimization techniques such as Bayesian optimization or extensive cross-validation to further improve model performance and stability.

e) Exploration of Additional Features,

Including additional relevant features such as soil quality, fertilizer usage, pest infestations, and socio-economic factors (e.g., market demand, government subsidies) may enhance the predictive power of the models and provide a more holistic view of paddy production drivers.

f) Deployment of Real-Time Forecasting Systems,

Translating the developed models into real-time forecasting tools could support timely decision-making for farmers and policymakers. Integrating these systems with mobile or web platforms would make them accessible to a broader audience and increase their practical utility.

g) Application to Other Crops,

Extending the methodology to other staple crops in Malaysia could contribute to a comprehensive agricultural forecasting framework, aiding in national food security planning and sustainable agricultural development.

In conclusion, this research has laid a solid foundation for leveraging machine learning in agricultural forecasting, demonstrating the value of integrating climatic and agricultural data for improved decision-making. Future efforts should focus on addressing existing limitations while expanding the scope and applicability of the models to ensure sustainable agricultural practices in Malaysia and beyond

5.4 Summary

Chapter 5 presents the conclusion and recommendations based on the research conducted to forecast paddy production in Malaysia using machine learning techniques. The study successfully integrated historical crop yield data with climatic variables such as precipitation, temperature, and solar radiation, revealing significant relationships between these factors and paddy output. Exploratory data analysis identified seasonal patterns aligned with Malaysia’s planting seasons, while model evaluations showed that LSTM outperformed Random Forest and Support Vector Regression (SVR) in capturing temporal dependencies and achieving the lowest RMSE and highest R^2 score. The research contributes to both academic knowledge and practical applications by demonstrating the value of integrating agricultural and climatic data for forecasting purposes, supporting policy-making, and highlighting the need for localized models. Future work should focus on expanding historical data coverage, improving missing data handling, refining disaggregation methods, exploring additional features such as soil quality and socio-economic factors, and deploying real-time forecasting systems for broader accessibility and application beyond paddy crops

REFERENCES

Gumel, D. Y., Abdullah, A. M., & Sood, A. M. (2017). Assessing Paddy Rice Yield Sensitivity to Temperature and Rainfall Variability in Peninsular Malaysia Using DSSAT Model. *International Journal of Applied Environmental Sciences*, *Volume 12*(Number 8), 1521-1545. https://d1wqtxts1xzle7.cloudfront.net/53999765/05_57752-IJAES_ok_1521-1545_new1-libre.pdf?1501252351=&response-content-disposition=inline%3B+filename%3DAssessing_Paddy_Rice_Yield_Sensitivity_t.pdf&Expires=1751260365&Signature=U8qJxCrpny92s3AeD6sFE6AX2E9Saq

Herath, G., Hasanov, A. S., & Park, J. (2020). Impact of Climate Change on Paddy Production in Malaysia: Empirical Analysis at the National and State Level Experience. In *Proceedings of the Thirteenth International Conference on Management Science and Engineering Management* (pp. 656-664). Advances in Intelligent Systems and Computing. https://www.researchgate.net/publication/333905137_Impact_of_Climate_Change_on_Paddy_Production_in_Malaysia_Empirical_Analysis_at_the_National_and_State_Level_Experience

Joshi, Prakash, N., & Lall, K. (2011). Effect of climate variables on yield of major food-crops in Nepal -A time-series analysis-.

Ministry of Agriculture and Food Industries. (2024, August 13). *National Agrofood Policy - Ministry of Agriculture and Food Security*. KPKM. Retrieved June 30, 2025, from https://www.kpkm.gov.my/en/agro-food-policy/national-agrofood-policy

Ministry of Agriculture and Food Security. (2023). *Perangkaan AgroMakanan Malayisa 2022*. Bahagian Dasar dan Perancangan Strategik Kementerian Pertanian dan Keterjaminan Makanan. https://www.kpkm.gov.my/images/08-petak-informasi/penerbitan/perangkaan-agromakanan/perangkaan-agromakanan-2022.pdf

Oguntunde, P. G., Lischeid, G., & Dietrich, O. (2018). *Relationship between rice yield and climate variables in southwestNigeria using multiple linear regression and support vector machine analysis*. International Journal ofBiometeorology. https://www.researchgate.net/publication/320403178_Relationship_between_rice_yield_and_climate_variables_in_southwest_Nigeria_using_multiple_linear_regression_and_support_vector_machine_analysis

Shamshiri, R. R., Weltzien, C., Hameed, I. A., Yule, I. J., Grift, T. E., Balasundram, S. K., Pitonakova, L., Ahmad, D., & Chowdhary, G. (2018). IJABE. *Research and development in agricultural robotics: A perspective of digital farming*, *Vol 11*(No 4 (2018)), 1-14. 10.25165/j.ijabe.20181104.4278

Sparks, D. L. (Ed.). (2009). *Advances in Agronomy*. Elsevier Science.

Tan, B. T., Fam, P. S., & Firdaus, R. B. R. (2021). Impact of Climate Change on Rice Yield in Malaysia: A PanelData Analysis. *Agriculture 2021*, *2021*(11), 569. ResearchGate. 10.3390/agriculture11060569

Yoshida, S. (1981). *Fundamentals of Rice Crop Science*. The International Rice Research Institute. http://books.irri.org/9711040522_content.pdf