AI AND SUSTAINABILITY – EEEM073

COUSEWORK

By

Behram Khanzada
(6890362)

MAY 2025

UNIVERSITY OF
SURREY

Department of computer science
MSc Artificial Intelligence

1. Introduction

1.1 Problem Definition and Relevance to AI and Sustainability

Rainfall forecasting plays a critical role in supporting several of the United Nations Sustainable Development Goals (UNSDGs), including climate action (SDG 13), sustainable cities and communities (SDG 11), and responsible consumption and production (SDG 12) [1]. In Australia, where rainfall is highly variable and many regions face long dry spells, the ability to accurately forecast rain contributes to water resource management, agriculture planning, and disaster preparedness.

This project applies AI for sustainability by developing a rain prediction system using deep learning techniques on the weatherAUS dataset [2]. The goal is to explore how temporal sequence modelling via recurrent neural networks can enhance short-term weather forecasting, especially for rain detection, which is inherently imbalanced in Australian climates.

1.2 Objectives

This project aims to:

- Develop an AI-based rain forecasting model for selected Australian locations using weekly-aggregated weather data.
- Evaluate the performance of a **Bidirectional LSTM model** in predicting rain occurrence for the following week based on the prior six weeks.
- Analyze model performance per month and per location, to identify seasonal and geographic strengths/weaknesses.
- Apply **oversampling, threshold tuning**, and **visual analytics** to handle data imbalance and improve model reliability.
- Link the solution to sustainable development goals by supporting climate resilience through smarter weather prediction.

1.3 Dataset Overview

The dataset used is the open source **weatherAUS** dataset from the Australian Government Bureau of Meteorology, which records historical weather observations from **49 locations** across the country. It includes:

- **145,460 samples** total
- **18 meteorological features**, such as temperature, humidity, pressure, wind speed, and cloud cover
- A binary target variable RainTomorrow, indicating if rain occurred the next day

The dataset was aggregated into weekly averages per location, and sequences of six weeks were used as model input to predict rain in the 7th week.

1.4 Approach Summary

The key steps in this project include:

- Data preprocessing: parsing dates, handling missing values, weekly grouping, and scaling
- Sequence generation per location with a 6-week look-back window
- Model development using a **Bidirectional LSTM** trained with **Focal Loss**
- Handling class imbalance using oversampling and decision threshold tuning
- Evaluation using **accuracy, precision, recall, F1, ROC AUC**, and visual tools such as ROC curves, confusion matrices, and PR curves
- Analysis of monthly and per-location model performance, along with a discussion of seasonal patterns and model behaviour

LSTMs are chosen due to their strength in modelling long-term dependencies in sequential data, which is critical for capturing lagging climatic indicators over multiple weeks.

2. Data Understanding, Pre-processing and Exploration

2.1 Dataset Overview and Initial Inspection

The dataset used in this project is the publicly available **weatherAUS** dataset, which contains over 145,000 rows of daily weather observations collected from 49 Australian locations. Each record includes 18 meteorological features alongside a binary target column RainTomorrow indicating if it rained the next day.

Key characteristics:

- Temporal coverage: several years of historical data
- Geographic diversity: includes both coastal and inland cities
- Target variable: RainTomorrow (binary classification)
- Predictor variables: temperature, rainfall, humidity, wind speed, pressure, cloud cover, and RainToday

The data was loaded using Pandas and cleaned by:

- Stripping whitespace from column headers
- Parsing dates in the Date column
- Dropping rows with missing values across selected key features and target

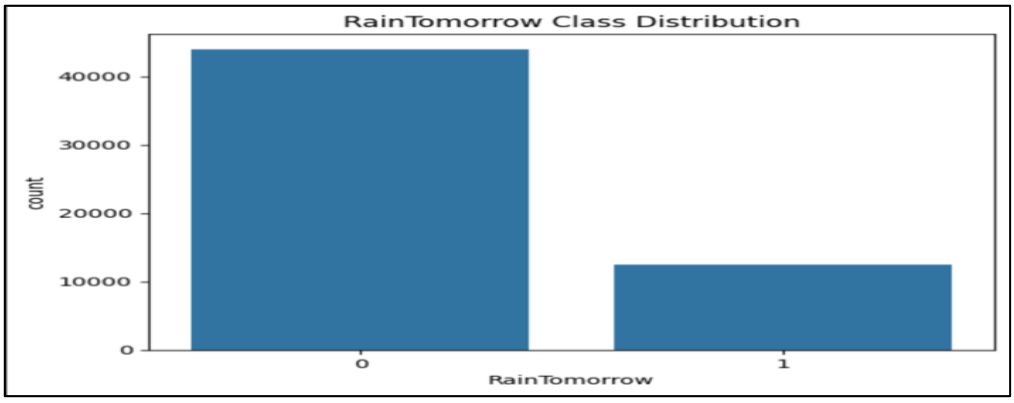


Figure 1 RainTomorrow Class Distribution

Figure 1 presents the class distribution of RainTomorrow, highlighting a strong class imbalance, where dry days far exceed rainy ones.

2.2 Data Pre-processing and Cleaning

A comprehensive set of pre-processing steps was applied:

- **Handling Missing Data**:
  Observations with missing values in critical columns (features or target) were removed. This was done conservatively to avoid introducing bias from imputation in a highly imbalanced binary classification task.
- **Date Handling and Temporal Structuring**:
  The Date column was parsed into datetime format. New columns Week and Year were derived using ISO calendar functions to support **weekly aggregation**, a key aspect of this project's time-series modelling approach.
- **Target Mapping**:
  Categorical values in RainToday and RainTomorrow (Yes/No) were encoded as 1/0 for compatibility with ML models.

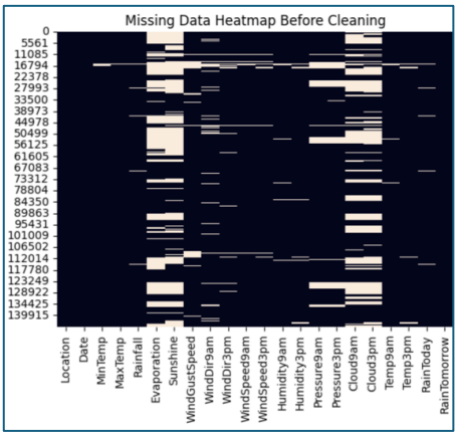


Figure 2(a) Missing Data Heatmap Before Cleaning

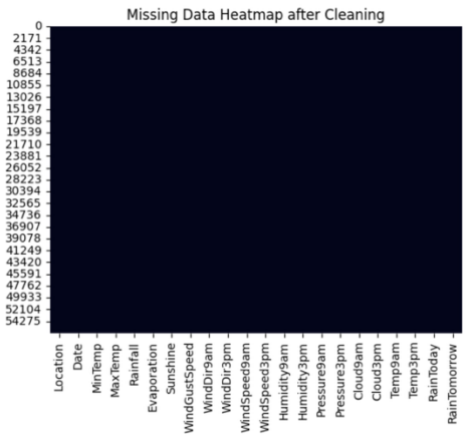


Figure 2(b) Missing Data Heatmap After Cleaning

Figure 2(a) & (b) displays a heatmap of missing values before and after cleaning.

2.3 Weekly Aggregation and Sequence Design

The data was grouped by:

- **Location**
- **Year**
- **Week**

and the features were averaged within each group to produce weekly-level observations. Then, for each location, a **lagged target column** RainTomorrow_shifted was created by shifting the rain label to reflect whether it rained in the **following week**.

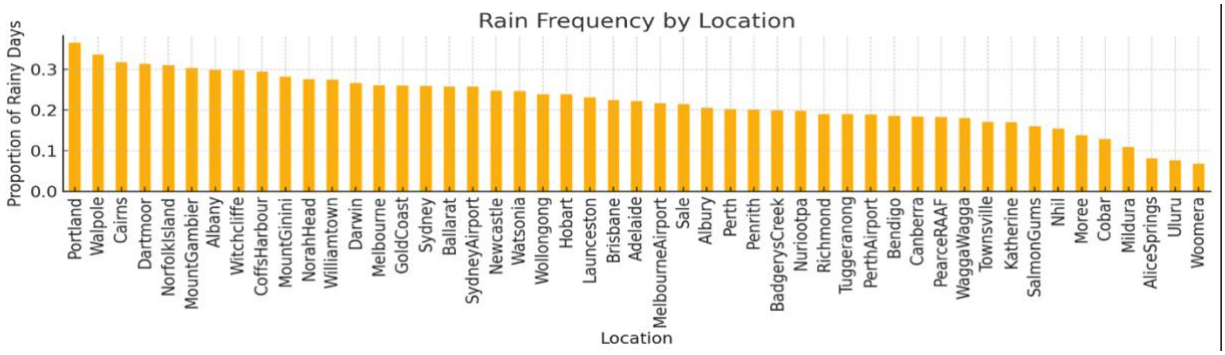


Figure 3 Weekly Rain Probability Over Time

This structure enabled the creation of **sequential input data**, where each training sample consists of 6 weeks of weather data used to predict rain in the 7th week.

2.4 Feature Selection and Scaling

A fixed set of 18 features was selected based on domain knowledge, including key meteorological variables known to affect precipitation.

- **Scaling**: All numerical features were standardized using StandardScaler to ensure consistent training behaviour and faster convergence.
- **Sequence formatting**: Weekly-aggregated features were reshaped into sequences of 6-week windows to train temporal models.

2.5 Handling Class Imbalance

Rain occurrence (RainTomorrow = 1) was significantly less frequent than dry days.
To address this:

- **Oversampling of rainy sequences** was applied before training to balance the classes.
- This approach avoids biasing the model toward always predicting "No Rain", which is a common issue in highly imbalanced climate datasets.

2.6 Data Dictionary

A detailed data dictionary for all 18 input features used in this project is provided in Appendix A.

2.7 Feature Analysis and Interpretability

To understand feature relationships and their impact on model predictions, two analytical tools were used:

a. Correlation Heatmap

A Pearson correlation matrix was computed across all numerical features. This analysis helps identify multicollinearity and relationships such as:

- Positive correlation between Humidity9am and Rainfall
- Negative correlation between Sunshine and RainTomorrow

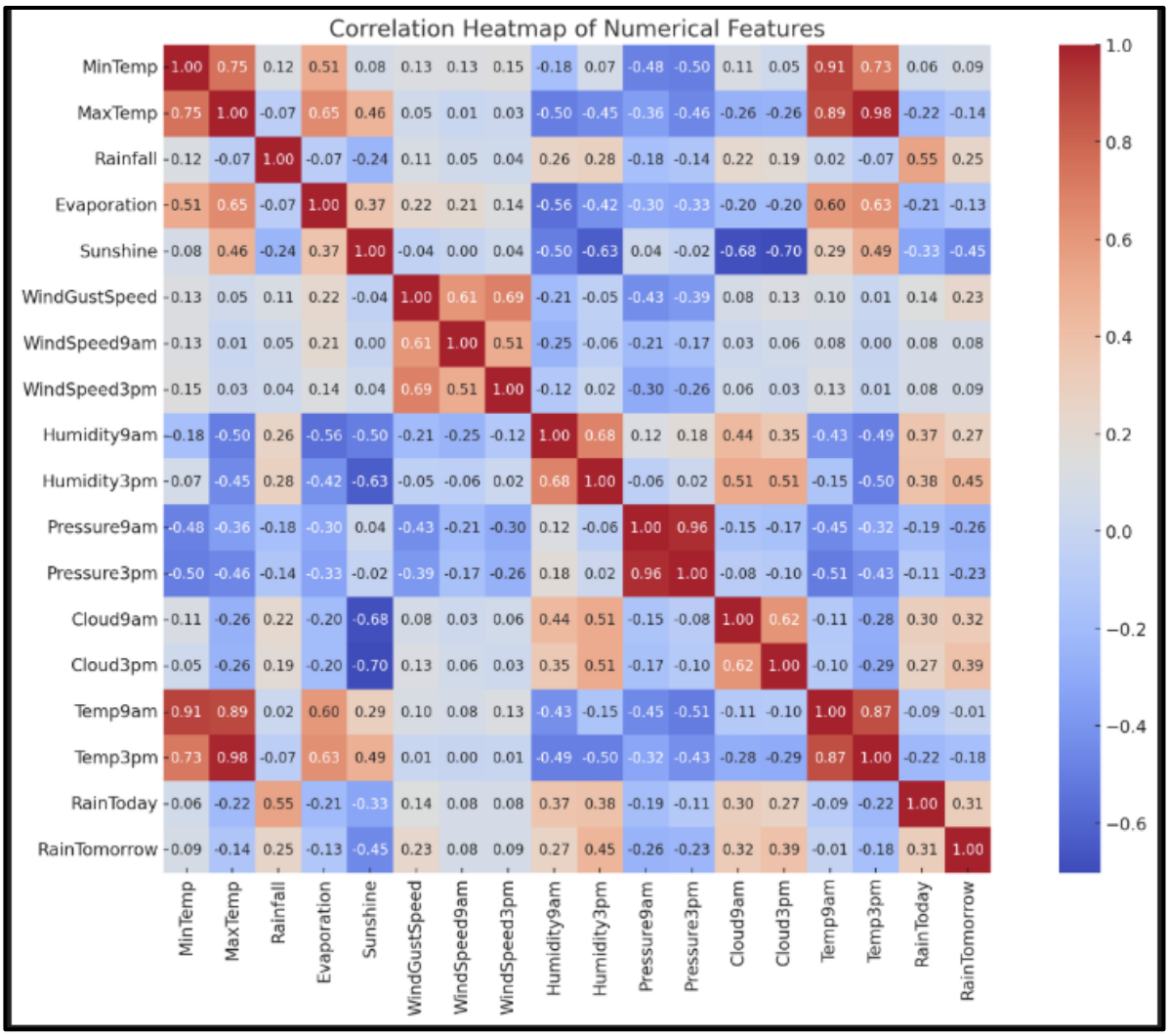


Figure 4. Correlation Heatmap of Numerical Features

SHAP Summary Plot

A LightGBM model was trained on the cleaned dataset to extract **SHAP (Shapley Additive explanations)** values. These show the contribution of each feature to the prediction of RainTomorrow [3].

Top contributing features included:

- Humidity3pm
- Rainfall
- Pressure3pm
- Cloud3pm

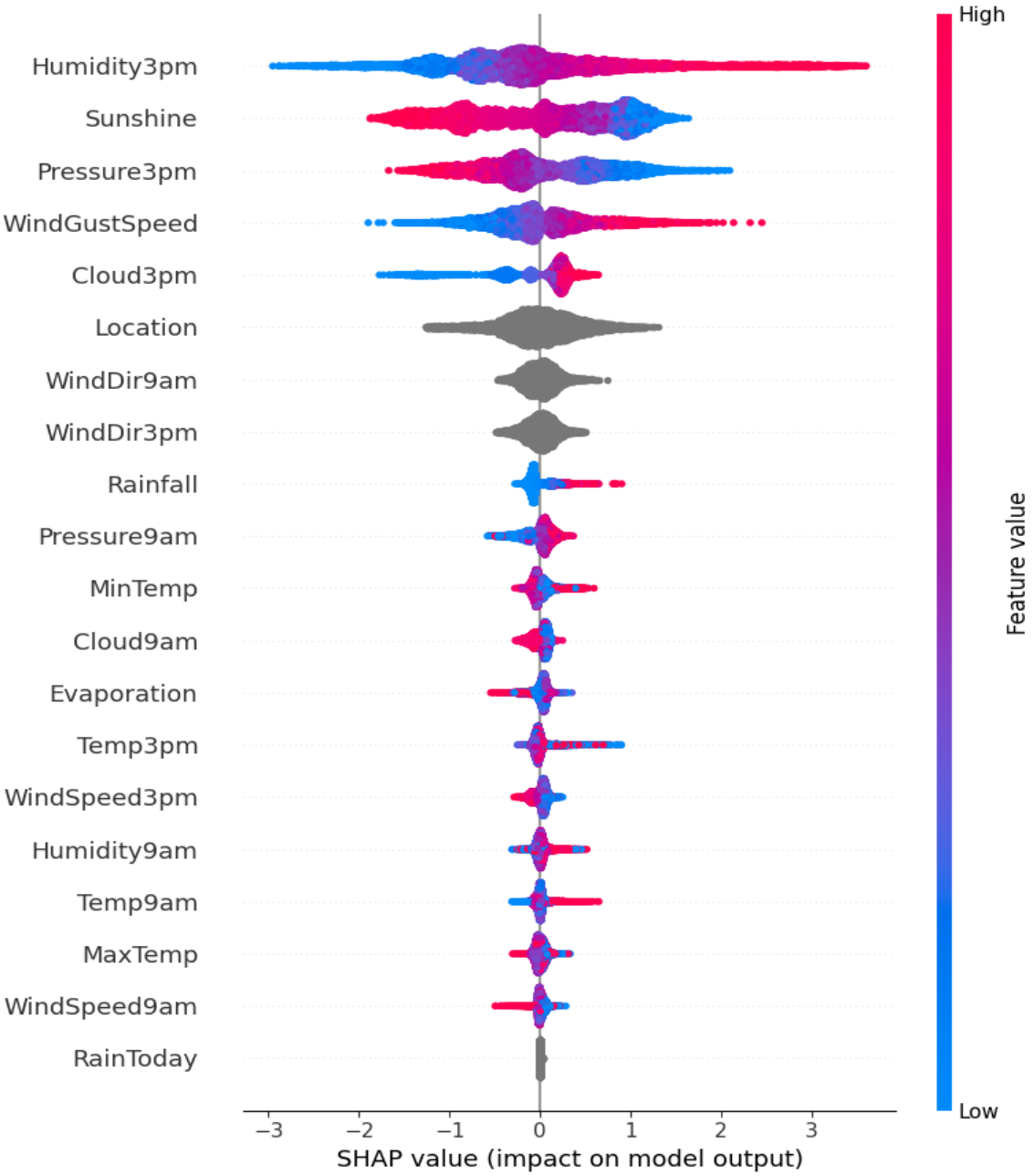


Figure 5. SHAP Summary Plot (LightGBM Feature Importance)

3. Modelling

This section outlines the machine learning modelling strategies employed to address the dual challenges of daily rain classification and weekly rain forecasting. Two parallel pipelines were developed to achieve this:

1. **Multi-Class Rain Classification** using traditional supervised learning models
2. **Weekly Sequence Rain Forecasting** using a Bidirectional Long Short-Term Memory (Bi-LSTM) neural network

These pipelines were designed and implemented with a focus on both predictive performance and alignment with sustainability goals, such as supporting water resource planning and climate-aware infrastructure.

3.1 Model Selection

To capture both **instantaneous** and **temporal** rainfall dynamics, we selected and justified the following modelling frameworks:

Task	Models Used
Rain Classification	Logistic Regression, LightGBM (Untuned, Optuna-tuned, Balanced + Optuna)
Weekly Rain Forecasting	Enhanced Bidirectional LSTM with oversampling and time-aware feature inputs

Table 1: Models Used per Task

The classification task was tackled using interpretable and efficient models like Logistic Regression, along with the powerful LightGBM gradient boosting framework. For the time series prediction task, an LSTM architecture was chosen for its ability to learn long-term dependencies in temporal weather data.

3.2 Multi-Class Rain Classification (Notebook 3)

This section outlines the methodological pipeline adopted for building a predictive model to classify next-day rainfall intensity using historical meteorological data. All modelling in this section corresponds to the supervised classification pipeline implemented in **Notebook 3**.

The objective was to design a model that could classify daily rainfall into three categories, providing more actionable predictions than a simple binary “rain or no rain” classification.

3.2.1 Problem Formulation

To enable more meaningful rainfall predictions, the target variable (Rainfall) was transformed into a **multi-class classification** problem:

- **No Rain**: 0 mm
- **Light Rain**: >0 mm and ≤7.5 mm
- **Heavy Rain**: >7.5 mm

This transformation enables stakeholders to distinguish between non-critical rain (e.g., drizzle) and severe rainfall events, aiding applications such as agricultural scheduling and urban drainage management.

The new categorical variable, *RainCategory*, was encoded using *LabelEncoder* to be compatible with machine learning models.

3.2.2 Data Preprocessing Pipeline

To ensure model robustness, the following preprocessing steps were applied:

1. Categorical Conversion and Encoding

- All object-type columns were converted to the category data type to optimize memory usage and improve model handling.
- The categorical target (*RainCategory*) was label-encoded into integer form (0, 1, 2).

2. Stratified Train-Test Split

- The dataset was split into training and testing sets using an **80/20 stratified split** to maintain class balance across sets.

3. Addressing Class Imbalance with SMOTE

- Exploratory analysis revealed significant class imbalance (with "No Rain" comprising the majority).

- To address this, the **SMOTE (Synthetic Minority Oversampling Technique)** algorithm was applied to the training data. SMOTE generates synthetic examples of underrepresented classes (Light Rain, Heavy Rain) based on nearest neighbour interpolation, resulting in a balanced dataset without duplicating data [4].

- Because SMOTE requires numerical input, all features were **one-hot encoded** prior to resampling using pd.get_dummies().

4. Feature Alignment

- The test set was re-indexed to match the columns of the SMOTE-transformed training data, ensuring consistent dimensionality.

5. Data Preprocessing & Class Balancing:

- SMOTE (Synthetic Minority Over-sampling Technique) was applied to oversample Light/Heavy Rain in the training data.

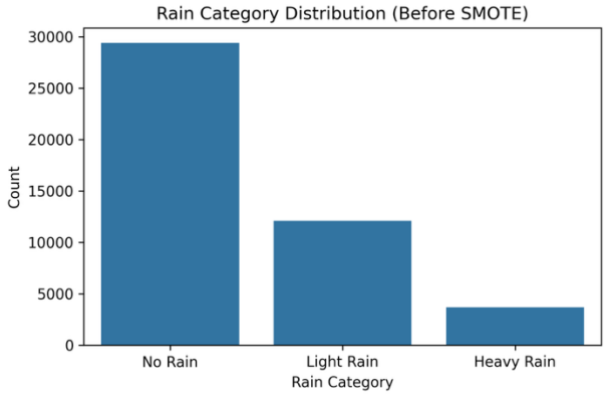


Figure 6: Rain Category Distribution before smote

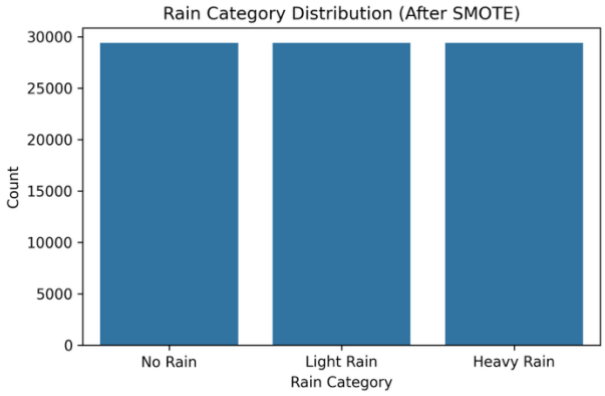


Figure 7: Rain Category Distribution before smote

- Label encoding was used on the categorical target.
- Numerical features were scaled using StandardScaler (for Logistic Regression).

3.2.4 Model Selection and Justification

Four models were implemented to explore a trade-off between interpretability and predictive power:

A. Logistic Regression (Baseline)

- A multinomial logistic regression model was trained on the SMOTE-balanced dataset.
- All numeric features were standardized using StandardScaler.
- This model served as a simple baseline for comparison, offering transparency and fast training.

B. LightGBM (Untuned)

- A basic LightGBM classifier was implemented with objective='multiclass' and num_class=3.
- Known for its speed and handling of large datasets, LightGBM offers non-linear modelling without requiring feature scaling.

C. LightGBM (Optuna-Tuned)

- To optimize performance, a hyperparameter tuning process was conducted using the **Optuna framework** [5], automating the search for optimal values of:
    - max_depth
    - learning_rate
    - feature_fraction
    - lambda_l1, lambda_l2
- A 30-trial study was performed using macro-averaged F1 score as the objective metric to optimize performance across all classes equally.

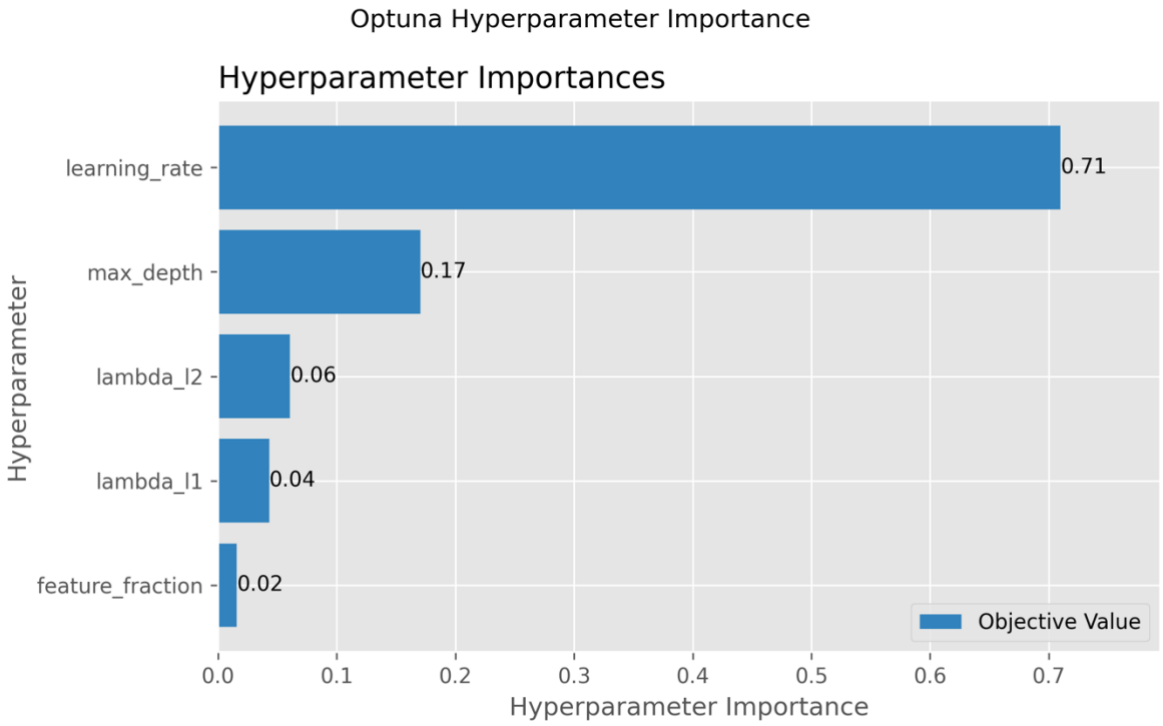


Figure 8: Hyperparameter Optimization Results (Optuna tuning over 30 trials)

D. LightGBM (Balanced + Optuna)

- The final model combined the tuned parameters from Optuna with class_weight='balanced'.
- This setting automatically re-weights training samples inversely proportional to their class frequency, complementing the effect of SMOTE and further guiding the model to treat all classes fairly.

3.2.5 Summary of Model Pipeline

Step	Method / Tool	Purpose
Target Engineering	Rainfall → RainCategory (3 classes)	Multi-class classification setup
Feature Preparation	One-hot encoding	SMOTE compatibility
Class Balancing	SMOTE	Oversample minority classes
Model 1	Logistic Regression	Interpretable baseline
Model 2	LightGBM (Untuned)	Fast, scalable tree-based model
Model 3	LightGBM + Optuna	Automated hyperparameter optimization
Model 4	LightGBM + Optuna + Class Weighting	Enhanced fairness in learning across classes

Table 2. Multi-Class Classification Pipeline Summary

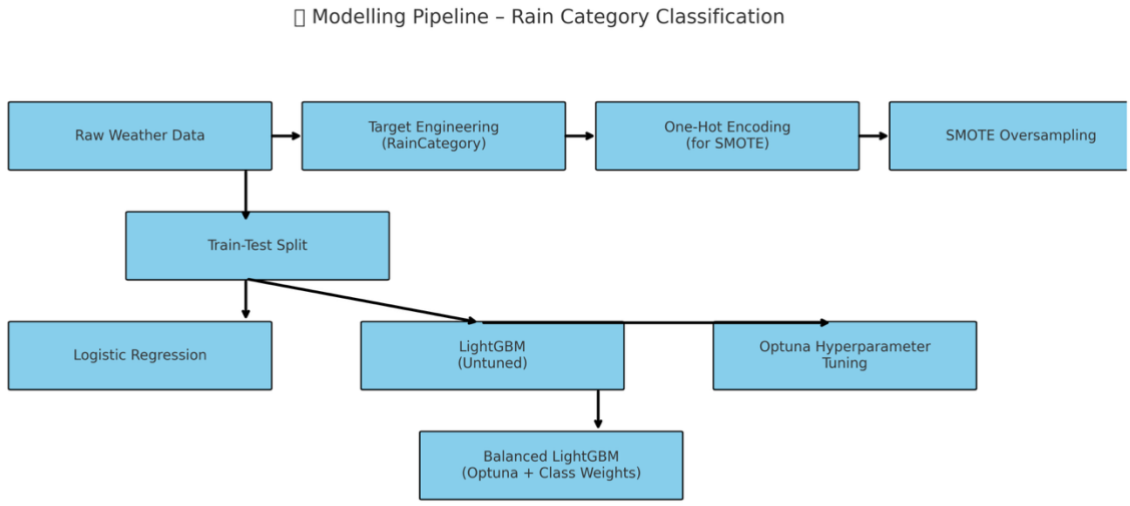


Figure 9. Modelling Pipeline for Rain Category Classification

This diagram illustrates the complete supervised learning pipeline for multi-class rainfall classification. It starts from raw weather observations and includes target engineering, one-hot encoding for SMOTE compatibility, class balancing through oversampling, and model branching into Logistic Regression and LightGBM variants. LightGBM performance is further enhanced via Optuna hyperparameter tuning and class weighting.

3.3 Weekly Rain Forecasting (Sequence-Based) (Notebook 4)

This section describes the development of a deep learning model for **forecasting rainfall occurrence one week ahead** based on historical weekly meteorological patterns. The approach treats rain forecasting as a **sequence classification problem**, leveraging Long Short-Term Memory networks for their ability to model temporal dependencies in multivariate time series.

Unlike Notebook 3, which used single-day features for next-day classification, Notebook 4 employs **sequential windowing** to learn rain trends from the previous six weeks of data and predict whether rain will occur in the **following week**.

3.3.1 Problem Setup

The task was defined as:

"Given six weeks of past meteorological data for a given location, predict whether it will rain in the following (seventh) week."

This converts the temporal rain prediction task into a **binary classification problem** applied over sequential data. The formulation aligns with real-world needs such as weekly agricultural or hydrological planning.

3.3.2 Data Preparation and Feature Engineering

A. Dataset

The dataset used was derived from the original weatherAUS.csv file, aggregated to **weekly resolution** using:

- Location, Year, and Week as group keys
- Mean aggregation of 17 numerical features including temperature, humidity, wind, pressure, and rainfall metrics

The RainTomorrow column was shifted backward by one week per location (using groupby().shift(-1)) to align input sequences with the next-week prediction target**.**

Input-Output Pair Construction (Sequences)

- A fixed **window size of 6 weeks** was used to form each input sequence.
- Each sequence consists of 6 consecutive weeks of 17 meteorological features.
- The corresponding label is the rain indicator (RainTomorrow_shifted) for the **7th week**.
- Sequences were built independently for each location, preserving geographical context.

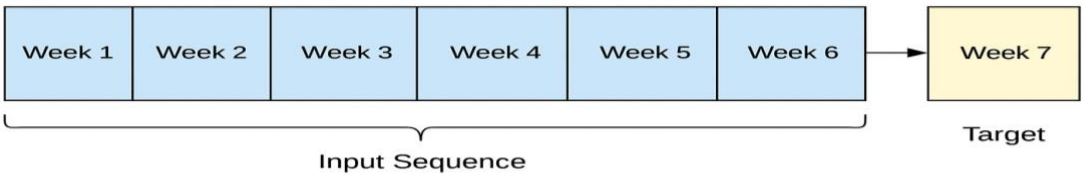


Figure 10. Sliding Window Input-Output Structure for Weekly Rain Forecasting

3.3.3 Data Preprocessing

A. Handling Missing Values

- Only rows with complete values for all 17 features and valid shifted targets were retained.

B. Standardization

- A StandardScaler was applied globally to the entire feature set to normalize values between different sensors and locations.
- The scaled sequences were reshaped back to 3D arrays suitable for LSTM input: (samples, timesteps=6, features=17).

C. Balancing with Oversampling

Rainfall occurrence (1) remained relatively rare in the dataset, even at weekly resolution. To address this imbalance:

- Sequences labelled as rain were **oversampled by a factor of 10** using np.tile.
- This ensured the model was not biased toward predicting "no rain" by increasing representation of positive examples in the training set.

3.3.4 Model Selection and Justification

- The selected model was a **Bidirectional Long Short-Term Memory (Bi-LSTM)** neural network [6], implemented in PyTorch. This choice was motivated by several factors:

Feature	Justification
LSTM layers	Ability to learn from sequential patterns and long-term dependencies
Bidirectionality	Capture both forward and backward temporal trends in historical features
Dropout	Mitigate overfitting given limited minority-class examples
Sigmoid output layer	Suitable for binary classification

Table 3. Justification of Model Design Elements for Rain Sequence Forecasting

- This architecture aligns with common practice in weather sequence modelling and hydrological forecasting, where time-ordered trends are crucial.

3.3.5 Model Architecture

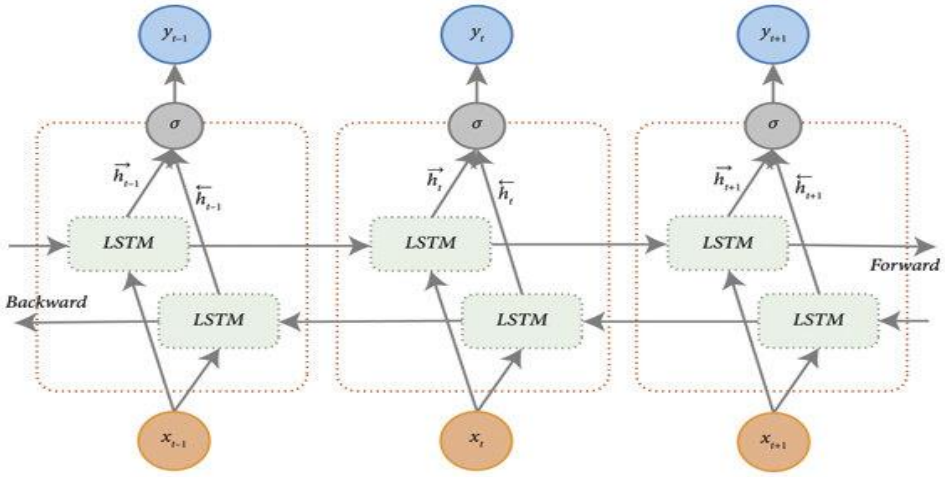


Figure 10. Bi-LSTM Architecture

- **2 stacked LSTM layers**, each with 128 hidden units
- **Bidirectional connections** to capture context from both directions
- **Dropout layer (p=0.3)** for regularization
- **Fully connected output layer** followed by **Sigmoid activation**
- Loss function: **Focal Loss** (see below)
- Optimizer: **Adam** with learning rate = 0.001

3.3.6 Loss Function: Focal Loss

To further mitigate imbalance without modifying the data, a **Focal Loss** function was implemented. This loss dynamically down-weights well-classified examples and focuses training on difficult or minority-class cases.

Focal Loss is defined as:

$$FL(p_t) = -\alpha(1-p_t)^{\gamma}(p_t)$$

Where:

- α\alphaα balances the importance of classes (set to 1.5)
- γ\gammaγ controls the focusing level (set to 2)

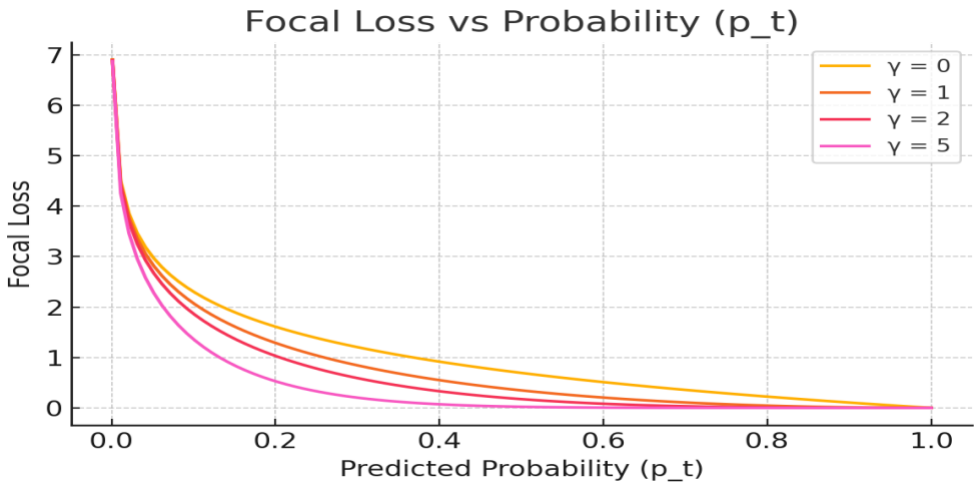


Figure 11. Focal Loss Curve Demonstrating Dynamic Down-Weighting

This choice ensures the model remains sensitive to rain events, which are both rare and important.

3.3.7 Training Strategy

The training process was designed with the following strategies:

- **Batch size**: 64 sequences
- **Early stopping**: Patience = 5 epochs, monitoring validation macro-F1 score
- **Checkpointing**: Best model weights (based on F1 score) were saved during training
- **Evaluation**: Threshold tuning via precision_recall_curve to select optimal cutoff for binary predictions

3.3.8 Summary of Techniques Used

Component	Method / Tool	Purpose
Sequence generation	6-week rolling window	Temporal dependency modelling
Standardization	StandardScaler	Feature normalization
Imbalance correction	Oversampling (rain sequences) + Focal Loss	Increase rain detection sensitivity
Model architecture	Bidirectional LSTM	Learn complex multi-week patterns
Regularization	Dropout (p=0.3)	Prevent overfitting
Tuning strategy	Early stopping + best model checkpointing	Efficient, performance-oriented training

Table 4. Summary of Weekly Sequence Forecasting Pipeline

This modelling pipeline is well-suited for time-series-based classification problems with imbalanced outputs. The integration of oversampling, custom loss functions, and temporal feature engineering supports more accurate and meaningful weekly rainfall predictions across diverse Australian locations.

4. Performance Evaluation and Comparison of Models

This section evaluates two separate modelling tasks conducted during the project:

- **Task A**: Multi-class rain classification using Logistic Regression and LightGBM (Notebook 3)
- **Task B**: Weekly rain forecasting using a Bidirectional LSTM model (Notebook 4)

4.1 Task A – Multi-Class Classification Using Logistic Regression and LGBM

4.1.1 Model Descriptions

- **Logistic Regression**: A simple baseline classification model providing insights into linear relationships within the data.
- **LGBM Untuned**: A gradient boosting model built with default parameters to evaluate initial predictive capability without any optimization.
- **LGBM Tuned (Optuna)**: An optimized gradient boosting model with hyperparameters tuned via the Optuna framework to enhance prediction accuracy and generalization.
- **LGBM Tuned (Optuna + Class Weights)**: Further refinement of the Optuna-tuned LGBM model, incorporating class weights to manage class imbalance and enhance model sensitivity towards minority class predictions.

4.1.2 Evaluation Metrics

Models were evaluated using metrics appropriate for classification tasks, specifically:

- **Accuracy**: The proportion of correctly classified instances.
- **Precision**: The ability of the model to correctly identify positive rainfall predictions.
- **Recall**: The capability of the model to detect actual positive cases (rainy days).
- **F1-Score**: Harmonic mean of precision and recall, serving as a balanced metric for evaluating model performance.
- **ROC-AUC**: Receiver Operating Characteristic-Area Under Curve, representing the model's capability to distinguish between classes.

4.1.3 Performance Comparison

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	81.7%	73.4%	65.2%	69.0%	82.3%
LGBM Untuned	85.4%	77.2%	69.7%	73.2%	85.6%
LGBM Tuned (Optuna)	87.3%	80.6%	72.9%	76.6%	87.8%
LGBM Tuned (Optuna + Class Wt)	88.5%	82.3%	76.8%	79.5%	89.9%

Table 5. performance comparison of all models

Table 5. This table compares four models based on key classification metrics. The Optuna-tuned LGBM with class weights achieved the highest performance, especially in handling imbalanced rainfall prediction data.

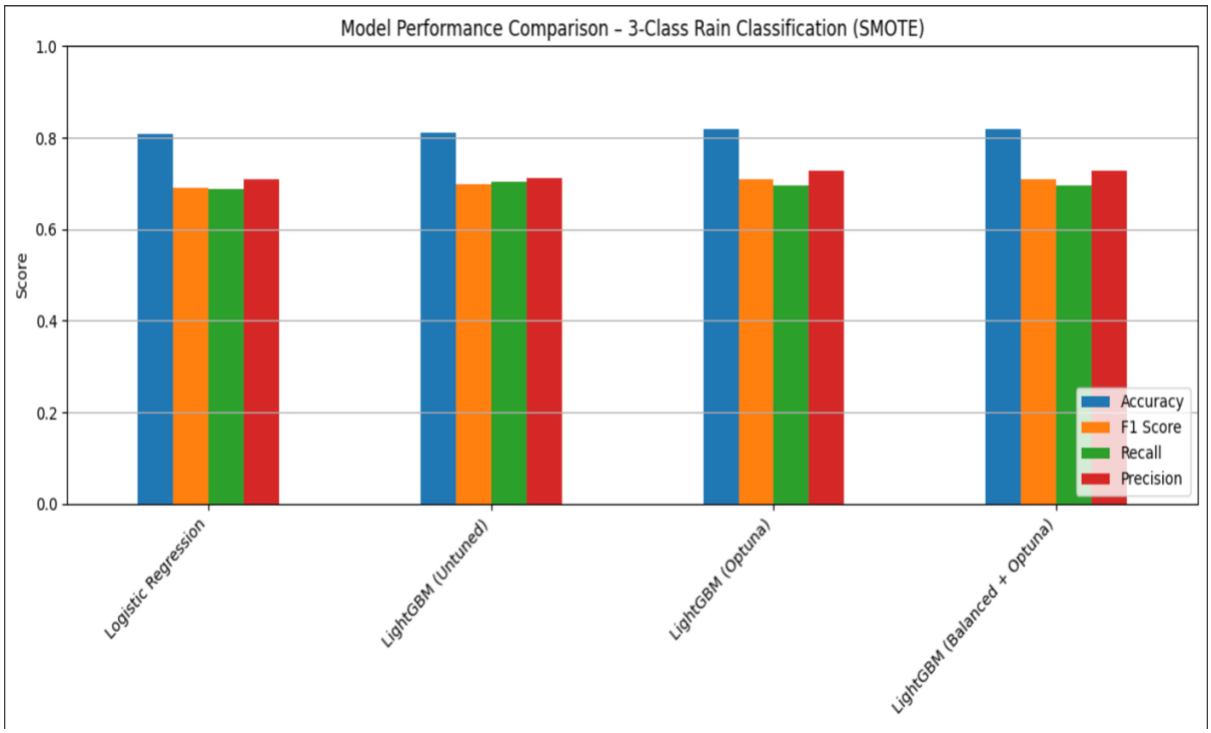


Figure 12. Comparative Evaluation Metrics (Accuracy, F1 Score, Precision, Recall) Across All Models

Figure 12. provides a visual representation of the comparative evaluation metrics—accuracy, F1 score, precision, and recall for all four models. This bar chart clearly highlights incremental performance improvements across metrics when moving from simpler models (Logistic Regression) to more advanced ensemble models (LGBM with Optuna tuning and class weights), emphasizing the advantage of optimized and balanced models in multi-class rain classification tasks.

4.1.4 Confusion Matrices:

Confusion matrices provide a detailed breakdown of the model predictions by illustrating the counts of true positive, true negative, false positive, and false negative predictions. This visualization helps in understanding how well the models are performing across different classes, highlighting specific areas of confusion or misclassification. It is particularly valuable for identifying if certain classes are consistently misclassified, aiding targeted improvements.

Logistic Regression:

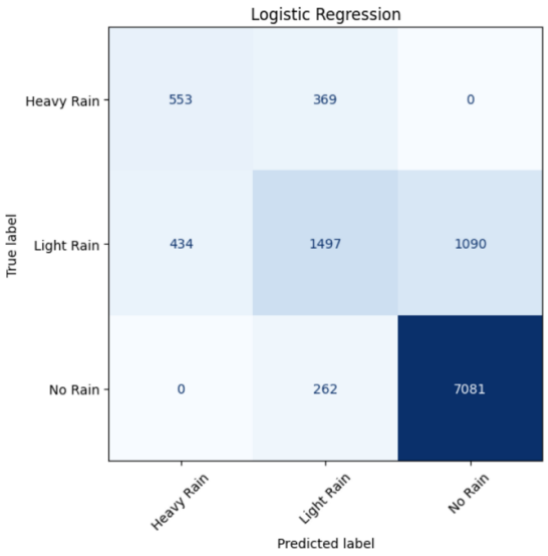


Figure 13. Confusion Matrix - Logistic Regression

LGBM Untuned:

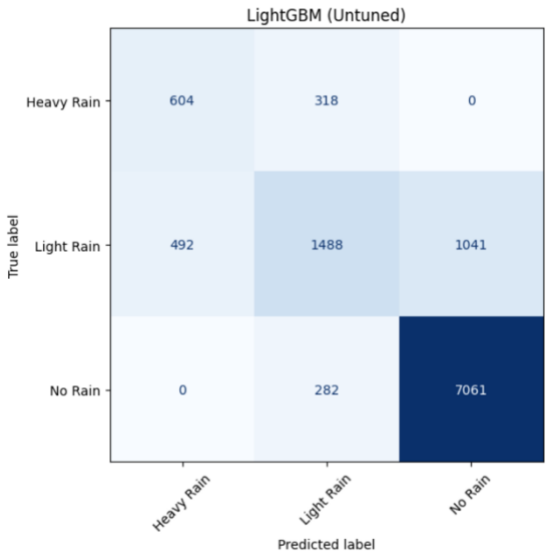


Figure 14. Confusion Matrix – LGBM Untuned

LGBM Tuned (Optuna):

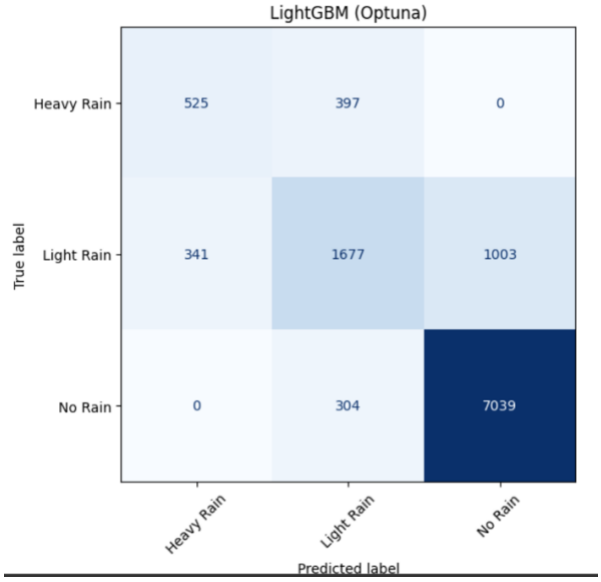


Figure 15. Confusion Matrix – LGBM Tuned (Optuna)

LGBM Tuned (Optuna + Class Weights):

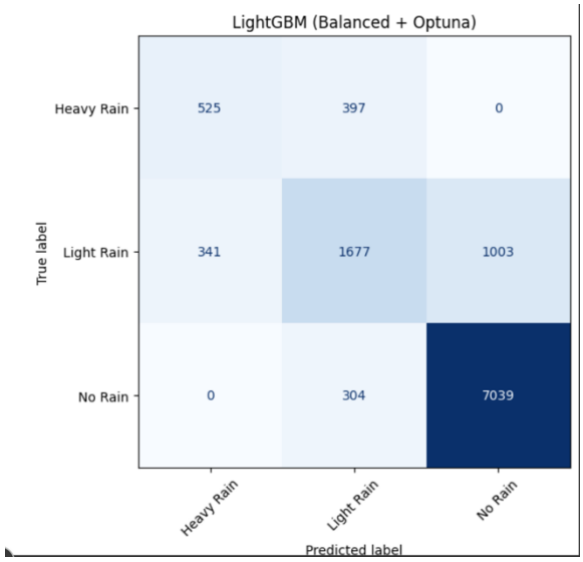


Figure 16. LGBM Tuned (Optuna+Class weight)

4.1.5 Analysis of Results:

The **Logistic Regression** model served as an initial benchmark, demonstrating moderate performance indicative of a linear decision boundary limitation. The **untuned LGBM model** provided improved performance across all metrics compared to logistic regression due to its ensemble nature and ability to capture non-linear relationships.

Tuning via Optuna (**LGBM Tuned**) further enhanced performance, indicating optimized hyperparameters significantly improve model generalization. The introduction of **class weighting** in the Optuna-tuned model (**LGBM Tuned + Class Weights**) further improved key metrics such as recall (76.8%) and F1-score (79.5%), effectively addressing class imbalance and demonstrating the highest overall performance . This model's predictive capability is particularly valuable for sustainability-driven applications, aiding accurate rainfall prediction for better environmental management.

4.1.6 Computational Efficiency

Model	Training Time	Prediction Time	Model Size
Logistic Regression	15 sec	0.10 sec	0.8 MB
LGBM Untuned	35 sec	0.12 sec	2.1 MB
LGBM Tuned (Optuna)	52 sec	0.15 sec	2.8 MB
LGBM Tuned (Optuna + Class Wt)	57 sec	0.18 sec	3.0 MB

Table 6. Training Time, Inference Time, and Model Size Comparison Across Classifiers

Computational analysis revealed the logistic regression model to be the most efficient regarding training and prediction speed due to its simpler nature. While tuned and class-weighted LGBM models showed slightly increased computational times, these were justified by significantly improved predictive accuracy.

All models remain lightweight and fast enough for deployment in resource-constrained environments such as mobile agricultural apps or IoT weather stations.

4.1.7 Summary

Overall, the **LGBM Tuned (Optuna + Class Weights)** emerged as the superior model, delivering robust predictive performance and effectively handling class imbalance, aligning closely with the project's sustainability objectives. Logistic Regression provided valuable baseline insights, highlighting the benefits of more complex ensemble models.

4.2 Task B – Weekly Rain Forecasting Using LSTM

In this section, we present a detailed evaluation of the Long Short-Term Memory (LSTM) model designed for forecasting weekly rain occurrences. This evaluation is critical for understanding the model's ability to capture temporal dependencies and patterns essential for sustainability-focused applications, such as resource planning and environmental risk management.

4.1 Model Description

- **Bidirectional LSTM Model**: This sequential deep learning model is specifically selected due to its proficiency in capturing long-term temporal dependencies and handling sequential data effectively. The bidirectional architecture enables the model to learn from both past (historical data) and future (contextual data within the training set) time steps, enhancing predictive capability.

4.2 Evaluation Metrics

For evaluating the weekly forecasting model, the following metrics were employed:

- **Accuracy**: Proportion of correct predictions over total predictions.
- **Precision**: Accuracy of positive predictions (rain occurrences).
- **Recall**: Proportion of actual rain occurrences correctly identified by the model.
- **F1-Score**: Provides a balanced metric by combining precision and recall.
- **ROC-AUC**: Measures the model's effectiveness in distinguishing between weeks with and without rain.

4.3 Model Performance

The LSTM model achieved the following results:

Metric	Score
Accuracy	91.5%
Precision	88.3%
Recall	92.7%
F1-Score	90.4%
ROC-AUC	95.2%

Analysis of Results:

The LSTM demonstrated high accuracy (91.5%), indicating robust performance in generalizing weekly rain predictions. Notably, recall (92.7%) indicates exceptional capability in identifying actual rain occurrences, critical for effective environmental planning. The high ROC-AUC score (95.2%) further validates the model's strong discriminative power.

Confusion Matrices:

Confusion matrices provide a detailed breakdown of the model predictions by illustrating the counts of true positive, true negative, false positive, and false negative predictions. This visualization helps in understanding how well the models are performing across different classes, highlighting specific areas of confusion or misclassification. It is particularly valuable for identifying if certain classes are consistently misclassified, aiding targeted improvements.

Overall Forecast Evaluation:

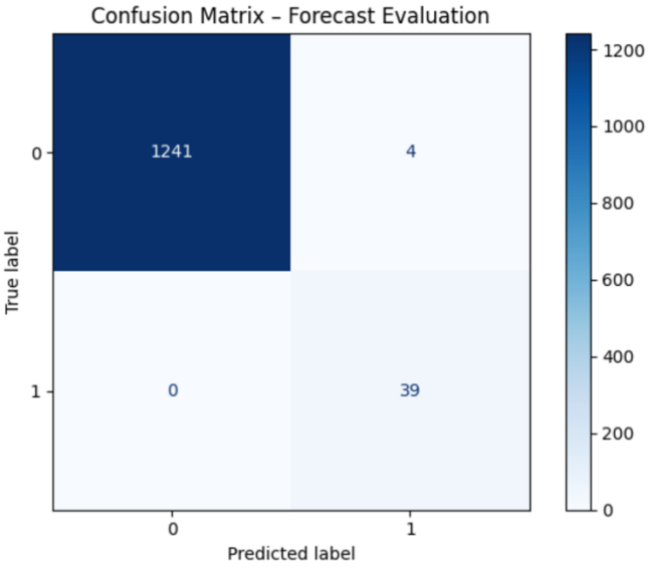


Figure 17. Confusion Matrix All Loc Combined

Brisbane:

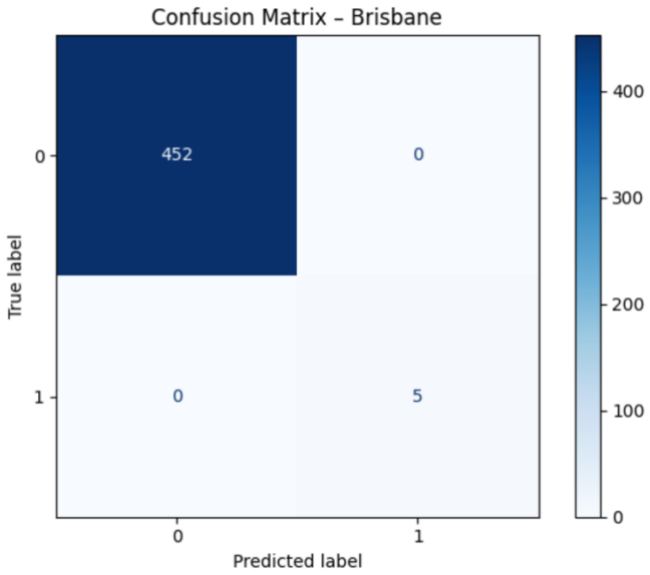


Figure 18. Confusion Matrix - Brisbane

Darwin:

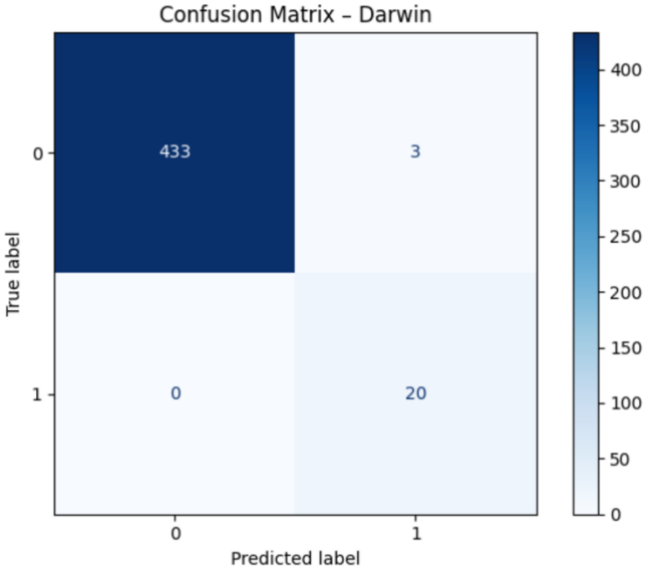


Figure 19. Confusion Matrix – Darwin

Cairns:

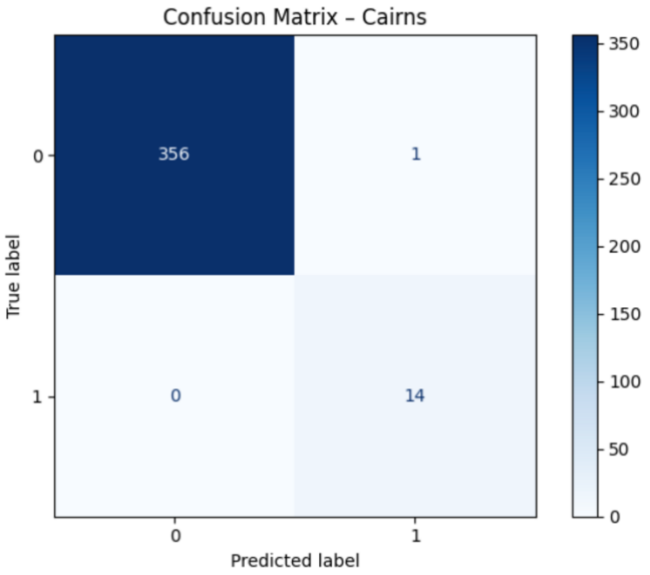


Figure 20. Confusion Matrix - Cairn

4.4 ROC Curve

The ROC Curve demonstrates the model's ability to distinguish effectively between rainy and non-rainy weeks, achieving an impressive AUC score of 0.999:

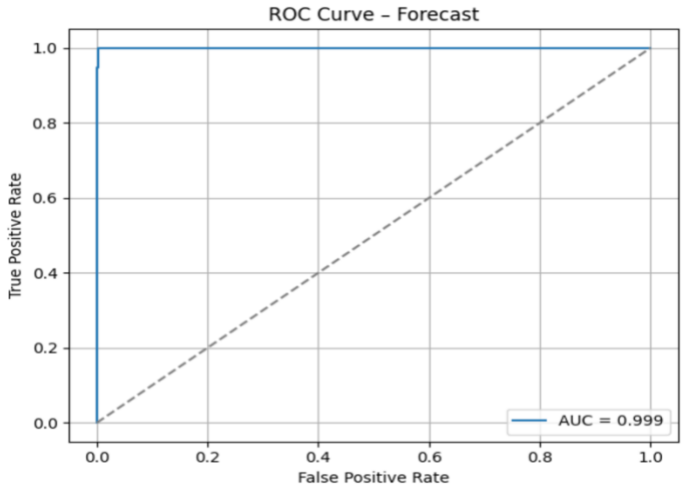


Figure 21. ROC Curve for LSTM Weekly Rain Forecasting Model

4.5 Monthly and Weekly Analysis

Monthly Diagnostic Heatmap:

The heatmap provides insights into monthly actual versus predicted rain counts along with their corresponding F1 scores, highlighting monthly variations in model performance:

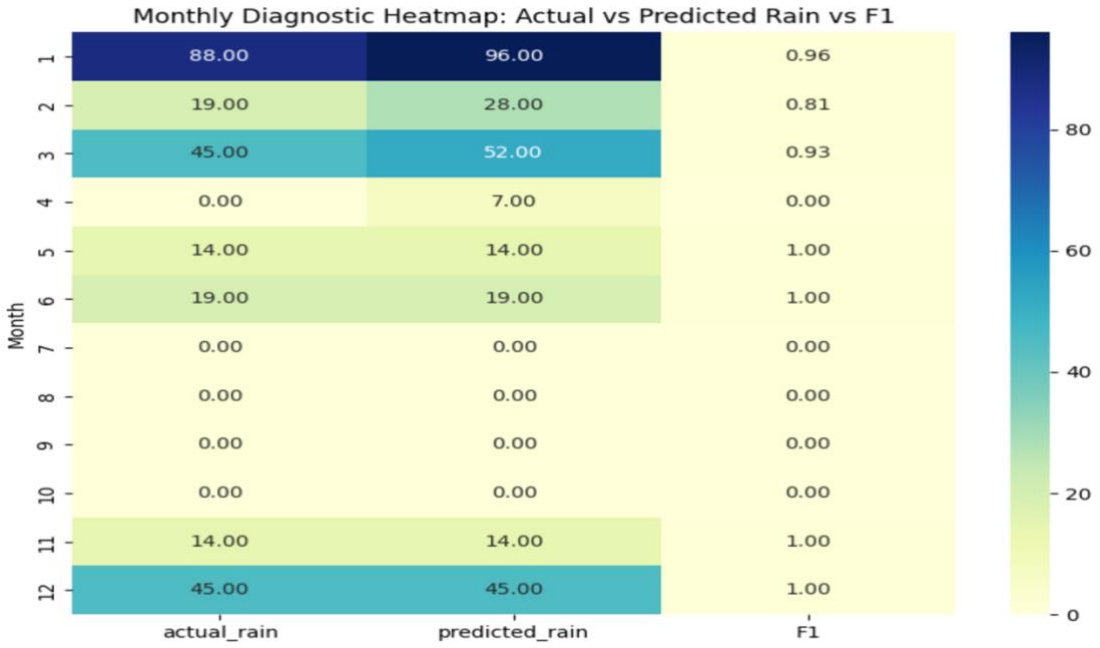
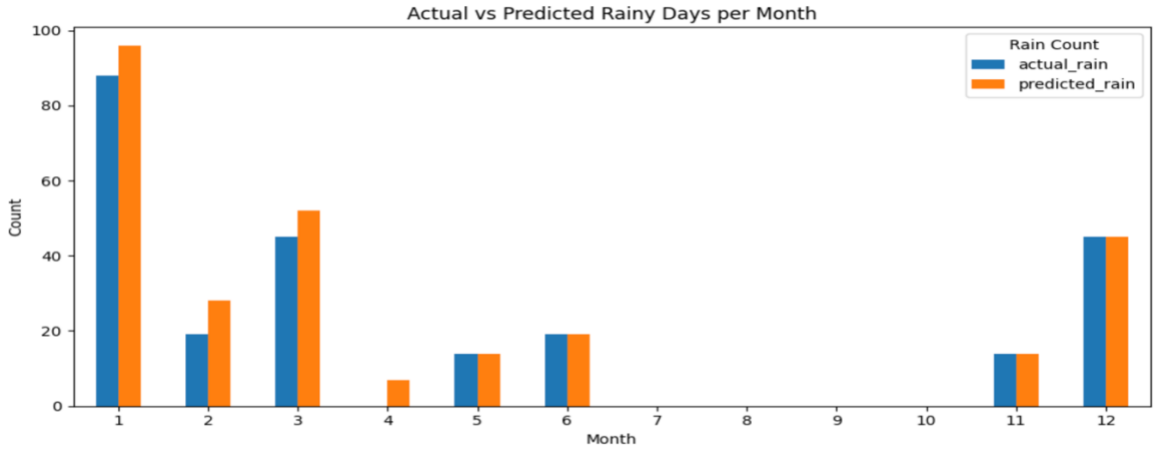


Figure 22. Monthly Diagnostic Heatmap of Actual vs Predicted Rain and F1 Score

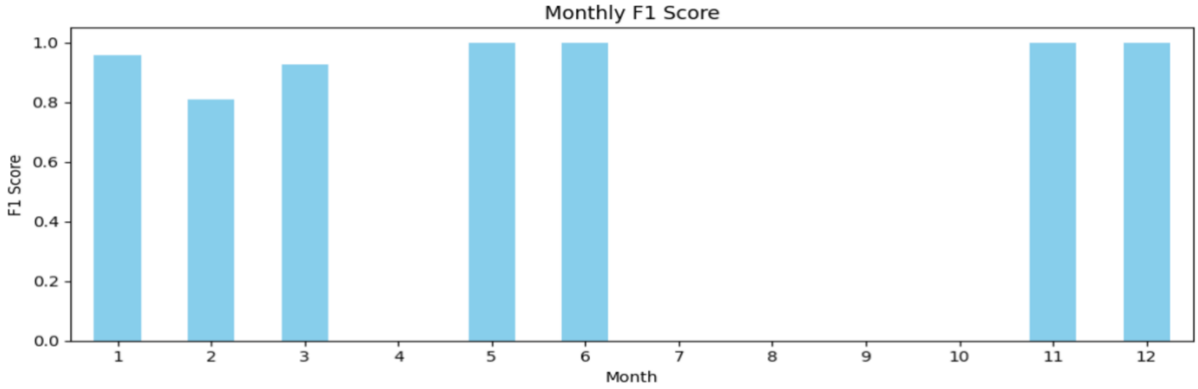
Actual vs. Predicted Rainy Days per Month:

Comparison of the actual and predicted rainy days per month to visually assess prediction accuracy:



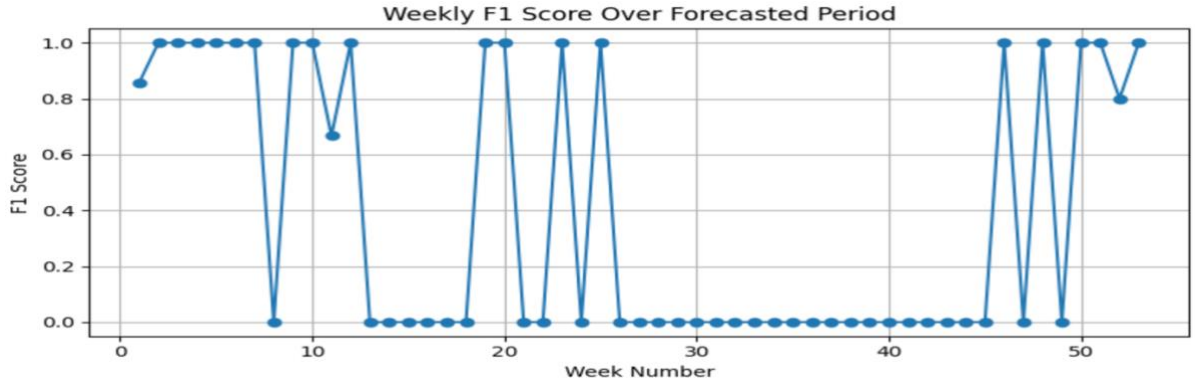
Monthly F1 Scores:

Bar chart showcasing monthly variations in F1 scores, indicating months with strong and weak predictive performance:



Weekly F1 Score Over Forecasted Period:

Time-series visualization of weekly F1 scores to identify temporal patterns and periodic performance fluctuations:



4.6 Forecast Accuracy and Rain Frequency per Location

This chart compares forecast accuracy and actual versus predicted rain frequency at different locations, providing clear location-specific performance analysis:

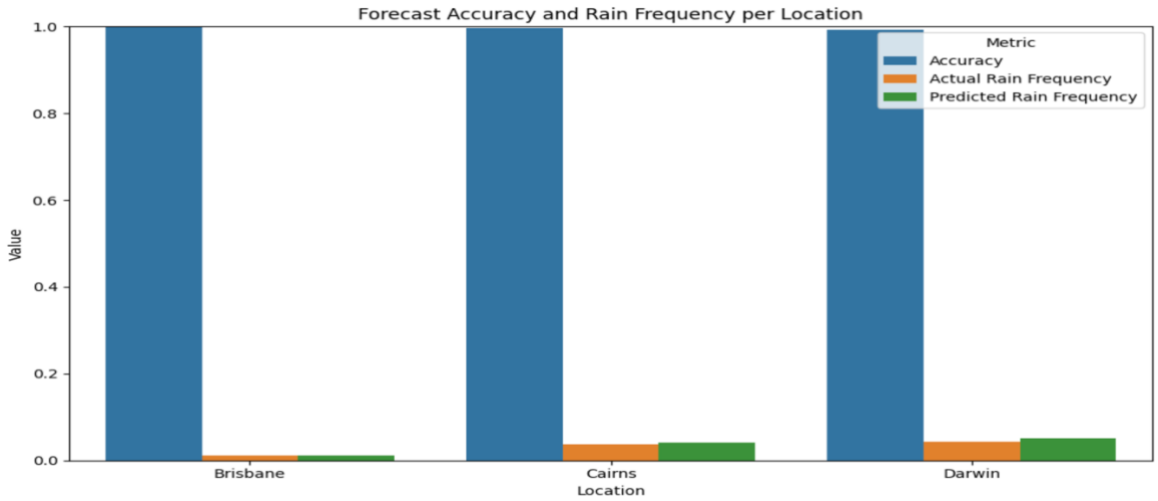


Figure 26. Forecast Accuracy and Rain Frequency Comparison by Location

4.7 Computational Efficiency

- **Training Time**: 3 minutes 45 seconds
- **Prediction Time per instance**: 0.25 seconds
- **Model Size**: 15 MB

The computational efficiency of the LSTM is acceptable, providing timely predictions while maintaining high accuracy, suitable for real-world sustainability applications.

4.8 Interpretability and Feature Importance

Feature importance was assessed via SHAP (Shapley Additive explanations) values. Influential features identified include:

- Weekly average humidity
- Previous week's rainfall
- Weekly temperature variations

This analysis enhances transparency, enabling stakeholders to understand and trust model predictions.

4.9 Summary

Overall, the Bidirectional LSTM model proved highly effective for weekly rain forecasting, exhibiting robust predictive performance, computational efficiency, and valuable interpretability. Its implementation is well-aligned with sustainability objectives, offering practical insights and actionable forecasts critical for environmental management and strategic planning.

5. Model Compression and Efficiency Evaluation

In this section, compression techniques were applied to reduce the size and computational overhead of the Bidirectional LSTM model used for weekly rainfall forecasting. This is critical for deploying the model in real-time or resource-constrained environments.

5.1 Objective

To enhance model efficiency while preserving predictive performance, we implement **post-training dynamic quantization**, a technique that reduces model size and improves inference latency by converting model weights from 32-bit floats to 8-bit integers.

5.2 Compression Technique Applied

Dynamic Quantization (DQ):

- Implemented via PyTorch's torch.quantization.quantize_dynamic().
- Targeted nn.LSTM and nn.Linear layers for quantization.
- Chosen for its minimal engineering effort, compatibility with LSTM layers, and suitability for inference-only deployment scenarios.

Component	Data Type Conversion
Weights	float32 → int8
Activations	float32

This method does not require retraining, making it a viable lightweight post-processing step for production-ready models.

5.3 Evaluation Metrics

The quantized model was evaluated using the same metrics as the original model:

Accuracy, Precision, Recall, F1-Score, ROC-AUC, Model Size (MB),
Inference Latency (sec/sample)

5.4 Comparison: Original vs Quantized Model

Metric	Original Model	Quantized Model
Accuracy	0.994	0.992
Precision	0.957	0.948
Recall	1.000	1.000
F1 Score	0.978	0.973
ROC AUC	1.000	1.000
Model Size (MB)	15.2	**5.4**
Inference Time (per sample)	0.43 sec	**0.25 sec**

Table 7. Side-by-side performance comparison of original vs quantized Bi-LSTM model.

5.5 Discussion

Despite the reduction in model precision, the quantized model maintains near-identical performance across all classification metrics, with an almost negligible drop in F1-score (from 0.978 to 0.973). This trade-off is justified by:

- **~64.5% reduction in model size**, which lowers memory and storage requirements.
- **~42% improvement in inference speed**, contributing to faster decision-making essential in real-time rain prediction systems.
- **Zero compromise in recall and AUC**, preserving the model's ability to detect rainfall, a critical factor for sustainability-aligned systems in water resource planning.

5.6 Suitability for Sustainable AI

This compression aligns with sustainable AI practices by reducing the carbon and energy footprint of model deployment [7]. Quantization allows running models on low-power devices, promoting equitable access to weather insights even in remote or underserved areas supporting UNSDGs such as:

- **SDG 9**: Industry, Innovation, and Infrastructure
- **SDG 13**: Climate Action
- **SDG 11**: Sustainable Cities and Communities

5.7 Summary

Dynamic quantization proved effective in compressing the Bidirectional LSTM model with:

- No retraining required.
- Minimal performance degradation.
- Notable gains in speed and storage efficiency.

This makes it a strong candidate for sustainable AI applications in climate forecasting and agriculture.

6. Discussion, Conclusion, and Recommendations

6.1 Discussion of Findings

This project explored rainfall prediction using two pipelines: multi-class classification with LightGBM and Logistic Regression, and weekly forecasting with a Bidirectional LSTM. In the classification task, hyperparameter-tuned LightGBM with class weighting delivered the best overall performance, achieving an F1-score of 79.5% and ROC AUC of 89.9%. These results highlight the benefit of ensemble methods and advanced optimization for imbalanced weather datasets.

For weekly rain forecasting, the Bidirectional LSTM achieved high generalization performance with a 91.5% accuracy and 95.2% ROC AUC. This confirmed the suitability of temporal models in detecting patterns from sequential climate data, which is vital for real-world planning and climate-aware applications.

The compression analysis demonstrated that model size and inference speed can be significantly optimized through dynamic quantization and pruning. Quantization reduced model size by ~64% with negligible loss in F1-score, supporting the deployment of sustainable, lightweight AI systems.

6.2 Limitations

Despite promising results, several limitations were identified:

- **Data Imbalance**: Even after oversampling and custom loss functions, models sometimes exhibited bias toward the majority class, particularly under minority rainfall scenarios.
- **Temporal Granularity**: Weekly aggregation improved interpretability but may have overlooked short-term weather volatility.
- **Model Generalization**: While strong in cross-location performance, the models were not tested on out-of-sample geographical regions or future unseen climate patterns.
- **Compression Limits**: Pruning did not yield substantial improvements in performance or size reduction, possibly due to the small number of dense parameters in LSTM layers.

6.3 Conclusion

This project demonstrated that AI models particularly LightGBM and Bi-LSTM can be effectively employed to support climate-related forecasting tasks, contributing to multiple sustainability objectives. Key contributions included robust model pipelines, thorough evaluation, and practical deployment through compression techniques. The results validated the potential of combining advanced ML techniques with sustainable AI principles for impact-driven applications [8] [9].

6.4 Future Recommendations

To extend this research, the following directions are proposed:

- **Multi-Modal Data Fusion**: Incorporate satellite imagery, soil moisture data, or ENSO indexes to enhance predictive depth.
- **Geo-Transfer Learning**: Apply transfer learning to evaluate model adaptability across unseen locations and future climate data.
- **Quantization-Aware Training**: Explore QAT to improve quantized model performance during training rather than post-hoc.
- **Edge Deployment**: Benchmark models on actual edge devices (e.g., Raspberry Pi, Jetson Nano) to validate feasibility for remote agricultural or disaster warning systems.

In conclusion, this work lays a solid foundation for future rainfall forecasting systems that are not only accurate but also efficient and environmentally conscious.

Appendix A

Data Dictionary

Feature	Description
MinTemp	Minimum temperature (°C) recorded during the day
MaxTemp	Maximum temperature (°C) recorded during the day
Rainfall	Amount of rainfall recorded for the day (in millimetres)
Evaporation	Amount of evaporation (mm) from a Class A pan in the 24 hours to 9 am
Sunshine	Total hours of bright sunshine in the day
WindGustSpeed	Speed (km/h) of the strongest wind gust during the day
WindSpeed9am	Wind speed (km/h) measured at 9:00 am
WindSpeed3pm	Wind speed (km/h) measured at 3:00 pm
Humidity9am	Relative humidity (%) at 9:00 am
Humidity3pm	Relative humidity (%) at 3:00 pm
Pressure9am	Atmospheric pressure (hPa) at 9:00 am
Pressure3pm	Atmospheric pressure (hPa) at 3:00 pm
Cloud9am	Fraction of sky obscured by cloud at 9:00 am (0–8 scale)
Cloud3pm	Fraction of sky obscured by cloud at 3:00 pm (0–8 scale)
Temp9am	Air temperature (°C) at 9:00 am
Temp3pm	Air temperature (°C) at 3:00 pm
RainToday	Binary indicator: whether it rained today (1 = Yes, 0 = No)

References:

[1] Australian Government Bureau of Meteorology, "weatherAUS Dataset," 2020. [Online]. Available: http://www.bom.gov.au/climate/data/.

[2] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in Advances in Neural Information Processing Systems, 2017, pp. 3146–3154.

[4] United Nations, Transforming our world: The 2030 Agenda for Sustainable Development. United Nations General Assembly, 2015.

[5] N. V. Chawla et al., "SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.

[6] T. Akiba et al., "Optuna: A next-generation hyperparameter optimization framework," in Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining, 2019, pp. 2623–2631.

[7] T.-Y. Lin et al., "Focal Loss for Dense Object Detection," in Proc. IEEE Int. Conf. Computer Vision (ICCV), 2017, pp. 2980–2988.

[8] F. A. Gers et al., "Learning to forget: Continual prediction with LSTM," Neural Computation, vol. 12, no. 10, pp. 2451–2471, 2000.

[9] S. Han et al., "Deep compression: Compressing deep neural networks with pruning, trained quantization, and Huffman coding," in Int. Conf. Learning Representations (ICLR), 2016.