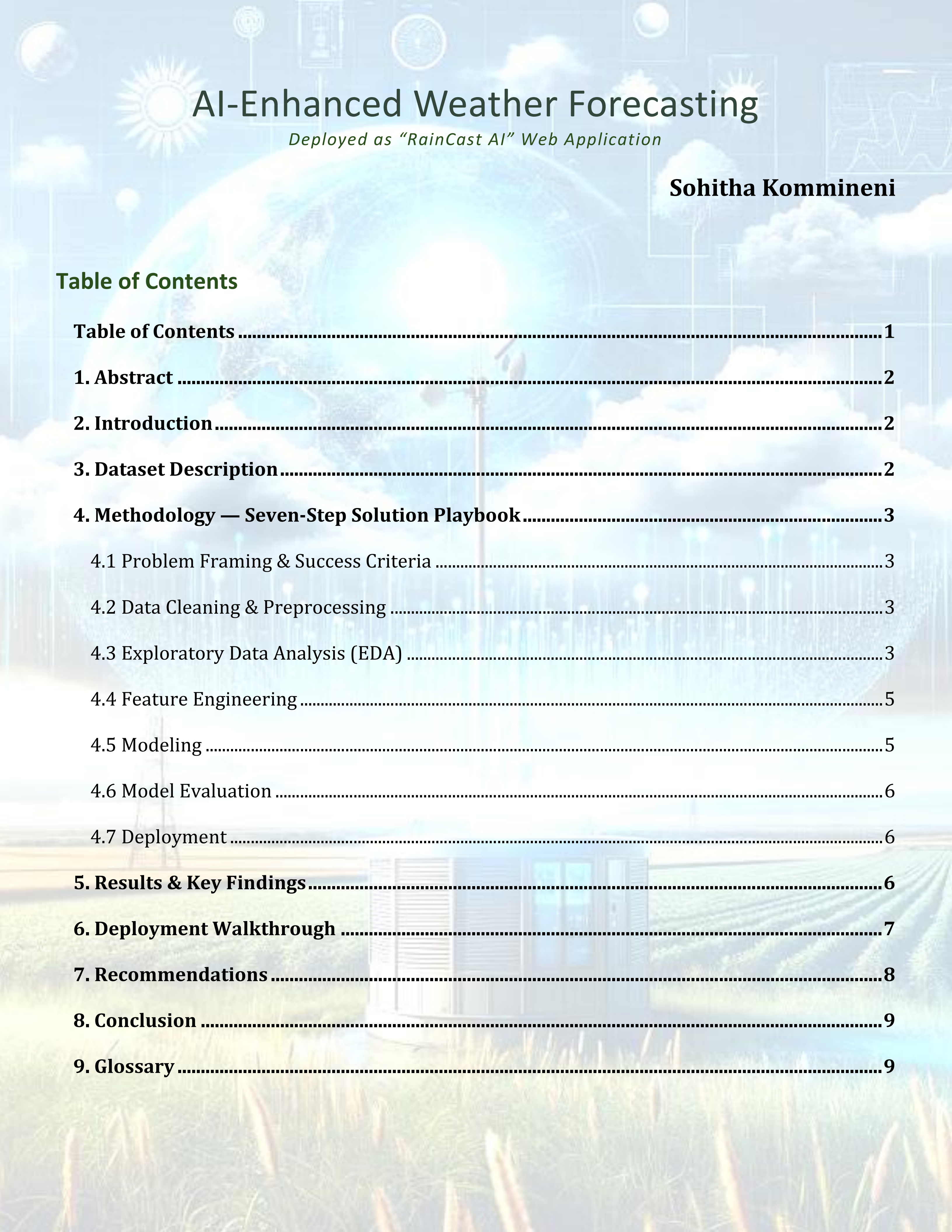


AI-Enhanced Weather Forecasting

Deployed as “RainCast AI” Web Application

Sohitha Kommineni

Table of Contents

1. Abstract

Weather forecasting is a critical problem that impacts daily life, agriculture, and disaster management. In this project, we ask key questions such as:

- What factors most strongly influence whether it rains tomorrow?
- Can machine learning provide better short-term forecasts than simple heuristics?
- How can we transform raw weather data into a usable tool for everyone?

To answer these, built a full pipeline that cleans and processes raw weather data, explores patterns through analysis, engineers meaningful features, and trains predictive models. Tested Logistic Regression, Random Forest, and XGBoost, finding that advanced ensemble models performed best. Finally, deployed the solution as an interactive Streamlit application called "RainCast AI," where anyone can input today's conditions and get a forecast for tomorrow.

2. Introduction

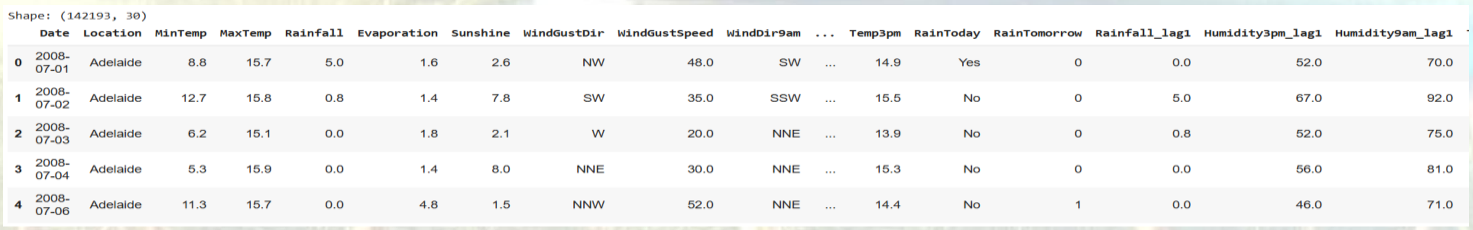
Weather influences nearly every aspect of life, from agriculture to daily commuting. Accurate forecasts can save resources, improve planning, and reduce risks. Traditional weather forecasting relies on physics-based numerical models that are powerful but computationally heavy. Machine learning offers a complementary approach: using historical data to detect patterns and make predictions more quickly.

This project, AI-Enhanced Weather Forecasting, demonstrates how machine learning can predict short-term rainfall. By building a pipeline and deploying the final model as a web app (RainCast AI), show how raw data can become an accessible tool for everyday users.

3. Dataset Description

Used data from the Australian Bureau of Meteorology. It contains ~142k daily weather observations across multiple locations. Key variables include:

- Temperature: Recorded at 9 AM and 3 PM.
- Humidity: Morning and afternoon readings.
- Rainfall: Amount of rain recorded in millimeters.
- Pressure: Atmospheric pressure at different times.
- Sunshine: Number of bright sunshine hours.
- Wind: Speed and direction.
- RainToday: Whether it rained today (Yes/No).
- RainTomorrow: Target variable — will it rain tomorrow (Yes/N0)



```
Shape: (142193, 30)
```

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Temp3pm	RainToday	RainTomorrow	Rainfall_lag1	Humidity3pm_lag1	Humidity9am_lag1
0	2008-07-01	Adelaide	8.8	15.7	5.0	1.6	2.6	NW	48.0	SW	...	14.9	Yes	0	0.0	52.0	70.0
1	2008-07-02	Adelaide	12.7	15.8	0.8	1.4	7.8	SW	35.0	SSW	...	15.5	No	0	5.0	67.0	92.0
2	2008-07-03	Adelaide	6.2	15.1	0.0	1.8	2.1	W	20.0	NNE	...	13.9	No	0	0.8	52.0	75.0
3	2008-07-04	Adelaide	5.3	15.9	0.0	1.4	8.0	NNE	30.0	NNE	...	15.3	No	0	0.0	56.0	81.0
4	2008-07-06	Adelaide	11.3	15.7	0.0	4.8	1.5	NNW	52.0	NNE	...	14.4	No	1	0.0	46.0	71.0

4. Methodology — Seven-Step Solution Playbook

Followed a structured methodology to ensure clarity, reproducibility, and completeness.

4.1 Problem Framing & Success Criteria

Framed this as a binary classification problem: predicting RainTomorrow (Yes/No). Success criteria were defined as achieving a ROC-AUC score above 0.88 and maintaining a balance between precision and recall.

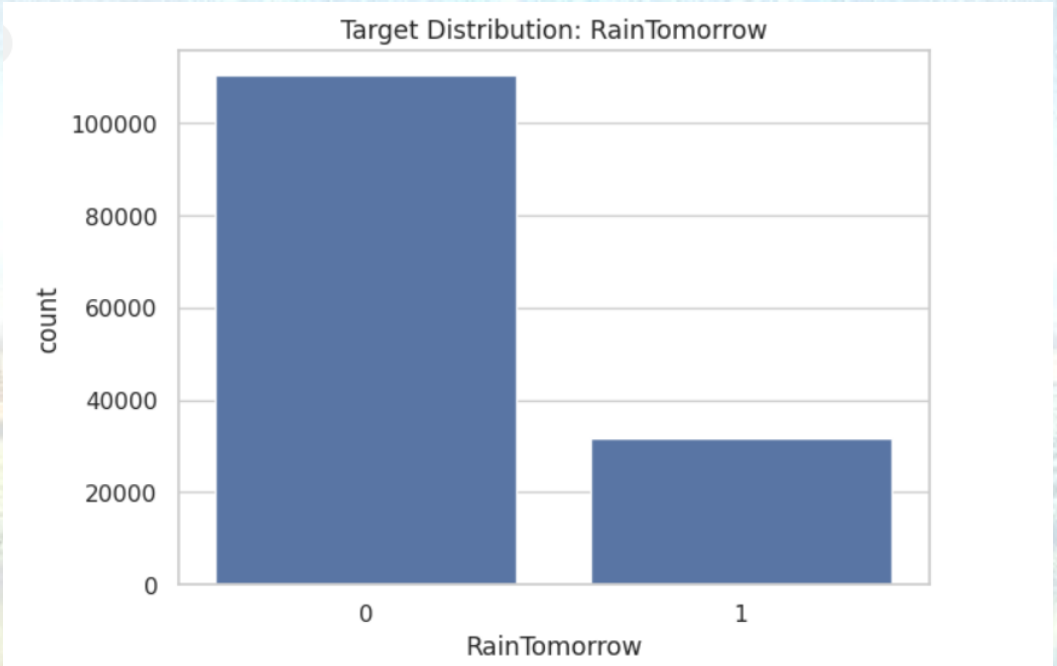
4.2 Data Cleaning & Preprocessing

Data cleaning involved handling missing values, encoding categorical features, and scaling numerical variables. Applied median imputation for numeric variables and mode imputation for categorical ones. Categorical variables like WindDirection were converted using one-hot encoding. Numeric features were standardized to improve model stability. And also performed a stratified train-test split to maintain the target class balance.

4.3 Exploratory Data Analysis (EDA)

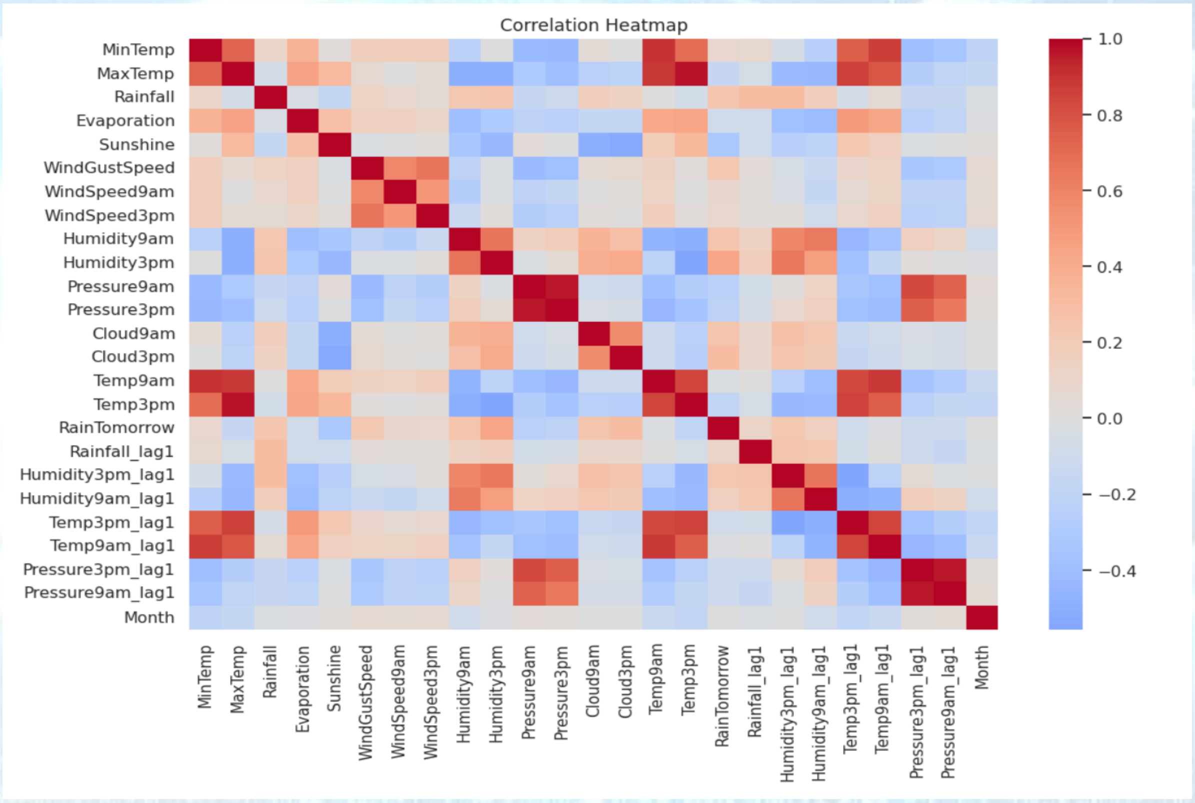
EDA helped to understand data distribution, relationships, and seasonal patterns. The target variable RainTomorrow showed imbalance, with rainy days forming only ~22%. Correlation analysis highlighted strong predictors like Humidity, Rainfall, and Pressure. Seasonal trends revealed rainfall peaks in mid-year months (winter in Australia).

- **Target distribution:** only about one in five days are rainy, so a naive “always no” model would score high accuracy but be useless.

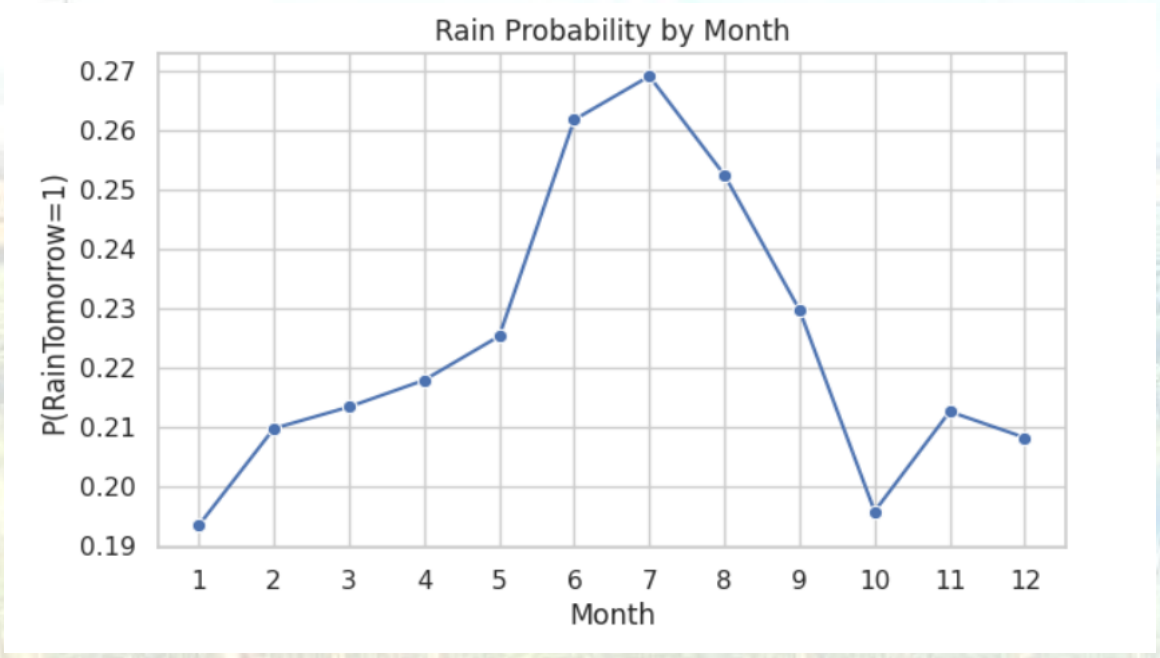


```
RainTomorrow=1 ratio: 22.42%
```

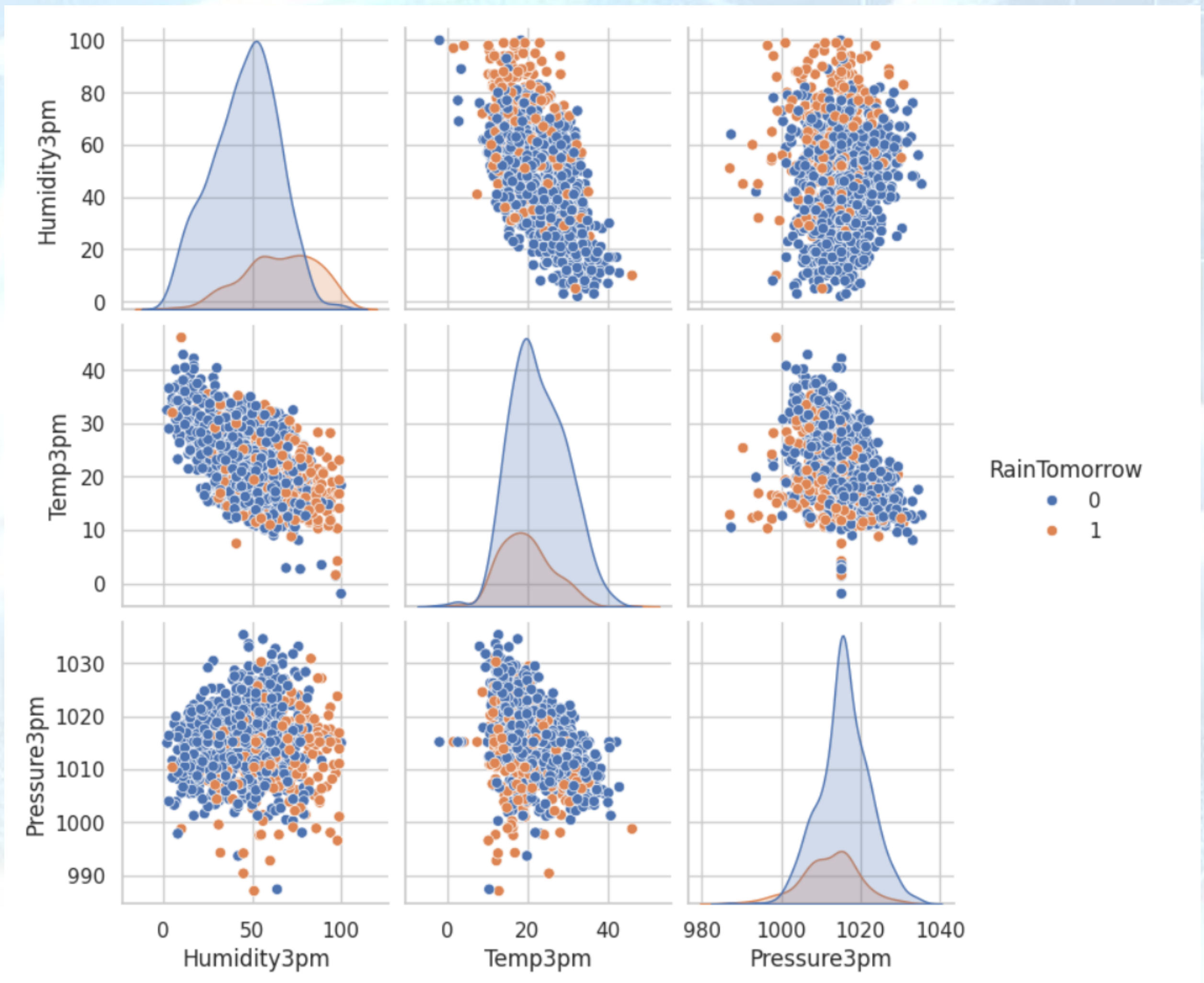
➢ **Correlation heatmap:** Humidity at 3pm, rainfall, and pressure show the strongest relationships with RainTomorrow. This view helps prioritize features and anticipate model behavior (e.g., multicollinearity among temperature metrics).



➢ **Seasonality:** rain probability peaks mid-year (June–July) and dips around October. This matches Australia's climate. Such structure suggests value in calendar features and cautions against evaluating only on short windows.



➢ **Pairwise views:** Observed cooler, more humid afternoons on rainy-tomorrow cases, and slightly lower pressure. The classes overlap—hence the need for non-linear models that capture subtle interactions beyond thresholds.



4.4 Feature Engineering

Engineered lag-1 features (e.g., Humidity3pm_lag1, Pressure9am_lag1, Rainfall_lag1). The intuition is simple: weather has memory. Yesterday's moisture and pressure carry signal for tomorrow. We also converted RainToday to 0/1. All engineered columns are created inside the training pipeline to avoid leakage and kept optional in the app (imputers handle missing user inputs).

4.5 Modeling

Trained three models:

- Logistic Regression: A baseline model that is simple and interpretable.
- Random Forest: An ensemble model that captures non-linear relationships and provides feature importance.
- XGBoost: A powerful gradient boosting algorithm that achieved the best overall performance.
- Hyperparameters were tuned sensibly for speed and stability. Class imbalance was handled via stratification and threshold tuning using the F1-optimal point from precision-recall curves.

4.6 Model Evaluation

Evaluated models using metrics such as Accuracy, Precision, Recall, F1-score, and ROC-AUC. Accuracy gives overall correctness, Precision measures correctness of positive predictions, Recall measures ability to detect rainy days, and ROC-AUC shows overall ranking quality. These metrics together ensured a fair and comprehensive evaluation. XGBoost delivered the best overall ROC-AUC (~0.91) with balanced precision/recall; Random Forest was a close second and more lightweight; Logistic Regression provided a strong baseline.

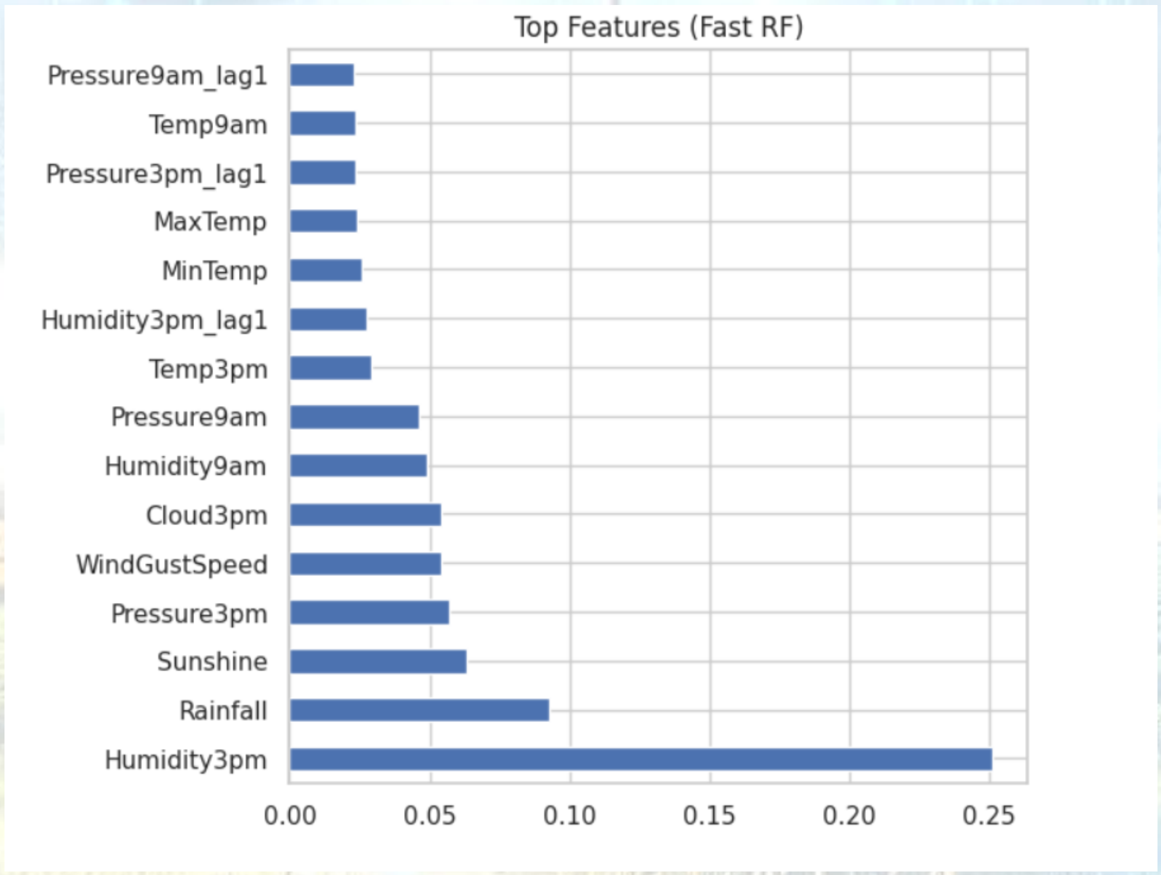
4.7 Deployment

The final model was deployed as a Streamlit application named "RainCast AI". Users can input today's weather conditions (temperature, humidity, pressure, etc.) and receive predictions for tomorrow.

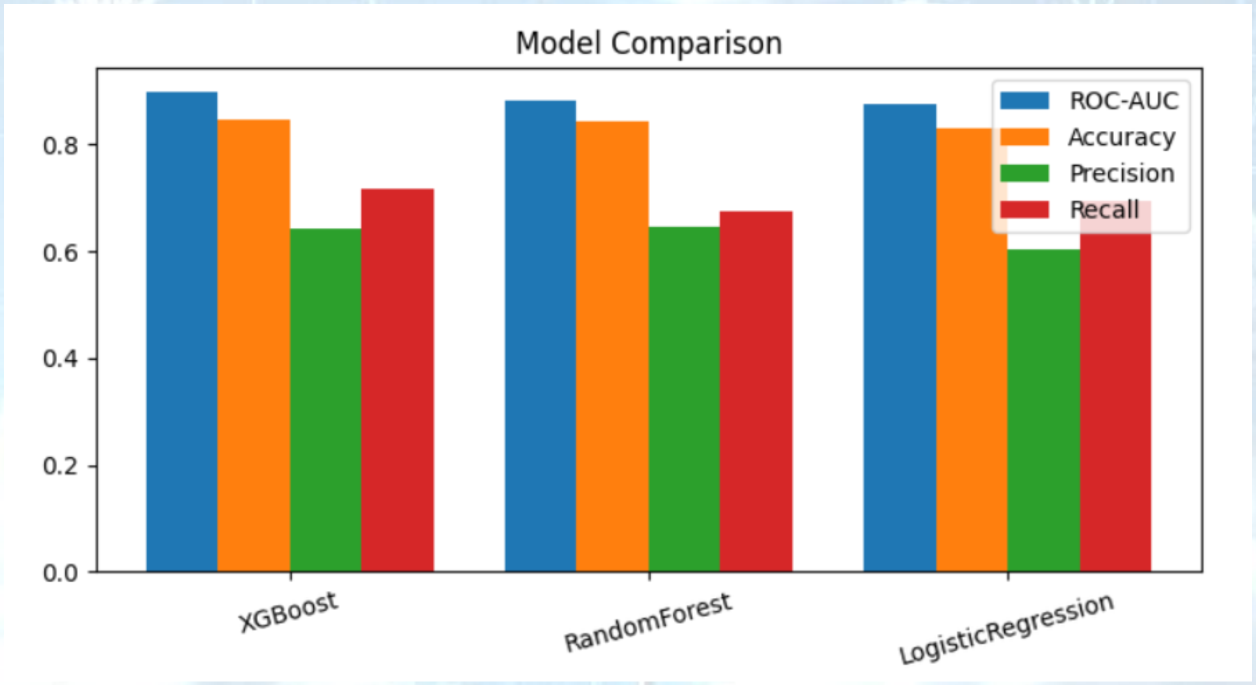
- **Live App:** RainCast AI – Predict Tomorrow's Rain

5. Results & Key Findings

- **Feature importance (from a fast Random Forest) confirms domain intuition:** afternoon humidity dominates, with rainfall, sunshine, and pressure also influential. This transparency supports trust and allows domain stakeholders to critique inputs.



- **Model comparison:** XGBoost marginally leads in ROC-AUC and recall, meaning it catches more rainy days at similar precision. Random Forest is competitive and simpler to maintain. Logistic Regression is fastest and most interpretable but sacrifices recall.



Feature importance analysis showed Humidity (3PM), Rainfall, Sunshine, and Pressure were the strongest predictors. Model comparison revealed that XGBoost achieved the highest ROC-AUC (~0.91), Random Forest closely followed (~0.90), and Logistic Regression provided a strong baseline (~0.87).

6. Deployment Walkthrough

Built a Streamlit frontend that loads a serialized model and (optionally) a thresholds JSON. To keep the GitHub repo light, the app can auto-download large artifacts from public URLs (e.g., Hugging Face) via environment variables.

Deployment involved two stages: local deployment and cloud deployment.

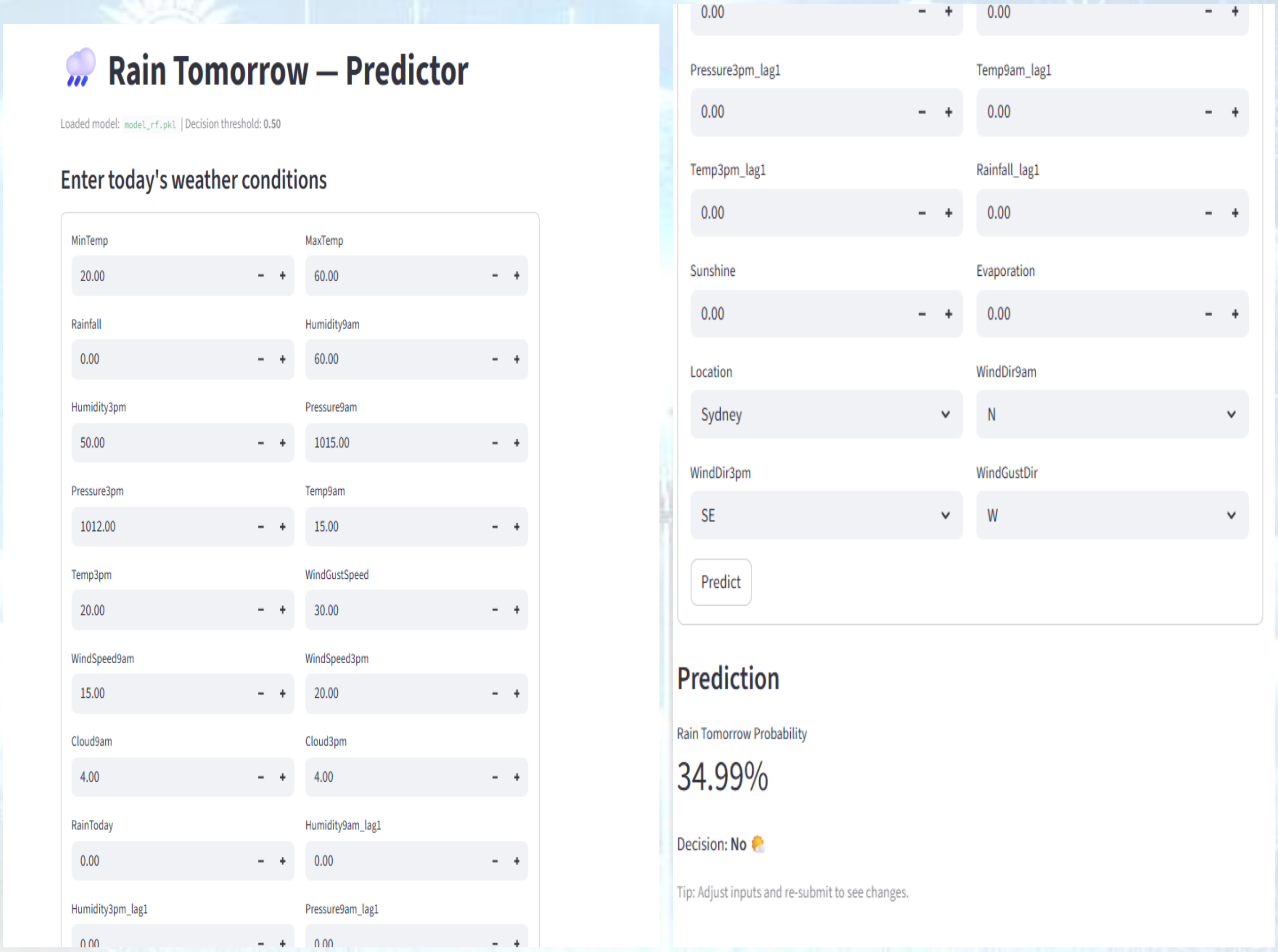
Local Deployment Commands:

- ✓ pip install -r requirements.txt
- ✓ streamlit run deployment/app.py

Streamlit Cloud Deployment Steps:

1. Push only small files (`app.py`, `requirements.txt`, README, Threshold, Clean_CSV) to GitHub. Excluded large `. pkl` file because of file size.
2. Hosted the model on Hugging Face and copied the raw file URL.
3. In Streamlit Cloud → App → Settings → Secrets, add environment variables in TOML format, e.g: MODEL_URL = "https://huggingface.co/yourname/weatherartifacts/resolve/main/model_rf.pkl"
4. Save and redeploy. On startup, the app downloads missing artifacts and becomes immediately usable from a public

App UI examples. The form accepts today's conditions; the app returns a probability and a clear Yes/No decision.



Rain Tomorrow — Predictor

Loaded model: `model_rf.pkl` | Decision threshold: 0.50

Enter today's weather conditions

MinTemp: 20.00	MaxTemp: 60.00
Rainfall: 0.00	Humidity9am: 60.00
Humidity3pm: 50.00	Pressure9am: 1015.00
Pressure3pm: 1012.00	Temp9am: 15.00
Temp3pm: 20.00	WindGustSpeed: 30.00
WindSpeed9am: 15.00	WindSpeed3pm: 20.00
Cloud9am: 4.00	Cloud3pm: 4.00
RainToday: 0.00	Humidity9am_lag1: 0.00
Humidity3pm_lag1: 0.00	Pressure9am_lag1: 0.00
Pressure3pm_lag1: 0.00	Temp9am_lag1: 0.00
Temp3pm_lag1: 0.00	Rainfall_lag1: 0.00
Sunshine: 0.00	Evaporation: 0.00
Location: Sydney	WindDir9am: N
WindDir3pm: SE	WindGustDir: W

Prediction

Rain Tomorrow Probability

34.99%

Decision: **No**

Tip: Adjust inputs and re-submit to see changes.

7. Recommendations

➢ **Short-Term Actions**

- Optimize Thresholds for Recall – Tune the prediction threshold to minimize missed rainy days, especially for agriculture and public safety use cases.
- Improve User Guidance in the App – Add tooltips and explanations in the Streamlit app so non-technical users understand how to interpret predictions.

➢ **Mid-Term Actions**

- Expand Data Sources – Integrate external signals like satellite imagery, cloud cover, and ENSO indices to strengthen predictive accuracy.
- Automate Model Retraining – Set up a pipeline that retrains the model periodically with the latest weather data to keep it adaptive and reliable.

➢ **Long-Term Actions**

- Extend Forecast Horizon – Move beyond predicting just tomorrow's rain to 2–7 day forecasts, increasing real-world planning value.
- Multi-Platform Deployment – Integrate the model into mobile apps, dashboards, or SMS/email alert systems so stakeholders can access predictions anytime, anywhere.

8. Conclusion

This project successfully built an end-to-end machine learning system that predicts whether it will rain tomorrow. By cleaning and preparing real-world weather data, exploring patterns through EDA, and engineering meaningful features, created models that balance accuracy and practical value. Among the models tested, XGBoost delivered the strongest performance, while Random Forest provided a reliable and interpretable alternative.

The final Streamlit application transforms this technical work into an accessible tool, enabling anyone to input today's conditions and instantly receive tomorrow's rain forecast. Beyond technical results, this project shows how data science can bridge raw data and real-world decision-making, providing insights that matter to both individuals and organizations.

9. Glossary

- **Lag Features:** Variables that use yesterday's values to help predict tomorrow.
- **ROC-AUC:** A metric showing how well the model ranks positive vs. negative cases; higher is better.
- **Precision:** Out of all predicted rainy days, how many were actually rainy.
- **Recall:** Out of all actual rainy days, how many were correctly predicted.
- **F1-score:** A balance between precision and recall.
- **Ensemble Models:** Models like Random Forest and XGBoost that combine many smaller models to improve performance.