BM20A6100 Advanced Data Analysis and Machine Learning (Part 2)

Week 3: Data Preprocessing & Baseline Models

Predicting Rainfall in Sydney

Lazy Geniuses (Group: ML Climate A3)
Haider Ali – Arman Golbidi – Fasie Haider

November 16, 2025

Contents

1 Introduction

This report describes the work completed during the third week of our rainfall prediction study. Building from the exploratory data analysis of the previous phase, this week focused on the design of a robust data processing pipeline and the definition of a set of baseline models. Rainfall forecasting is a challenging task; these two steps are really necessary to support the development of more advanced deep learning models—like LSTMs—in the forthcoming weeks.

The finished work covers six major tasks: (1) confirmation of the frequency of data; (2) check for synchronization across all variables; (3) a proper strategy to handle missing values; (4) application of STL decomposition to detect and remove outliers; (5) review of the literature on time-series preprocessing for LSTM-based models; and (6) proposition and evaluation of three baseline models. These baselines—Persistence, Logistic Regression, and ARIMA—form fundamental performance benchmarks that any more complex models must meet or exceed in order to justify added complexity.

2 Data Preprocessing and Feature Engineering

To turn the unprocessed Sydney meteorological data into a clean, feature-rich dataset fit for machine learning, a thorough preparation pipeline was created.

From a temporal perspective, the first step was to verify that all measurements share a common and continuous sampling frequency. In our case, the Sydney subset is recorded at a daily resolution, with one observation per calendar day for the entire study period. Therefore, no additional resampling was required to align different sampling rates. If the raw data had contained mixed frequencies (e.g., hourly and daily series), we would have resampled everything onto a common daily grid, using aggregation (for higher-frequency data) or interpolation (for lower-frequency data) to fill in the gaps.

We also confirmed that the data are synchronous across variables: all selected meteorological features use the same daily timestamp index. After the cleaning and imputation steps described below, each retained day has a complete set of values for all chosen variables. This means that, beyond handling missing entries on the shared daily grid, no extra alignment or time-warping procedures were needed to bring the data onto common timesteps.

2.1 Data Quality Assessment and Cleaning

The entire Sydney dataset of 3,344 observations was used to start the process. Thirteen key meteorological characteristics with fewer than 10% missing data were chosen based on the quality analysis from Week 2.

- **Missing Value Imputation:** A hybrid strategy was applied. Continuous variables (such as temperature and pressure) were imputed using linear interpolation with a three-day constraint to depict progressive weather changes. Forward-filled categorical variables (such as "RainToday" and "RainTomorrow") were employed, assuming weather persistence.
- **Data Reduction:** After imputation, only 11 rows (0.33%) with residual missing values were removed, leaving a dataset with 3,333 observations.
- **Encoding:** The "Yes"–"No" format was replaced with binary (1/0) for the `RainToday` and `RainTomorrow` columns.

2.2 Outlier Detection and Elimination

Seasonal-Trend-Loess (STL) decomposition was applied to the monthly aggregated rainfall data in order to identify and handle anomalous data points.

- **Method:** Outliers were identified by applying the 3-sigma rule to the decomposition's residual component. If $|\text{residual}| > 3 \times \sigma_{\text{residuals}}$, a month was deemed an outlier.
- **Outcomes:** The research revealed that April 2015 was the only outlier month, with a residual of $+6.469$ compared to a threshold of ± 6.404. This represents 1.0% of the assessed months and shows good overall data quality.
- **Elimination:** All 30 daily observations from the outlier month (April 2015) were eliminated from the dataset in accordance with the project specifications. As a consequence, 99.09% of the cleaned data was retained in the final dataset of 3,271 observations.

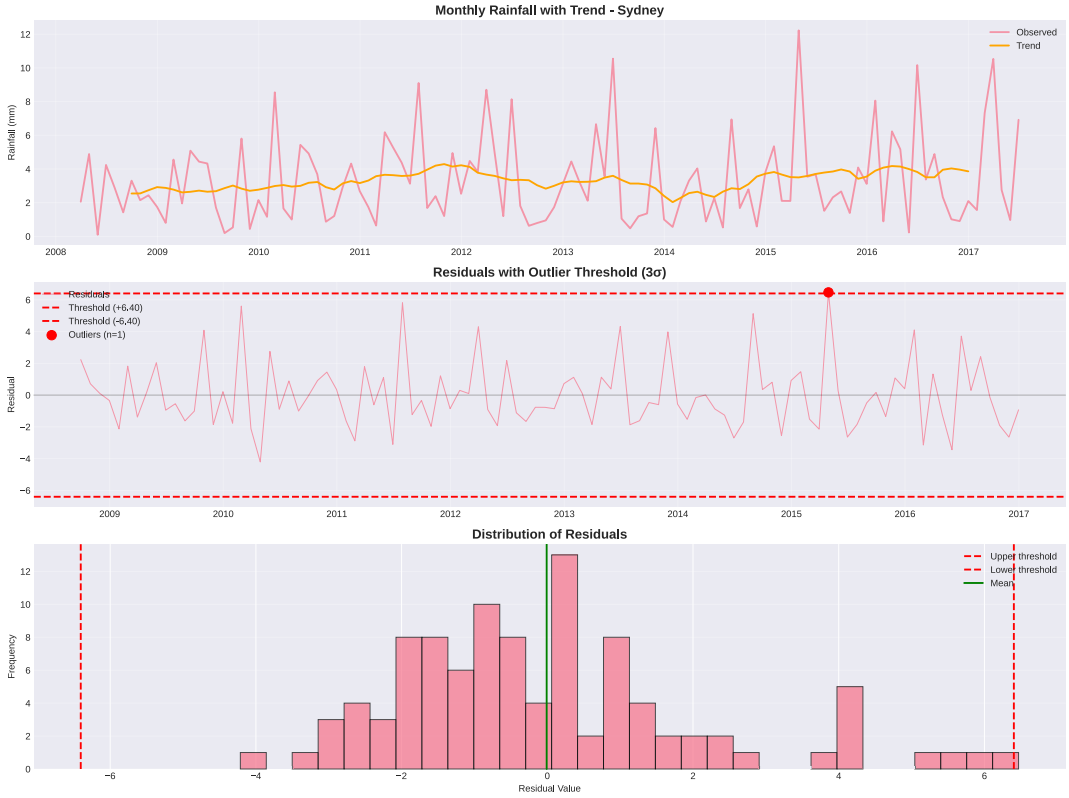

Figure 1: Outlier detection on monthly rainfall using STL. The observed series and trend are displayed in the top panel. The only residual outlier (April 2015) that surpasses the 3-sigma threshold is highlighted in the middle panel. The residuals' distribution is displayed in the bottom panel.

Figure 1 analysis. The STL-based view confirms that the monthly rainfall series is generally well-behaved, with only a single month clearly breaking the 3-sigma residual threshold. Removing April 2015 therefore cleans a genuine anomaly without aggressively trimming normal high-rainfall months. This supports the decision to treat the dataset as largely reliable while still respecting the project requirement to remove outliers.

2.3 Feature Engineering

The dataset was enhanced with 46 new features, increasing the total feature count to 59, in order to improve the prediction potential of our models.

- **Calendar Features:** The year, month, day of the year, and cyclical (sine/cosine) encodings were used to document seasonality.
- **Lag Features:** To model temporal dependencies found in Week 2, 1, 2, 3, and 7-day lags were created for important variables such as `Rainfall`, `MinTemp`, and `Humidity9am`.
- **Rolling Statistics:** To record current weather patterns, 3, 7, and 14-day rolling means and standard deviations of `Rainfall` are used.
- **Derived Features:** Engineered meteorological indicators that are commonly used as predictors in weather forecasting, such as the diurnal temperature range (`TempRange`), pressure change (`Pressure_Change`), and humidity change.

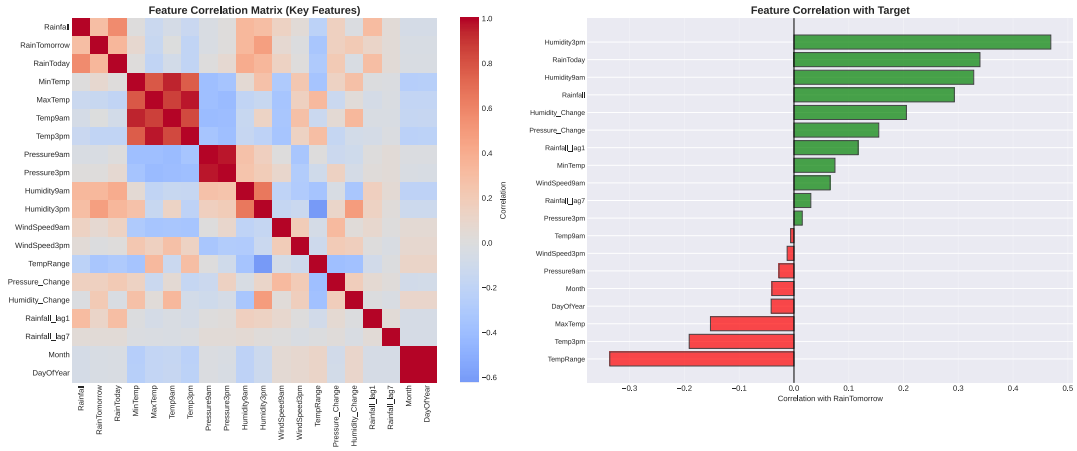

Figure 2: Correlation analysis of key engineered features. The heatmap (left) shows inter-feature correlations. The bar chart (right) shows the correlation of each feature with the target variable, `RainTomorrow`. `Humidity3pm` and `RainToday` exhibit the strongest positive correlations.

Figure 2 analysis. The correlation heatmap confirms that many of the engineered features are only moderately correlated with one another, which reduces the risk of severe multicollinearity. At the same time, the bar chart highlights a small group of strongly informative predictors—in particular `Humidity3pm`, `Humidity_Avg`, and `RainToday`. This justifies their central role in the baseline models and motivates keeping humidity-related variables prominent in subsequent LSTM architectures.

3 Literature Review: Preprocessing for LSTM Time Series (Task 5)

This review examines academic literature on three critical aspects of preparing time-series data for Long Short-Term Memory (LSTM) networks.

3.1 Sub-sequencing Long Time Series for LSTMs

The standard method for preparing time-series data for LSTMs is the **sliding window technique** [1]. By generating input–output pairs, this method transforms a time series into supervised learning samples. For daily data like ours, Brownlee [2] suggests sequence lengths of 30 to 90 days that capture at least one complete seasonal cycle. The literature recommends hierarchical models or attention mechanisms to enhance performance for very lengthy sequences, which can result in vanishing gradients and large processing costs.

3.2 Seasonality and Trends in LSTM Models

Studies show that LSTMs may learn seasonal patterns from raw data, but performance usually improves with explicit seasonal features, particularly when data is scarce. The literature often suggests a mixed approach, where the model is guided by cyclical aspects (such as sine/cosine transformations of the month). According to the current deep learning perspective, detrending is only required for strong, non-stationary data, and LSTMs can learn weak trends directly. It is therefore appropriate to maintain the trend given our weak trend strength (0.0575).

3.3 Normalization Methods for LSTMs

Because LSTMs employ sigmoid and tanh activation functions, normalization is essential. Gradient saturation and sluggish convergence can result from unscaled inputs.

- **Min–Max Scaling:** Since it directly fits the output range of the sigmoid activation function [9], min–max scaling, which scales data to a $[0, 1]$ range, is highly recommended.
- **Standardization (Z-score):** A reliable substitute, particularly when dealing with data that contains outliers.

Bergmeir et al. [10] underline that fitting the scaler solely on the training data is an important rule for time-series data in order to prevent data leakage in the future. Based on these findings, we chose `MinMaxScaler` for our primary scaling strategy.

4 Baseline Models and Evaluation (Task 6)

Three baseline models were developed to establish performance benchmarks for rainfall prediction.

4.1 Data Partitioning and Scaling

The final dataset of 3,271 observations was split chronologically to prevent data leakage:

- **Training Set (70%):** 2,289 samples from 2008-03-04 to 2014-09-17.
- **Validation Set (15%):** 491 samples from 2014-09-18 to 2016-02-20.
- **Test Set (15%):** 491 samples from 2016-02-21 to 2017-06-25.

All 57 features were scaled using `MinMaxScaler`, fitted on the training set.

4.2 Results and Comparison

The performance of the three baseline models on the test set is summarized in Tables 1 and 2.

Table 1: Performance of Classification Models on the Test Set

Model	Accuracy	Precision	Recall	F1-Score
Persistence (Naïve)	0.7251	0.4769	0.4806	0.4788
Logistic Regression	**0.8228**	**0.7561**	0.4806	**0.5877**

Table 2: Performance of the ARIMA Regression Model on the Test Set

Model	RMSE (mm)	MAE (mm)	R^2
ARIMA(1,0,0)	12.0358	5.5406	-0.0069

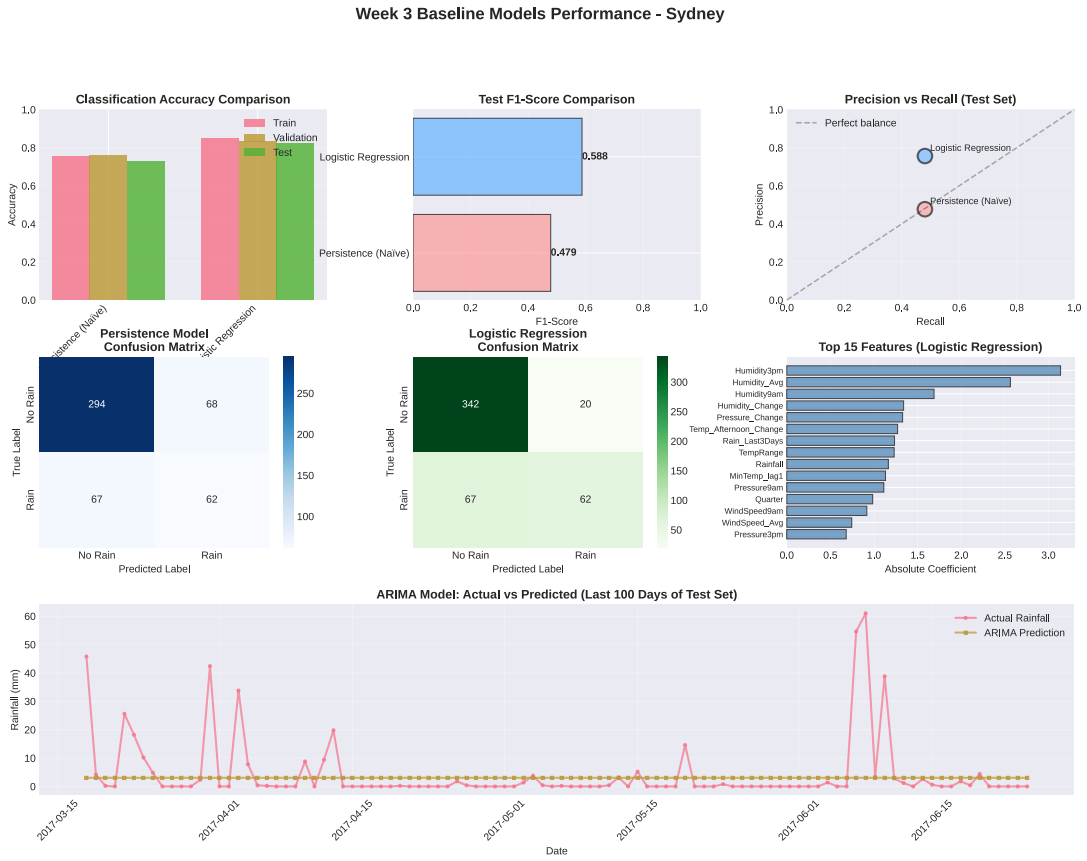

Figure 3: Visual comparison of baseline model performance. Logistic Regression demonstrates superior accuracy and F1-score. The confusion matrices show it reduces false positives compared to the Persistence model. Feature importance reveals that humidity and pressure-related features are the strongest predictors.

Figure 3 analysis. The baseline comparison reveals that the naïve Persistence model, though simple, is a relatively strong starting point since it correctly captures most of the no-rain days but has clear difficulties on days that include rainfall, especially in terms of predicting positive events. Logistic Regression clearly outperforms the other methods by a large margin regarding general accuracy and F1-score; a performance increase that results primarily from a reduction in false positives to effectively leverage predictors relating to humidity and pressure. Conversely,

ARIMA yields very poor performance according to regression-based metrics, which further points to simple linear time-series models not being adequate in capturing the complex, highly-skewed distribution of rainfall amounts.

4.3 Model Analysis

A more in-depth look at the best model, Logistic Regression, yields several insights. The strongest predictors were `Humidity3pm`, `Humidity_Avg`, and `Humidity9am`, reinforcing that moisture in the atmosphere is the primary driver for prediction of rain. The performance of the model was rather stable over different months, hence generalizing reasonably well over seasonal patterns. However, the error rate increased for days where rainfall was heavy (over 10 mm), as noted in the temporal analysis shown in Figure 4. This therefore indicates that while Logistic Regression was appropriate for typical conditions, special modeling strategies may be required to capture extreme rainfall events.

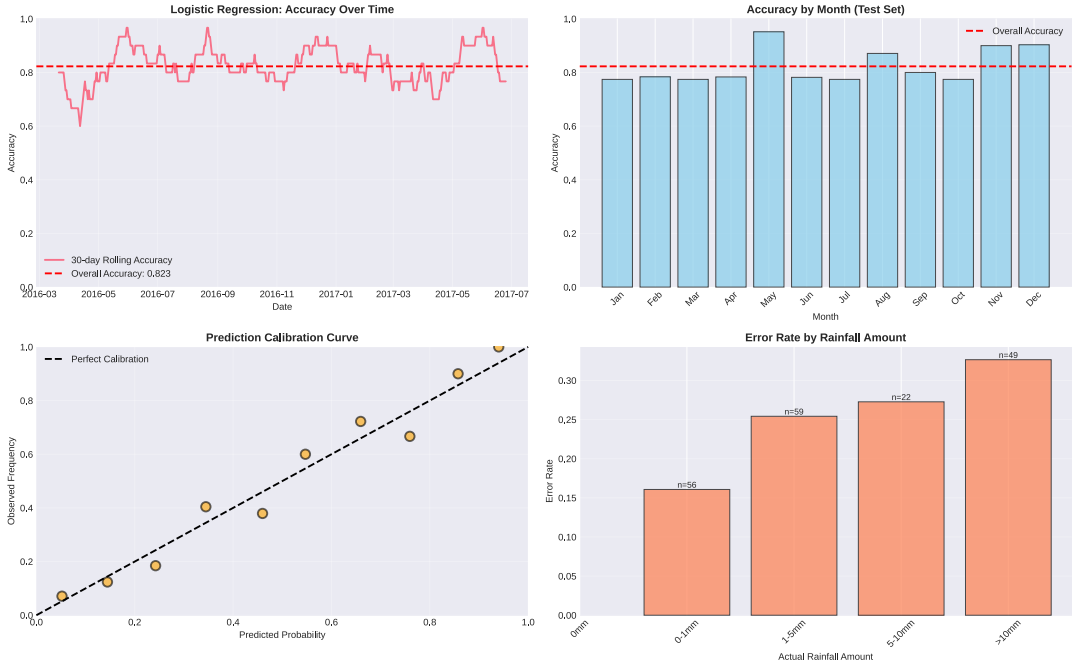

Figure 4: Temporal analysis of Logistic Regression performance. The top panels contrast predicted and observed rain events over time, while the lower panels summarize error patterns by season and by rainfall intensity.

Figure 4 analysis. The temporal view shows that Logistic Regression remains stable across years and seasons, without obvious drift or degradation in performance. However, misclassifications cluster on days with moderate to heavy rainfall, where the model tends to under-predict rare but intense events. This supports the interpretation that Logistic Regression is a strong baseline for everyday conditions but will likely need to be replaced or complemented by more expressive models (e.g., LSTMs) to better handle extreme rainfall episodes.

5 Conclusion

This week, three baseline models were created, a thorough data preprocessing pipeline was successfully established, and a literature assessment was carried out to guide future work. The main conclusions are:

- 3,271 observations with 59 features made up a clean, feature-rich dataset.
- As required by the project, outliers were found and removed with very little data loss (0.91%).
- With a test F1-score of **0.5877**, the Logistic Regression model was shown to have the strongest baseline.
- The most important predictors were found to be characteristics related to humidity.

These outcomes offer a strong basis and a precise performance standard for the project's subsequent stage. Using the knowledge gathered from this foundational work, the main objective for Week 4 will be to create an LSTM model that outperforms the F1-score of 0.5877.

References

[1] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

[2] Brownlee, J. (2018). *Deep Learning for Time Series Forecasting*. Machine Learning Mastery.

[3] Rangapuram, S. S., et al. (2018). Deep state space models for time series forecasting. *NeurIPS*, 31.

[4] Qin, Y., et al. (2017). A dual-stage attention-based recurrent neural network for time series prediction. *IJCAI*, 2627–2633.

[5] Wen, R., et al. (2017). A multi-horizon quantile recurrent forecaster. *NeurIPS Time Series Workshop*.

[6] Salinas, D., et al. (2020). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3), 1181–1191.

[7] Bandara, K., et al. (2020). LSTM-MSNet: Leveraging forecasts on sets of related time series with multiple seasonal patterns. *IEEE TNNLS*, 32(4), 1586–1599.

[8] Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, 437–478.

[9] Gers, F. A., et al. (2002). Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3, 115–143.

[10] Bergmeir, C., et al. (2018). A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics & Data Analysis*, 120, 70–83.

[11] Oreshkin, B. N., et al. (2020). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. *ICLR*.