# BOSTON WEATHER PREDICTION USING MLP AND TIME SERIES

**Siraj Akmal and Adam Ma**
DS4420
Professor Eric Gerber
Spring 2025

**Recognition and Statement of the Problem**

As students in Boston, one of the more frustrating parts of our day-to-day life is getting caught in the rain while walking to and from campus. The weather here is unpredictable, and sudden showers can ruin an otherwise ordinary commute. Carrying an umbrella "just in case" isn't always convenient, and weather apps often feel too general or inconsistent for our specific needs. This frustration led us to wonder: could we use machine learning to better predict whether it will rain on a given day?

As children we are often told that there are “rainy” seasons and that “April showers bring May flowers”. During our time in Boston we found that rainy days come often in clumps. In other words, if it rains one day, it seems more likely to rain again the next. This pattern hinted at underlying temporal dynamics that might be captured with time series modeling to see if there truly are seasonal patterns to weather. With that in mind, we set out to build two different kinds of models:

1. A multi-layer perceptron (MLP) model to classify whether it will rain on a given day.
2. Autoregressive (AR), Moving Average (MA), and Autoregressive Moving Average (ARMA) models to forecast total monthly precipitation, capturing trends over time.

**Overview of Past Work**

Weather prediction is a longstanding challenge that has been approached through various methods, ranging from physics-based simulations to statistical and machine learning techniques. Traditional weather forecasting models rely on large-scale atmospheric simulations, but these can be computationally expensive and are often designed for large scale production, as opposed to just deciding whether or not to bring an umbrella.

In recent years, researchers such as Stephan Rasp and Peter Dueben (WeatherBench: A

Benchmark Data Set for Data-Driven Weather Forecasting) have explored the use of deep learning in weather prediction, showing that machine learning models can emulate the behavior of physical models in certain scenarios. Others, like Aolin Zhang and Xinru Liu (Precipitation Forecast Based on Multi-Scale Stgnn), have experimented with neural networks to model rainfall intensity and duration, trying to capture complex patterns and relationships in meteorological data.

For time series forecasting, especially on a monthly scale, autoregressive models are a reliable choice. Past methods, such as that of researchers Kothapalli and Totad (A real-time weather forecasting and analysis) have applied ARIMA techniques to model seasonal rainfall patterns and monthly precipitation totals. These models are particularly good at capturing temporal dependencies, which made them well-suited to our goal of understanding longer-term trends in Boston's precipitation data.

**Data Collection**

To address both problems in our project we needed a dataset that was detailed, historical, and hyperlocal to Boston. We utilized an API provided by Weather Underground, a publicly accessible weather tracking website that maintains comprehensive daily records. We collected four years' worth of hourly weather data (from the start of 2020 to the end of 2023), which provided a sufficient sample size for training and evaluating both classification and time series models.

For MLP, we first converted the date column to a standardized datetime format, enabling easier extraction of time-based features and sorting. Missing values, which often represented zero precipitation on Weather Underground, were filled with zeroes across the dataset. We created a binary target column that labeled each day as either rainy or not, based on whether precipitation was recorded. To help the model learn temporal patterns, we generated new features such as the month, day of the year, and day of the week. Wind direction was transformed into sine and cosine components to preserve its cyclical nature. Additionally, pressure trend indicators, labeled as falling, steady, or rising, were

encoded to numeric values to make them usable by the model.

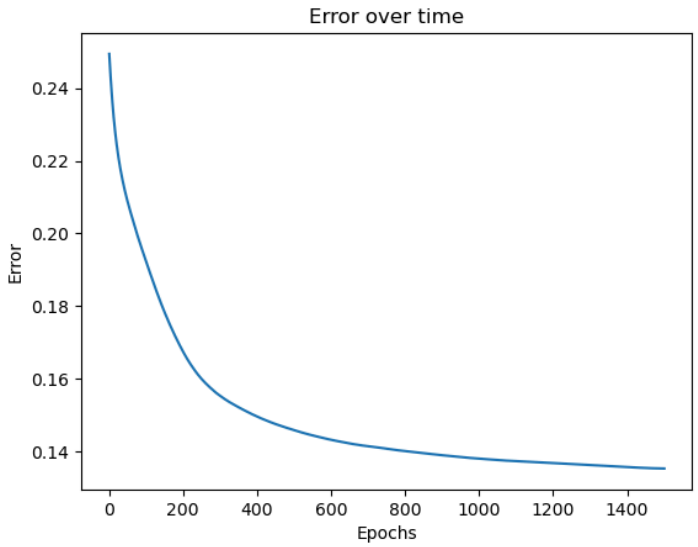
For the MLP classification model, we began by calculating numerical correlation scores between each feature and the occurrence of precipitation. From this analysis, we selected several of the most strongly correlated features to include in the model. We then added a few additional inputs based on common-sense reasoning and domain knowledge. The final set of input features included wind speed, pressure trend, humidity, UV index, atmospheric pressure, and the hour of the day. For example, humidity and UV index are widely understood to relate to weather patterns. High humidity often comes before rain, while lower UV levels may indicate increased cloud cover. This combination of data-driven and intuitive feature selection helped us create a model that reflects both statistical relationships and real-world weather behavior.

For the time series models, we originally experimented with models that attempted to predict hourly forecasts, but the sheer number of entries and seemingly stochastic nature of the data did not yield a meaningful signal, even with lag numbers as large as 50 or 100. We then experimented with an approach that aggregated rows by day, to attempt to predict daily forecasts, but this again generated unreliable predictions. Upon re-examination of the data, we discovered that there were very few rows in which rain was recorded in Boston. To solve this, we decided to group by month. We graphed the monthly rainfall over the course of the four years, and then performed and visualized a seasonal decomposition. Next, we plotted both an Autocorrelation Function (ACF) and a Partial Autocorrelation Function (PACF) applied onto the data. Finally, we created an AR model of lag 2, an MA model of lag 2, and an ARMA model of lag 2, and computed both the MSE and RMSE for each model on the test set of data.

**ML Results and Interpretation**

To evaluate our model's performance fairly, we created a balanced test dataset drawn from our

original dataset of over 40,000 rows. Specifically, we selected 500 rainy days and 500 no-rain days. This was an important step because in Boston, the number of dry days significantly outweighs rainy ones. Without balancing the test set, a naive model that predicted “no rain” every day might still appear to perform well simply due to class imbalance. By evaluating on a balanced subset, we ensured that the model's performance was not a result of bias toward the majority class.

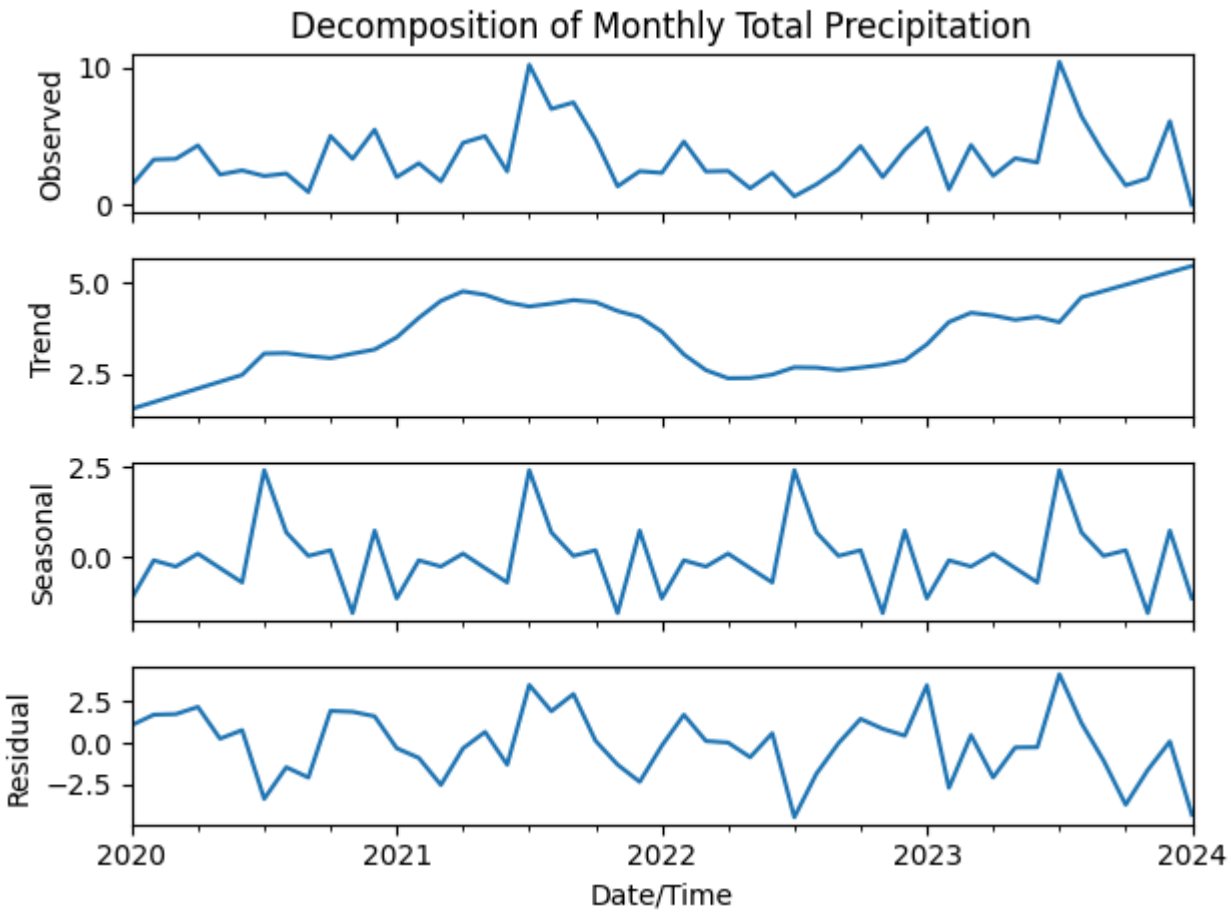



The graph above shows the error (loss) of our MLP model over 1,500 training epochs. The error begins around 0.25 and steadily decreases as training progresses, eventually stabilizing near 0.135. This consistent downward trend demonstrates that our model was learning effectively, minimizing prediction error through backpropagation.

On the balanced test set, the MLP achieved an accuracy of approximately 83%. This confirms that the model wasn’t simply guessing the dominant class, but was learning meaningful patterns in the data that allowed it to distinguish between rainy and dry days. Overall, the decreasing error curve and strong accuracy on a balanced test set provide clear evidence that our manually coded MLP was successful in learning to predict daily rainfall based on weather features.

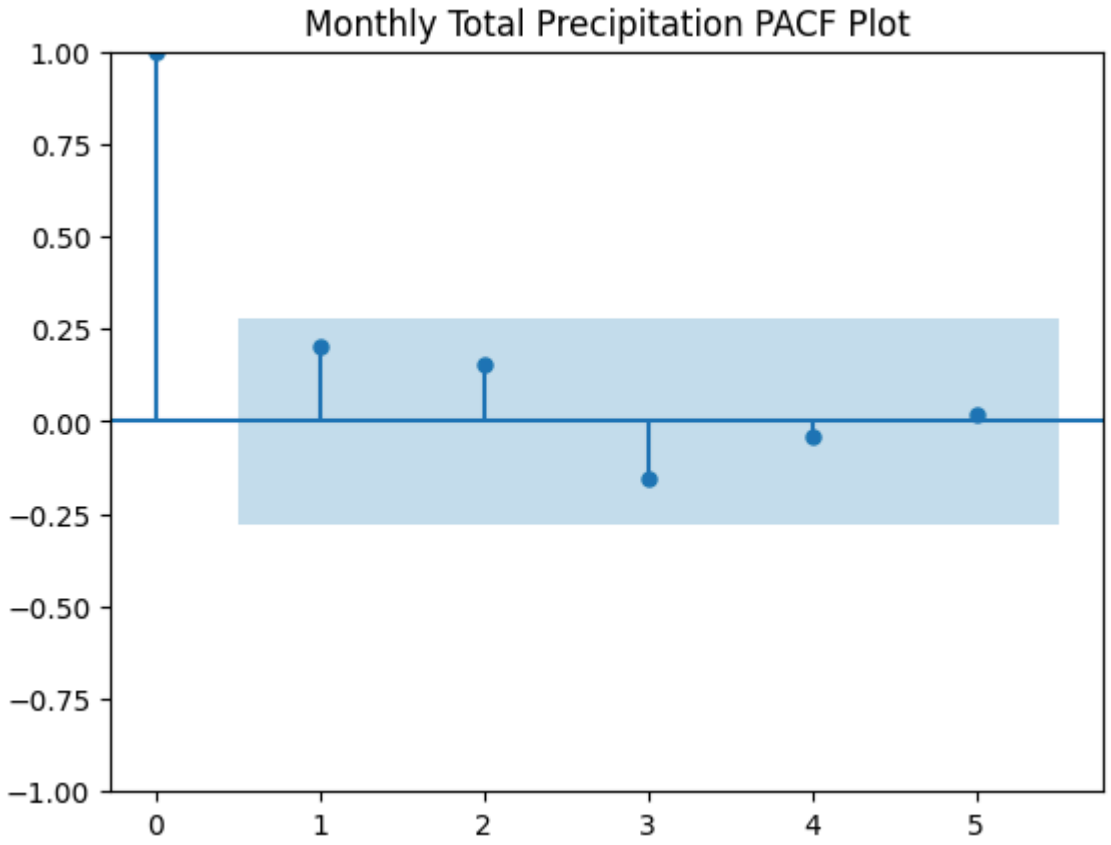
For the time series analysis, we began by decomposing the monthly precipitation data into

observed, trend, seasonal, and residual components. The trend line shows low-frequency fluctuations without a strong linear direction, indicating possible longer weather cycles. The seasonal component highlights recurring variation that aligns with expected seasonal changes in precipitation. The residual component still contains noticeable variance, suggesting that while trend and seasonality explain a portion of the variation, precipitation remains inherently noisy and influenced by external weather conditions.



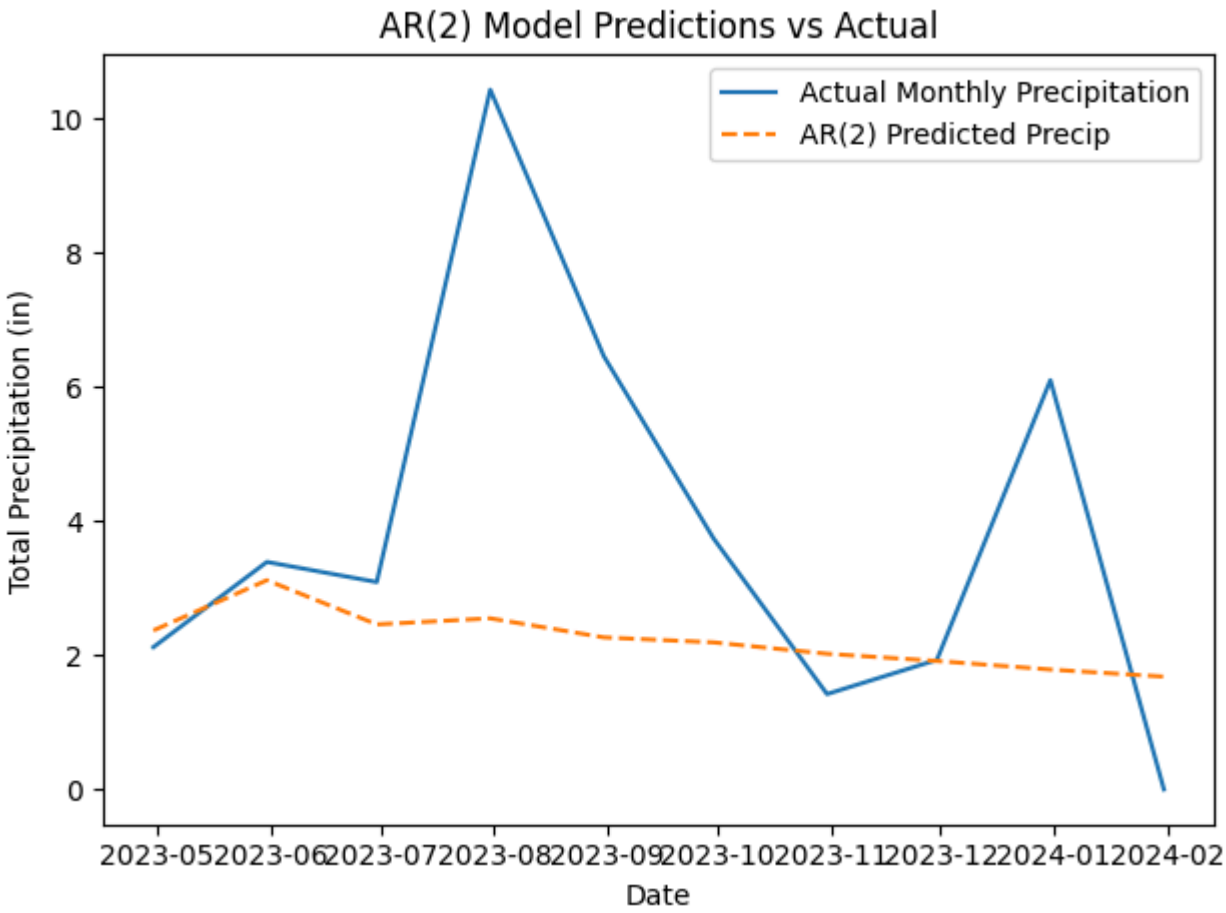

The PACF plot shows relatively strong significance at both lags 1 and 2, with remaining lags within the confidence bounds. This suggests that an autoregressive model with p = 2 is most appropriate for the data, as the PACF exhibits a sharp cutoff after lag 2. The pattern indicates that recent precipitation values within the past two months are predictive of precipitation levels in the

current month.




The ACF plot exhibits very similar patterns to the PACF plot, showing relatively strong significance at both lags 1 and 2 before a sharp cutoff. Using this, we can set q = 2 for a moving average model.

The AR(2) model included significant coefficients at both lag 1 (0.418, p = 0.004) and lag 2 (0.488, p = 0.001), reinforcing the belief that precipitation in the prior two months has a strong positive effect on current values. Furthermore, the roots are stationary. On the test set, the model produces smoothed predictions centered around the mean monthly precipitation, but cannot capture the sharp peaks and troughs observed in the actual data. The final mean absolute error and root mean squared error were 2.14 and 3.23 inches, respectively.




The MA(2) model included a significant constant term (3.27, $p < 0.001$), while both MA coefficients, lag 1 (0.187, $p = 0.180$) and lag 2 (0.394, $p = 0.058$), were positive but not statistically significant at the 5% level. This indicates that recent forecast errors may contribute weakly to predictions. Again, the roots are stationary. On the test set, the model produces nearly constant predictions after the first month (around roughly 3 inches). This suggests that the model failed to capture meaningful temporal structure in the residuals. As seen in the plot, the model does not react to the significant variability present in the data. The final mean absolute error and root mean squared error were 2.05 and 2.93 inches, respectively.

The ARMA(2,2) model included a significant constant term (3.27, $p < 0.001$), while all four lag coefficients, AR(1) (0.220, $p = 0.776$), AR(2) (–0.107, $p = 0.853$), MA(1) (–0.007, $p = 0.992$), and MA(2) (0.454, $p = 0.318$), were not statistically significant. This indicates that the model was unable to

identify meaningful contributions from either past values or past errors. The roots are stationary. On the test set, the ARMA(2,2) model, similar to the MA(2) model, produces relatively flat predictions after the initial month (again around 3.2 inches). This indicates that the model failed to leverage meaningful patterns from both past values and residuals. The final mean absolute error and root mean squared error were 2.09 and 2.92 inches, respectively.

**Conclusion and Future Work**

One major drawback of our analysis is that it focuses solely on Boston. While this localized approach helped us tailor the model to our daily experience, it limits generalizability. Weather patterns vary widely by region, and a model trained only on Boston data may perform poorly elsewhere. Expanding the dataset to include multiple cities or climate zones would allow for broader insights and more flexible applications.

Additionally, our model relied only on structured, numerical inputs like temperature, pressure, and humidity. While effective to a degree, this ignores richer data sources like radar imagery, satellite data, or textual weather reports. Incorporating these types of inputs could help capture more complex patterns and improve prediction accuracy. If we were to continue this project, we'd focus on scaling the model beyond Boston and integrating more diverse input data to enhance both accuracy and generalizability.