An Interpretable ML Pipeline for Day-Ahead Rain/No-Rain Prediction in Basel

Purvi Garg*

Department of Computational Mathematics, Science and Engineering

Michigan State University, East Lansing, MI 48824

(Dated: November 24, 2025)

Abstract

This study addresses the problem of predicting the next-day rainfall in Basel using only daily surface-station observations, without relying on numerical weather prediction products. Using the Kaggle "Weather Prediction" dataset (derived from ECAD, 2000–2010), I frame the task as a binary classification problem: rain versus no rain tomorrow. After restricting the data set to the Basel station, I built a time-sensitive target (RainTomorrow) by shifting the daily precipitation one day forward. The models use a small, meteorologically motivated feature set that includes surface pressure, humidity, temperature, sunshine, their one-day lags, and a simple month indicator to capture seasonality. Training is performed on the first 80 percent of days, and evaluation on the final 20 percent (2008–2010), in order to preserve temporal structure.

A trivial majority baseline that always predicts "no rain"" achieves a precision of approximately 0.52 and an F1 score of 0 for rainy days. In contrast, a tuned logistic regression model with standardized features and class weighting achieves 0.668 test accuracy and an F1 score of 0.668 for rain, with reasonably calibrated probabilities. A random forest and a shallow neural network reach slightly higher accuracies (approximately 0.67–0.68) but provide only modest gains over the logistic model. These results suggest that a simple, interpretable model based on basic station variables is sufficient to obtain useful day-ahead rain/no-rain predictions for Basel, while remaining transparent and easy to analyse and reproduce in a teaching setting.

BACKGROUND AND MOTIVATION

The central question in this project is whether next-day rainfall in Basel can be predicted from information available at the station at the end of the current day, without resorting to large-scale NWP products or radar fields. Many practical users of weather information (small municipalities, teaching labs, student projects) often have long tabular station records but lack the infrastructure to run or routinely access NWP. A lightweight and interpretable ML pipeline that extracts useful signal from such tabular data would therefore make short-range rainfall guidance more widely accessible.

Current operational forecasting is dominated by physics-based NWP and data assimilation, which deliver high-quality guidance but at nontrivial computational and organizational cost. At the other extreme, simple statistical rules such as climatology, persistence, or a single pressure-tendency rule are cheap but cannot combine multiple weak precursors. Machine learning provides a middle path: by formulating the task as supervised learning on a well-defined input–output map, we can let the model discover how pressure, humidity, sunshine, and recent rainfall interact, quantify improvement over realistic baselines, and inspect the learned patterns for meteorological plausibility. The specific contribution here is to demonstrate, on a real station-derived dataset, that a modest and transparent ML pipeline can outperform naive rules while respecting a leakage-safe time-based split.

DATA DESCRIPTION

Dataset and Exploratory Analysis

The dataset is the public "Weather Prediction" Kaggle dataset, derived from the European Climate Assessment & Dataset (ECA&D). ECA&D collects and quality-controls daily surface-station observations across Europe using standard instruments and procedures. The Kaggle version restricts the archive to 2000–2010, includes 18 stations (including Basel, Switzerland), and applies an initial cleaning step: columns with more than 5% invalid entries are removed and remaining missing values are imputed with column means. The file used in this project therefore has no explicit `NaN`s.

In this form, the dataset has about 3,654 daily records and roughly 160 mostly numerical variables: surface pressure, minimum/maximum/mean temperature, relative humidity, sun-

shine or global radiation, cloud cover, wind, and daily precipitation for each station, plus a DATE column. Because the prediction task is local to Basel and one day ahead, I keep only the Basel columns and treat them as a single-station time series embedded in the larger table. To define a classification target, I construct a same-day Basel rain indicator

$$\texttt{RainToday}_t = \mathbf{1}\{\text{BASEL\_precipitation}_t > 0\},$$

and then shift it forward to obtain

$$\texttt{RainTomorrow}_t = \texttt{RainToday}_{t+1},$$

so that day-t features always predict rain at day $t+1$. The last row is dropped because no label can be assigned, leaving 3,653 Basel days with a well-defined next-day label.

From the many available columns I keep a compact, meteorologically motivated predictor set: MONTH (1–12), RainToday, Basel surface pressure, relative humidity, mean temperature, and sunshine, together with one-day lags of the four Basel weather variables. All predictors passed to the models are numerical; DATE is used only for sorting and plotting. Because Kaggle preprocessing has already removed high-missingness columns and imputed

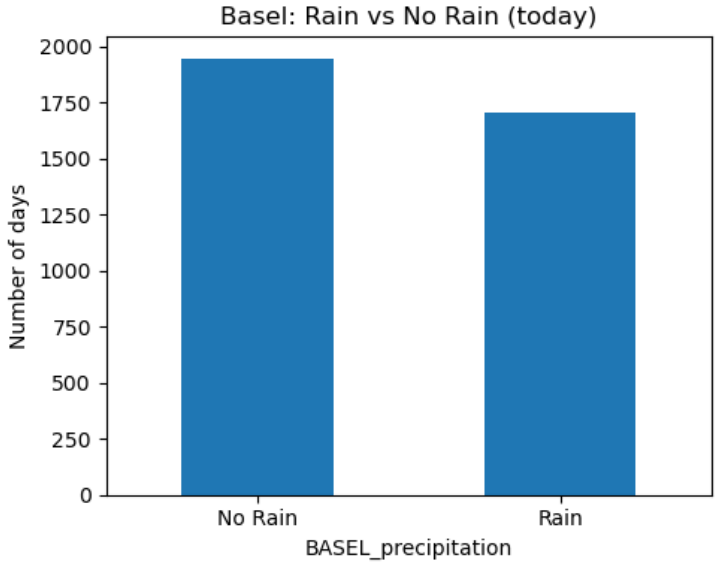

FIG. 1: Class balance for the derived Basel next-day rain label. About 53% of days are dry and 47% are rainy, so the task is only mildly imbalanced.

remaining gaps, the Basel subset used here (DATE, Basel variables, derived rain indicators, and one-day lags) contains no missing entries after the shift and lag operations apart from the first and last rows, which are dropped. No additional imputation is required.

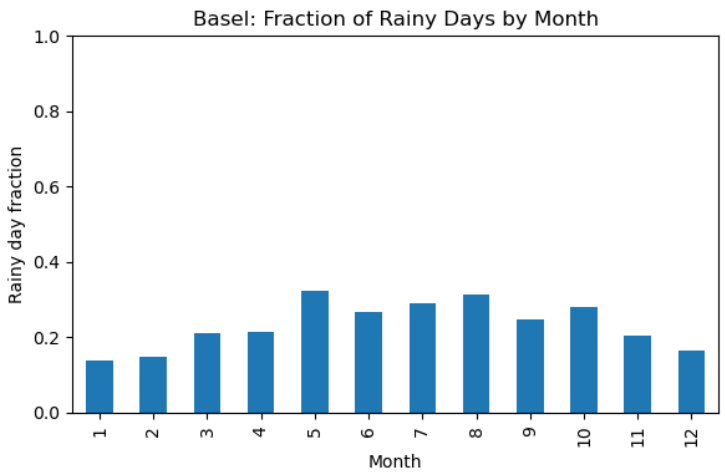

FIG. 2: Seasonal pattern of rainfall occurrence in Basel (2000–2010). Higher rain probabilities in late spring and summer justify adding calendar features such as MONTH.

Figure 1 shows the class balance for the derived next-day rain label: about 53% of days are dry and 47% are rainy, so the task is only mildly imbalanced. I therefore do not apply resampling, but for logistic regression I use `class_weight="balanced"` so that rainy days receive slightly higher weight during training.

Seasonality is visible in the fraction of rainy days by month (Figure 2): late spring and summer have a higher proportion of rainy days, while some winter months are drier. This motivates including MONTH as a simple seasonal feature. A multi-panel scatter plot of Basel precipitation versus key drivers (pressure, sunshine, humidity, maximum temperature; Figure 3) shows that low pressure and low sunshine are associated with more frequent and heavier rainfall, whereas high pressure and high sunshine correspond mainly to dry days. These diagnostics confirm that the chosen features have interpretable relationships with the target.

PREPROCESSING

The preprocessing pipeline respects the time-ordered nature of the data and constructs a clean Basel-only feature table for modeling. All transformations are applied in temporal order to avoid leaking information from future days into the past.

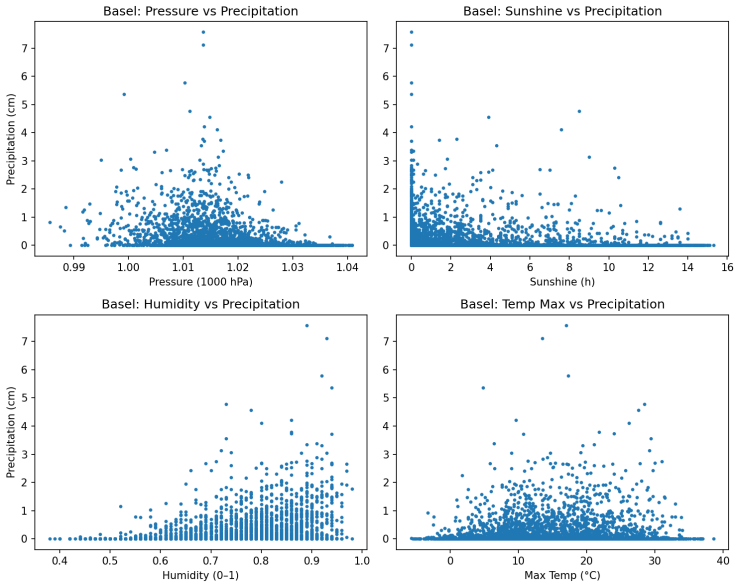

FIG. 3: Basel precipitation versus key drivers (pressure, sunshine, humidity, maximum temperature). Low pressure and low sunshine are clearly associated with rainy days, supporting the choice of these features.

Data Splitting

Daily records are first parsed and sorted by DATE. I then use a time-based split rather than random or stratified splitting. Specifically, the first 80% of days (2000–early 2008) are used for training and the last 20% (2008–2010) are held out for testing. This chronological split ensures that models are trained only on "past" data and evaluated on "future" data, mimicking a real forecast system.

The split is applied after constructing the RainToday and RainTomorrow labels and after adding lagged features, but before any model fitting or cross-validation. Logistic regression, random forest, and neural network models are tuned using only the training period (with internal cross-validation on that period), and their final performance is reported on the fixed test period.

Feature Engineering and Scaling

Feature engineering is deliberately simple and guided by basic meteorology. After restricting the multi-station file to Basel columns, I construct RainToday and RainTomorrow as above, extract MONTH from the DATE column, and create one-day lags of Basel pressure,

humidity, mean temperature, and sunshine. Rows that become undefined due to lagging (the first day) or shifting (the last day) are dropped. No interaction or high-degree polynomial terms are added.

All predictors are numerical. For logistic regression and the neural network, features are standardized using `StandardScaler` within a `Pipeline`. The random forest uses unscaled features because tree-based methods are invariant to monotonic transformations. No additional transformations (such as logs) are applied.

MACHINE LEARNING TASK AND OBJECTIVE

The objective is to predict whether it will rain on the following day in Basel using only information available at the local station at the end of the current day. Human forecasters and simple heuristic rules can exploit cues such as low pressure, high humidity, and recent rainfall, but assigning consistent weights to these cues across thousands of days is difficult to do by hand. NWP systems provide high-quality guidance but are expensive and hard to reproduce in a teaching setting. This project asks how far one can get with a compact, interpretable ML model applied directly to a single-station time series.

Task Type

The task is supervised binary classification at the daily time scale. Each sample corresponds to a single day at the Basel station. The input features are MONTH, RainToday, Basel surface pressure, relative humidity, mean temperature, sunshine, and one-day lags of the four Basel weather variables. The target `RainTomorrow` takes value 1 if there is measurable precipitation at Basel on the following day and 0 otherwise. Models are trained to map the feature vector at day t to a probability $P(\texttt{RainTomorrow}_t = 1 \mid \text{features}_t)$, which is then thresholded to obtain a rain/no-rain prediction.

Although the data form a time series, the modeling approach treats each day as an individual training example, subject to the time-respecting split described in Section . Short temporal dependence is represented through the lagged features. Within this framework I compare a trivial majority baseline, a tuned logistic regression model as an interpretable baseline, a random forest, and a shallow multilayer perceptron (MLP) neural network.

MODELS

The modeling stage compares a small set of supervised classifiers of increasing flexibility on the same Basel feature table and train/test split. All learned models output a probability of rain tomorrow, which is thresholded to produce a binary rain/no-rain prediction. Performance on the held-out 2008–2010 period is evaluated using accuracy and the rain-class precision, recall, and F1 score defined in Section . A trivial "always no rain" rule serves as a naive reference.

Model Families

Model 1: Regularized Logistic Regression

The simplest model is logistic regression with ℓ_2 regularization, implemented in scikit-learn and wrapped in a pipeline with feature standardization. After standardization the model fits a linear decision boundary in feature space and maps the standardized feature vector to a probability of next-day rain via the logistic link function. Logistic regression is chosen for its simplicity and interpretability and serves as the main interpretable baseline.

Model 2: Random Forest Classifier

The second model is a random forest classifier, an ensemble of decision trees trained on bootstrap resamples of the training data with random subsets of features considered at each split. This architecture can represent nonlinear relationships and interactions between current-day and lagged predictors without explicit feature transformations and is relatively robust to outliers. Here, the random forest is used as a flexible nonparametric benchmark and as a source of impurity-based feature-importance scores.

Model 3: Shallow Neural Network (MLP)

The most flexible model considered is a shallow feedforward neural network, implemented as an `MLPClassifier`. The inputs are the same standardized features used in the logistic regression pipeline. The MLP consists of one hidden layer with rectified linear unit (ReLU) ac-

tivations and a single sigmoid output node returning the probability of rain tomorrow. This architecture can represent smooth nonlinear decision boundaries and interactions among predictors while remaining small enough to train quickly in a teaching context.

TRAINING METHODOLOGY

All models are trained on the pre-2008 portion of the Basel dataset using the chronological 80/20 split described in Section . Hyperparameters are tuned only on the training period, and the final selected models are evaluated once on the held-out test period.

The logistic regression and neural-network models are trained as probabilistic classifiers using binary cross-entropy losses with ℓ_2 regularization. For logistic regression, the objective is a class-weighted binary cross-entropy that upweights rainy days via `class_weight="balanced"`, together with an ℓ_2 penalty on the coefficients controlled by the inverse regularization strength C. The MLP is trained on the same standardized inputs using binary cross-entropy with weight decay (the `alpha` parameter), optimized via stochastic gradient-based methods. The random forest grows each tree greedily using Gini-impurity splits and estimates the conditional rain probability by the fraction of trees voting for the rain class.

Hyperparameters are chosen via 5-fold cross-validation within the training period, using the rain-class F1 score as the selection metric.

For logistic regression, a `Pipeline` standardizes all features and then fits an ℓ_2-regularized model. A small grid search over $C \in \{0.1, 1.0, 3.0\}$ and `class_weight` $\in \{\text{None}, \text{"balanced"}\}$ is carried out; the best configuration uses $C = 3.0$ and `class_weight="balanced"`.

For the random forest, I fix 300 trees and a reproducible random seed and keep other hyperparameters near the scikit-learn defaults to balance flexibility and computational cost. Informal checks confirmed that increasing the number of trees beyond 300 has little effect on validation performance, so the forest is used primarily as a robust nonlinear benchmark.

For the neural network, I again use a `Pipeline` with standardization followed by an `MLPClassifier`. A small grid search explores hidden-layer sizes $(16,)$, $(32,)$, and $(32, 16)$ combined with weight-decay values $\alpha \in \{10^{-4}, 10^{-3}\}$. The best configuration is a single hidden layer with 16 units, $\alpha = 0.001$, and a maximum of 500 training iterations. The training loss for this configuration decreases smoothly and then plateaus, indicating stable convergence without obvious overfitting (Figure 6).

All probabilistic models output a continuous rain probability that is thresholded to obtain a binary decision. For the main comparison in Section , metrics are reported at the default threshold of 0.5; thresholds between 0.3 and 0.7 are also examined to illustrate the precision–recall trade-off.

Table I summarizes the main configurations and training objectives for the three models.

TABLE I: Summary of model configurations and training objectives.

Model	Key hyperparameters	Objective
Logistic regression	$C = 3.0$; balanced weights	weighted BCE + ℓ_2
Random forest	300 trees	tree ensemble; majority vote
MLP (neural net)	1 hidden layer (16); $\alpha = 10^{-3}$	BCE + ℓ_2

METRICS

Model performance is evaluated on the held-out 2008–2010 test period using classification metrics computed from the predicted rain/no-rain labels. Because the practical aim is to detect rainy days without ignoring the majority class of dry days, both class-specific and overall measures are reported. Unless otherwise noted, the rain class (`RainTomorrow` = 1) is treated as the positive class.

Primary Metric

The primary evaluation metric is the F1 score for the rain class. The F1 score is the harmonic mean of precision (the fraction of predicted rainy days that are actually rainy) and recall (the fraction of rainy days that are correctly identified). This choice reflects the mild class imbalance in the dataset (about 53% dry versus 47% rainy): a classifier that simply predicts the majority class can achieve reasonable accuracy while performing poorly on rainy days. By contrast, the rain-class F1 penalizes both missed rainy days and false alarms, and therefore provides a more informative summary of performance for this application. Hyperparameter tuning and model selection optimize this metric on the training folds.

Secondary Metrics

Accuracy, precision, recall, and the confusion matrix are used as secondary metrics to provide complementary information. Accuracy measures the overall fraction of correctly classified days and is useful for comparison with the trivial "always no rain" baseline. Precision for the rain class indicates how often a predicted rainy day is truly rainy and thus reflects the false-alarm rate. Recall for the rain class indicates how many of the rainy days are detected and thus reflects the missed-event rate. The confusion matrix summarizes these quantities in terms of the counts of correctly and incorrectly classified rainy and dry days, making the trade-off between rain and no-rain errors explicit.

Metric Definitions

For reference, let TP, FP, TN, and FN denote the numbers of true positives, false positives, true negatives, and false negatives for the rain class on the test set. The derived metrics are

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \quad \text{Precision}_{\text{rain}} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall}_{\text{rain}} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{1}$$

and

$$\text{F1}_{\text{rain}} = \frac{2\,\text{Precision}_{\text{rain}}\,\text{Recall}_{\text{rain}}}{\text{Precision}_{\text{rain}} + \text{Recall}_{\text{rain}}}. \tag{2}$$

These metrics are applied to the held-out 2008–2010 test period. The resulting values for accuracy, rain-class precision, rain-class recall, and rain-class F1 for each model are reported in Section .

RESULTS AND MODEL COMPARISON

This section reports the test performance of the baseline and three learned classifiers and compares their behavior. All metrics are computed on the fixed Basel test set described in Section , using the definitions in Section . As before, rainy days (`RainTomorrow` = 1) are treated as the positive class.

Performance Comparison

Table II summarizes test-set performance for the trivial majority baseline (always predict no rain), the simple baseline logistic regression using a small set of Basel predictors, the tuned logistic regression on the engineered feature set, the random forest, and the shallow neural network (MLP). Accuracy is computed over both classes, while precision, recall, and F1 are reported for the rain class.

TABLE II: Test-set performance on the Basel test period. Accuracy is overall; precision, recall, and F1 are for the rain class. Values are rounded to three decimal places.

Model	**Accuracy**	**Precision$_{\text{rain}}$**	**Recall$_{\text{rain}}$**	**F1$_{\text{rain}}$**
Majority baseline (always no rain)	0.521	0.000	0.000	0.000
Logistic regression (baseline, few features)	0.568	0.552	0.517	0.534
Logistic regression (tuned, full feature set)	0.668	0.640	0.697	0.668
Random forest	0.670	0.655	0.657	0.656
Neural network (MLP)	0.680	0.657	0.694	0.675

The majority baseline achieves 0.521 accuracy simply by predicting the dominant no-rain class, but it has zero precision, recall, and F1 for rainy days because it never predicts rain. This confirms that accuracy alone is not a sufficient criterion for this task. The simple logistic regression baseline on a small set of Basel predictors already improves substantially, reaching 0.568 accuracy and an F1 of 0.534 for the rain class, showing that even a low-dimensional linear model can extract signal from local pressure, humidity, temperature, and sunshine.

The tuned logistic regression on the engineered feature set, using feature scaling and balanced class weights, makes a further jump to 0.668 accuracy and a rain-class F1 of 0.668, with precision 0.640 and recall 0.697. This classifier detects a large fraction of rainy days while still correctly identifying many dry days and serves as a strong linear baseline.

The random forest yields very similar overall performance (0.670 accuracy and F1 = 0.656). Its precision (0.655) and recall (0.657) are both slightly lower than those of the tuned logistic regression. Feature-importance scores highlight current and lagged pressure, mean temperature, and sunshine as the most influential predictors, consistent with earlier

physical intuition.

The neural network (MLP) attains the highest scores among the models tested, with 0.680 accuracy and a rain-class F1 of 0.675. Its precision and recall for rainy days (0.657 and 0.694, respectively) are close to those of the tuned logistic regression but yield a small net gain in F1. The improvement over the tuned logistic model is modest, suggesting that most of the structure in this station-level feature set can be captured by a well-regularized linear classifier.

Computational Efficiency and Final Choice

All three learned models are computationally inexpensive on this dataset of 3,653 daily samples and 10 numerical predictors. The tuned logistic regression is the fastest to train and evaluate. The random forest requires more computation because 300 trees must be grown, but training still completes in seconds on a single CPU. The MLP has the highest per-iteration cost, but with a single small hidden layer its total training time remains practical.

Taking predictive performance, interpretability, and computational cost together, I treat the tuned logistic regression as the *primary final model* for the Basel rain prediction task. The random forest and MLP confirm that there is no large untapped nonlinear gain and mainly serve as nonlinear checks that the chosen linear model is already close to the achievable performance. By the primary metric (rain-class F1), the MLP is technically the best-performing model (F1 = 0.675), but the gain over the tuned logistic regression (F1 = 0.668) is very small, while the logistic model remains much easier to interpret and explain.

MODEL INTERPRETATION

The interpretation focuses on the tuned logistic regression, treated as the primary final model, with the random forest and neural network providing nonlinear checks. For the neural network, the training loss curve in Figure 6 decreases smoothly and then flattens, with no signs of divergence or large oscillations. Together, these models help identify which aspects of the Basel time series are most informative for next-day rain and how the predicted probabilities behave.

Feature Importance

For the logistic regression, inspecting the fitted coefficients (not shown) confirms that the signs match basic meteorological expectations. Lower surface pressure and lower sunshine are associated with higher predicted rain probability, while higher humidity and a positive `RainToday` flag also increase the odds of rain tomorrow. The lagged versions of pressure, mean temperature, and sunshine enter with similar signs, indicating that persistent low pressure and cloudy conditions are more predictive than a single-day anomaly.

The random forest feature-importances provide a complementary nonlinear view. Table III lists the top predictors ranked by mean decrease in impurity. Current and lagged surface pressure are the strongest drivers, followed by mean temperature and sunshine (again with both same-day and lagged values), and then recent rain and humidity. Month enters with smaller but non-negligible importance, reflecting the seasonal modulation of rain probability seen in the exploratory analysis.

TABLE III: Top random-forest feature importances for next-day rain prediction.

Feature	**Importance**
BASEL_pressure	0.174
BASEL_pressure_lag1	0.129
BASEL_temp_mean	0.104
BASEL_temp_mean_lag1	0.102
BASEL_sunshine	0.094
RainToday	0.091
BASEL_humidity_lag1	0.086
BASEL_sunshine_lag1	0.085
BASEL_humidity	0.083
MONTH	0.051

The close agreement between the logistic coefficients and the random-forest importance ranking supports the conclusion that the engineered Basel features are physically meaningful: pressure and sunshine dominate, with temperature, humidity, and recent rain providing additional context, and seasonality acting as a weaker but consistent background signal.

Model Behavior Analysis

Beyond global feature importance, it is useful to understand how the final model's predicted probabilities behave. Figure 4 shows a reliability diagram for the tuned logistic regression on the test set. The curve lies close to the 1:1 line over most of the probability range, with only moderate deviations at the lowest and highest bins. This indicates that when the model assigns, for example, a 70% probability of rain, roughly 70% of those days are indeed rainy in the held-out period. In other words, the logistic model is reasonably well calibrated as a probabilistic forecaster, not only as a classifier.

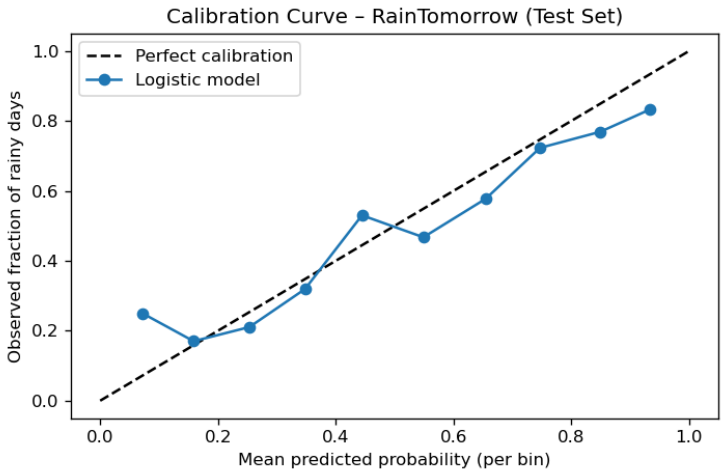

FIG. 4: Calibration curve for the tuned logistic regression on the test set. The dashed line indicates perfect calibration. The logistic curve lies close to the diagonal over most probability bins, indicating reasonable reliability.

Figure 5 plots the distribution of predicted rain probabilities separately for truly dry and truly rainy days on the test set. Dry days cluster mainly in the 0.2–0.4 range, while rainy days concentrate between about 0.6 and 0.8, but there is substantial overlap in the intermediate region. The vertical line marks the default 0.5 decision threshold used for the headline metrics. This visualization makes the trade-off between precision and recall concrete: lowering the threshold would move more of the ambiguous 0.4–0.5 days into the "rain" category, increasing recall at the cost of more false alarms, while raising it would do the opposite.

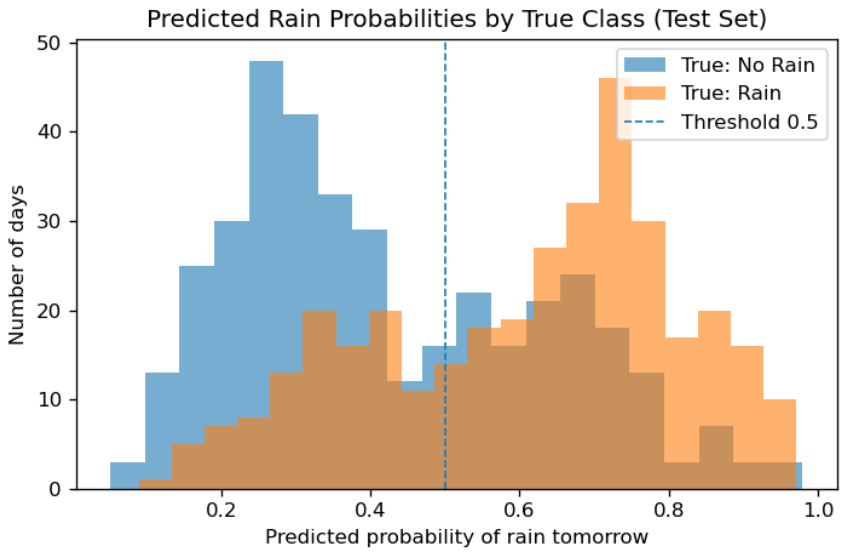

FIG. 5: Distribution of predicted rain probabilities from the tuned logistic regression on the test set, conditioned on the true class. The dashed line marks the default decision threshold of 0.5.

For the neural network, the training loss curve (Figure 6) decreases smoothly and then flattens, with no signs of divergence or large oscillations. This suggests that, despite the convergence warnings in scikit-learn, the network is approaching a stable minimum and is not obviously overfitting the limited dataset. Together with the similar test metrics reported in Section , this supports the view that the MLP is capturing essentially the same patterns as the tuned logistic regression, but with only a modest gain in flexibility.

CONCLUSION

Summary of Findings

This project framed "Will it rain tomorrow in Basel?" as a supervised binary-classification problem on a station-only time series derived from the ECA&D archive. After constructing a clean Basel feature table with calendar information, same-day weather variables, and one-day lags, I trained and evaluated several models using a chronological 80/20 train/test split. The majority-class baseline, which always predicts no rain, achieved 0.521 accuracy but an F1 score of 0.0 for rainy days. A simple logistic regression on a small set of Basel variables improved performance to 0.568 accuracy and an F1 of 0.534 for the rain class. The

tuned logistic regression with scaling, balanced class weights, and the engineered feature set further increased performance to 0.668 accuracy and a rain-class F1 of 0.668. The random forest reached a similar level, and the neural network (MLP) achieved the best raw test performance with 0.680 accuracy and F1 = 0.675 for rainy days. Thus, **if "best" is defined purely by the primary metric (rain-class F1), the neural network (MLP) is the best-performing model on this task**. At the same time, the tuned logistic regression is only slightly behind and remains far more interpretable. For the purposes of this project, I therefore treat the tuned logistic regression as the preferred final model: it captures almost all of the available predictive signal while providing coefficients that can be directly linked to physical understanding of the weather.

Limitations and Future Work

The analysis is restricted to a single station and a single decade of data, so the conclusions may not generalize to other locations or longer time spans. The feature set is intentionally modest: it includes only same-day values and one-day lags for a small number of surface variables. Longer memory, multi-day summary features, or explicit sequence models might capture additional temporal structure. The rain label is defined as any measurable precipitation; models optimized for specific impact-relevant thresholds (e.g., moderate or heavy rain) might behave differently. All classifiers use a fixed decision threshold of 0.5 for the main comparison, whereas in operational settings thresholds are often tuned to reflect asymmetric costs of false alarms and missed events. More systematic threshold selection, or direct optimization of cost-sensitive objectives, would make the forecasts more application-specific. Finally, the models rely solely on local station data and do not incorporate spatial information or NWP guidance, which are central to modern operational systems. Future work could therefore extend this project by incorporating additional lags and multi-day summaries, testing the models on other stations in the ECA&D archive, or augmenting the predictors with coarse-resolution reanalysis fields. It would also be natural to explore time-series cross-validation, cost-based threshold tuning, and alternative evaluation metrics such as the Brier score and reliability diagrams across multiple lead times, as well as uncertainty quantification via bootstrapping or Bayesian logistic regression to assess the robustness of the reported coefficients and scores.

Final Remarks

Despite these limitations, the project demonstrates that a relatively simple pipeline—careful preprocessing, physics-guided feature engineering, and a regularized logistic regression—can deliver a calibrated and interpretable next-day rain/no-rain forecast for Basel with a rain-class F1 around 0.67. Nonlinear models such as random forests and shallow neural networks yield only incremental gains, suggesting that most of the structure in this single-station feature set is effectively linear. In a teaching context, this provides a useful case study of how modest machine-learning tools, combined with domain knowledge, can turn a standard climate archive into a practical probabilistic forecasting system. In summary, the MLP delivers the highest test F1 score, but the tuned logistic regression offers a more attractive balance of performance, interpretability, and simplicity, and is therefore the model I would recommend for practical day-ahead rain/no-rain prediction in Basel.

I would like to thank Prof. Luciano Germano Silvestri, instructor of CMSE 492, for designing the course and providing the framework within which this project was carried out. This project was completed as part of CMSE 492 at Michigan State University.

---

* gargpurv@msu.edu

[1] Klein Tank, A. M. G., et al., "Daily dataset of 20th-century surface air temperature and precipitation series for the European Climate Assessment," *International Journal of Climatology* **22**, 1441–1453 (2002).

[2] Kaggle, "Weather Prediction Dataset," https://www.kaggle.com/, accessed 2024.

[3] Pedregosa, F., et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research* **12**, 2825–2830 (2011).

[4] Breiman, L., "Random forests," *Machine Learning* **45**(1), 5–32 (2001).

[5] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., O'Reilly Media, Sebastopol, CA (2019).

[6] OpenAI, "ChatGPT (GPT-5.1)," large language model used for grammar and writing refinement, https://chat.openai.com (accessed Nov. 23, 2025).

Additional Figures

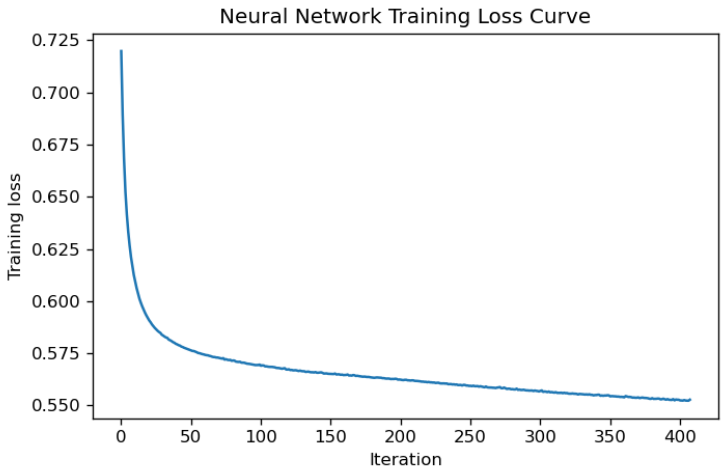

FIG. 6: Training loss curve for the MLP classifier on the Basel training set. The loss decreases smoothly and then plateaus, indicating stable convergence without obvious overfitting.

Code Availability

The complete code for this project is available at: https://github.com/purvigarg2004-design/cmse492_project