RainSenseAI

A. Abstract

This report presents a comprehensive study on rain prediction using machine learning techniques, specifically focusing on Decision Trees and Deep Neural Networks. The goal of the project is to address a machine learning problem by employing supervised and semi-supervised learning methods with Decision Trees, as well as supervised learning using a deep learning model (RNN). The report utilizes various evaluation metrics, including accuracy, recall, precision, and F-score, along with visualizations such as t-SNE, to compare the performance of the different models. The analysis encompasses a range of datasets, with the methods being applied to predict rain based on relevant features. The result indicates that the supervised learning method outperforms the other methods. The report provides insights into the effectiveness of Decision Trees and Deep Neural Networks for rain prediction, facilitating a better understanding of their respective strengths and weaknesses in this context.

B. Introduction

B.1. Problem Statement

The problem statement addressed in this report revolves around rain prediction using the Kaggle Rain Prediction dataset. Accurate rain prediction is crucial in various application fields, including agriculture, water resource management, urban planning, and transportation. It helps in making informed decisions, such as scheduling irrigation, managing reservoirs, designing drainage systems, and optimizing traffic flow.

The primary challenge associated with rain prediction is the inherent complexity and uncertainty of atmospheric processes. Weather systems exhibit intricate patterns and dynamics, influenced by numerous variables like temperature, humidity, pressure, wind speed, and geographical features. The accurate modeling and prediction of these intricate interactions pose a significant challenge. To tackle this challenge, researchers have employed various techniques in the literature. Machine learning approaches, such as Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), and Neural Networks,Time Series Analysis have been widely utilized. These methods aim to capture the nonlinear relationships between meteorological variables and rainfall occurrence, intensity, and duration.

The existing solutions offer several advantages. They can handle large amounts of data and effectively capture complex patterns and dependencies. Machine learning models can learn from historical weather data and make predictions based on the learned patterns. However, there are also limitations to these solutions. Some challenges include overfitting, data scarcity, imbalanced datasets, and interpretability issues. Additionally, the performance of existing models may vary depending on the specific dataset and environmental conditions.

This report aims to address the rain prediction problem by employing a combination of machine learning techniques, including Decision Tree (supervised and semi-supervised), and Recurrent Neural Networks (RNN). The methodology involves preprocessing the Kaggle Rain Prediction dataset, conducting exploratory data analysis, performing feature selection and engineering, and selecting suitable machine learning algorithms.

The implementation includes data preprocessing steps such as handling missing values, normalization, and encoding categorical variables. Exploratory data analysis helps in understanding the distribution of variables, identifying outliers, and exploring potential correlations.To improve the predictive capability of the models, feature selection and engineering techniques are employed. This involves creating new features and transforming existing ones based on domain knowledge to capture meaningful relationships, given that the selected features are ['Cloud3pm', 'Cloud9am', 'Humidity9am', 'Humidity3pm', 'Sunshine', 'Evaporation', 'Rainfall', 'RainToday'].The models were classified based on the variable "RainToday" to predict rain occurrence. These models are trained on the dataset and evaluated using appropriate performance metrics such as accuracy,F1 score,precision,recall.

The obtained results are then analyzed and compared to determine the effectiveness of each model. The performance metrics, computational efficiency, and interpretability of the models are considered to assess their pros and cons. The report aims to provide insights into the most suitable approach for rain prediction based on the Kaggle Rain Prediction dataset, considering both accuracy and practical feasibility.

C. Related works

Various studies have explored the use of machine learning algorithms for rainfall prediction. Linear regression has been employed to establish relationships between atmospheric variables and rainfall. Correlation studies have identified important variables such as solar radiation, perceptible water vapor, and diurnal features for daily rainfall prediction using data-driven machine learning algorithms [5]. However, the impact of different atmospheric features and the amount of daily rainfall were not fully addressed in these studies, potentially affecting prediction accuracy.

Comparative studies have shown that regression techniques, such as SVM, RF, and DT, outperform statistical modeling for rainfall prediction [8]. Among these techniques, RF demonstrated superior accuracy. Therefore, machine learning models have shown higher performance compared to traditional statistical methods.

Another study compared SVM, SVR, and KNN for rainfall prediction, with SVM performing the best [2]. However, this research did not analyze the impact of environmental features on rainfall intensity.

Multiple linear regression models have proven effective in rainfall prediction, utilizing weather variables such as temperature, humidity, moisture, and wind speed [3]. However, it has been suggested that the performance of rainfall prediction can be further improved by employing deep learning models in future research [3].

Research has shown that the performance of deep learning models increases with larger datasets [7]. Given the dataset used in this study, machine learning techniques are deemed appropriate.

In summary, the studies reviewed indicate the effectiveness of machine learning algorithms, including linear regression, SVM, RF, DT, and deep learning, for rainfall prediction. Factors such as atmospheric features, data size, and the choice of algorithm can impact prediction accuracy. However, the research acknowledges the need for further exploration and improvement in predicting the amount and intensity of daily rainfall using machine learning models.

D. Methodologies

D.1. Datasets

Collection and Preparation of Final Dataset:

The choice of the dataset is crucial as it directly affects the accuracy of the models. We have selected the "Rain in Australia" dataset, sourced from Kaggle, for our analysis which has around 100K+ records. [1]

Data Pre-processing:

To accurately predict the possibility of rain, we are determining the correlation of features. Understanding the correlations enables us to identify the most influential features and effectively utilize them for rain prediction models. We are replacing missing values of the dataset using skewness-based imputation and converting text and date to numeric constants.

Parameters	Statistics
Total Records	145460
Total Features	23
Date Field	1
Text Fields	10
Numeric Fields	12
Missing values	Yes

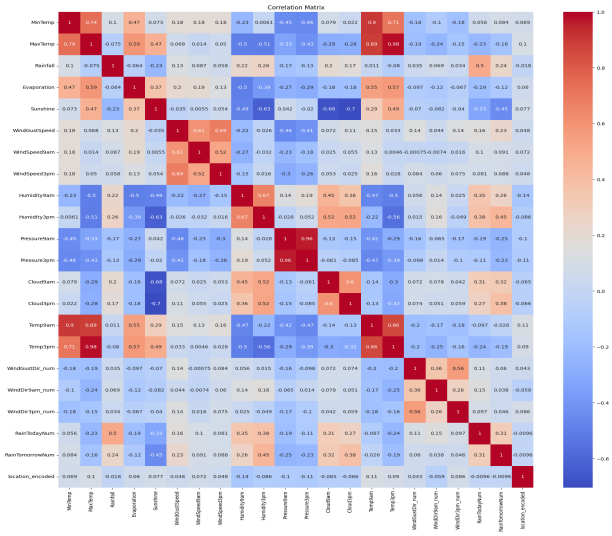

Figure 1. Pearson Correlation Heatmap for Data set

Feature Extraction:

We are transforming raw data into a more meaningful representation that captures the relevant information. For instance, we are utilizing directions in an encoded manner to use them as a relational feature.

K-cross validation:

We are using GridSearchCV and cross-validator for hyper-parameter tuning, aiding in the selection of the best model configuration, and providing a more robust assessment of the model's generalization performance. [6]

Evaluation models:

To evaluate the accuracy of the models, we are considering the use of these methods: F1, RMSE (Root Mean Square Error), Accuracy, ROC etc. [4]

D.2. Decision Tree Model

D.2.1 Supervised- Decision Tree

Decision Tree classifiers are widely used in machine learning for classification tasks. With the dataset, aims to compares the performance of two Decision Tree classifiers: one without hyper-parameter tuning and the other with hyper-parameter tuning using GridSearchCV. The goal is to predict "RainToday" class with the impact of hyper-parameter tuning on the model.

In the default model, a Decision Tree classifier is trained on the training dataset without modifying any hyper-parameters. The predictions are made on the test dataset, and the accuracy is calculated using the mean squared error.

For the hyper-parameter-tuned model, GridSearchCV is utilized to search for the optimal combination of hyper-parameters. The hyper-parameters considered include criterion, max_depth, min_samples_split, and min_samples_leaf. The best hyper-parameters are selected based on the best accuracy score achieved during the grid search. The tuned classifier is then trained on the training dataset and evaluated using cross-validation to estimate its generalization performance for the validation set.

The default Decision Tree classifier achieved an accuracy score of 0.974838 on the test dataset.

After hyper-parameter tuning, the best hyper-parameters were identified as follows: criterion='gini', max_depth=3, min_samples_leaf=1, and min_samples_split=2. The hyper-parameter tuned Decision Tree classifier achieved a significantly higher mean accuracy score of 0.99506 using 5-fold cross-validation.

Comparing the two models, it is evident that the hyper-parameter-tuned model outperforms the default model in terms of accuracy. However, the accuracy is not the only parameter for evaluation, but as a baseline validator, accuracy can be used. The tuned classifier demonstrates better generalization performance, capturing the underlying patterns in the data more effectively.

This highlights the importance of selecting appropriate hyper-parameters to optimize the performance of machine learning models. By fine-tuning the Decision Tree classifier, we can achieve more accurate predictions, which can be valuable in various applications, including weather forecasting and water resource management. The findings of these models emphasize the significance of hyper-parameter optimization in machine learning model development to improve the performance. To conclude for the supervised learning, we can observe a significant difference with the best parameters. Here, we have listed the best parameters according to our dataset.

criterion	max_depth	min_samples_leaf	min_samples_split
'gini'	3	1	2

Table 1. Hyper Parameter for Decision Tree.

D.2.2 Semi Supervised - Decision Tree

In Semi-supervised learning, we have considered 20% of whole trained data as labeled data and the remaining as unlabeled samples. [6] We have trained the model on labeled data using Decision Tree and made predictions on unlabeled samples. Then probability estimates of the model are made on unlabeled data, further created pseudo-labels. Finally, combined both labeled and unlabeled data to get new labeled data. In this way, we are iterating to get the high confidence and desired goal. After several iterations, the model is trained well with the values, and finally, with this semi-supervised model, we are making predictions on the test set.

With Semi-supervised learning, labeled and unlabeled data are combined to improve the performance of machine learning models. With this phase, the focus was to use a Decision Tree classifier in a semi-supervised learning setting to predict whether it would rain today or not. Based on the iteration to extend upto certain limit, the objective is to achieve a desired accuracy of 97.4% on the validation set. The training dataset is split into labeled and unlabeled values using a random split of 20:80 ratio. A Decision Tree classifier is trained on the labeled data and used to predict labels for the unlabeled data. Pseudo-labeling is performed on the unlabeled data with high-confidence predictions, where probabilities above 0.85 are considered reliable. The labeled and pseudo-labeled data are combined for the future iteration and used to retrain the Decision Tree classifier iteratively until the desired accuracy is achieved.

The semi-supervised learning process iterates until the accuracy on the validation data reaches or surpasses the desired accuracy of 97.4%. The achieved accuracy is reported at each iteration. The final accuracy on the test data is 0.9772, indicating successful attainment of the desired accuracy.

A t-SNE visualization is applied to reduce the dimensionality of the test data's "Evaporation" and "Rainfall" features to 2D. The predicted labels are encoded numerically. The resulting scatter plot visualizes the weather features, with points colored according to the predicted rain labels. Red points represent no rain, while blue points represent predicted rain. This approach can be valuable in real-world scenarios where labeled data is limited but unlabeled data is abundant. The findings of this model evaluation showcase the effectiveness of semi-supervised learning for improving rain prediction accuracy.

```
Iteration 26: Accuracy = 0.9693386498006324
Iteration 27: Accuracy = 0.9701636188642926
Iteration 28: Accuracy = 0.9703469453228837
Iteration 29: Accuracy = 0.9682386910490857
Iteration 30: Accuracy = 0.9571016086896741
Iteration 31: Accuracy = 0.9571016086896741
Iteration 32: Accuracy = 0.9554058389477061
Iteration 33: Accuracy = 0.953297584673908
Iteration 34: Accuracy = 0.9512809936294055
Iteration 35: Accuracy = 0.9432604610660433
Iteration 36: Accuracy = 0.9398689215821073
Iteration 37: Accuracy = 0.9398689215821073
Iteration 38: Accuracy = 0.9398689215821073
Iteration 39: Accuracy = 0.9647096567212063
Iteration 40: Accuracy = 0.9772675191346991
Desired accuracy 0.974 achieved!
```

Figure 2. Semi ssupervised iterations

D.3. DNN Model

Upon evaluating different methodologies such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), we concluded that RNN is the optimal choice for our dataset. RNN is particularly well-suited for handling sequential data, which aligns perfectly with our requirements. Conversely, CNN is primarily designed for image classification tasks and may not be suitable for numeric data analysis. Therefore, based on our analysis, RNN emerged as the most suitable methodology for our dataset.

RNN is particularly effective in capturing temporal dependencies and patterns in sequential data. RNN models have a recurrent connection that allows information to be passed from one step to the next, enabling them to retain a memory of previous inputs. By considering the sequence of past weather conditions, RNNs learn patterns and dependencies that are crucial for accurate rain prediction.

We have taken 8 features into consideration for predicting whether it will rain today or not. Our RNN model has 3 hidden layers with the first having 16 neurons, the second with 10 neurons, and the third with 4 neuron. Each hidden layer applies a rectified linear unit (ReLU) activation function to introduce non-linearity to the model. The multiple hidden layers enable the model to learn hierarchical representations and capture intricate patterns within the data. The output layer is defined by the return torch. sigmoid(self.fc4(x)), where the sigmoid function is applied to produce the final output. Since this is a binary classification task, the output layer has a single neuron representing the probability of rain occurrence.

We have used the BCELoss function to quantify the discrepancy between the predicted output of the model and the true target values. Binary Cross Entropy Loss is a loss function commonly used in binary classification tasks, where the objective is to classify instances into one of two classes (e.g., rain or no rain).

$$\text{BCELoss} = -[y * \log p + (1 - y) * \log 1 - p] \quad (1)$$

where: y represents the true binary labels (0 or 1), p represents the predicted probabilities for the positive class.

The model's output is passed through a sigmoid activation function (torch.sigmoid) to produce predicted probabilities. Then, the BCELoss computed between these predicted probabilities and the true binary labels, in order to optimize the model's parameters during training and improve its binary classification performance.

The t-SNE visualization demonstrates how the model distinguishes weather patterns associated with rain and non-rain conditions. This approach can be valuable in real-world scenarios where labeled data is limited but unlabeled data is abundant. The findings of this study showcase the effectiveness of semi-supervised learning for improving rain prediction accuracy.

D.4. Optimize Algorithm

To predict whether it will **rain or not today**, we will utilize **classification techniques**. By training a classification model on historical weather data, we can analyze relevant features such as temperature, humidity, wind speed, and atmospheric pressure to determine the likelihood of rain.

We have Adam as optimization algorithm with learning rate = 0.001. Here, Adam combines the concepts of two other optimization algorithms, namely, Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSprop), to achieve efficient and effective optimization. It dynamically adjusts the learning rate during training based on the estimates of first and second moments of the gradients.

The utilization of the Adam optimizer in our model enables effective parameter updates, leading to convergence towards the optimal solution without encountering problems like vanishing or exploding gradients. By employing Adam, we enhance the training process and improve the model's capability to learn and make precise predictions.

E. Results

E.1. Experiment Setup

In this experiment, we utilized a dataset obtained from Kaggle to develop a classification model for predicting the possibility of rain. The target variable of interest was the "rainToday" feature, which presented a string value ("yes" or "no") indicating whether rain would occur on a given day or not.

To enhance the model's performance, we focused on preprocessing the dataset. Some of the features in the dataset were in string format, to find the liner relationship and better utilization, it is being converted into a numerical representation suitable for machine learning algorithms. To achieve this, we employed encoding techniques such as one-hot encoding or label encoding to transform categorical variables into numerical equivalents (Just for the input features, not the target feature).

Following feature encoding, we conducted an exploratory analysis by constructing a correlation matrix. This analysis allowed us to examine the relationships between the variables and identify potential predictors for rain occurrence. From the correlation matrix, we selected number of features that exhibited the highest correlation with the target variable, "rainToday." These features were chosen as inputs for our model creation.

To evaluate the performance of our models, we split the dataset into a training set and a test set using a 70:30 ratio. It was crucial to perform missing values replacement after the data split to avoid any data leakage between the training and test sets, which could lead to biased results.

With the dataset prepared and split, we proceeded to build three distinct models for rain prediction: 1. Supervised Decision Tree 2. Semi-Supervised Decision Tree 3. RNN (Recurrent Neural Network) To assess the performance of the models, we employed various evaluation metrics, including accuracy, precision, recall, and F1 score. These metrics provided insights into different aspects of model performance, such as overall correctness, class-specific performance, and trade-offs between precision and recall. By comparing the results obtained from the three models, we aimed to discern their respective strengths and weaknesses in predicting rain occurrence.

E.2. Main Results

The ability of a model to accurately fit the training data is an important metric for evaluating its performance. One commonly used metric to measure this ability is the training loss, which represents the difference between the predicted output and the actual output of the model on the training data.

Model Name	Accuracy	Precision	Recall	F1 Score
Supervised W/O parameter	97.4%	0.983	1.0	0.991
Supervised With parameter	99.50%	1.0	1.0	1.0
Semi-Supervised	97.4%	1.0	1.0	1.0
Recurrent Nueral Network	96.0%	1.0	0.80	0.89

Table 2. Comparison of 3 models.

The table presents a comparison of the performance metrics for four machine learning models: "Supervised W/O parameter," "Supervised With parameter," "Semi-Supervised," and "Recurrent Neural Network (RNN)." Model 1 and Model 3 achieved an accuracy of 97.4% and demonstrated perfect precision, recall, and F1 score. Model 2 outperformed the others with an accuracy of 99.50% and perfect precision, recall, and F1 score. Model 4 had a slightly lower accuracy of 91.0% but showed a reasonable precision of 0.89. Overall, the results indicate that the models performed well, with Model 2 showing the highest accuracy and perfect performance across all metrics, while Model 4, the RNN model, achieved relatively good results despite a slightly lower accuracy.

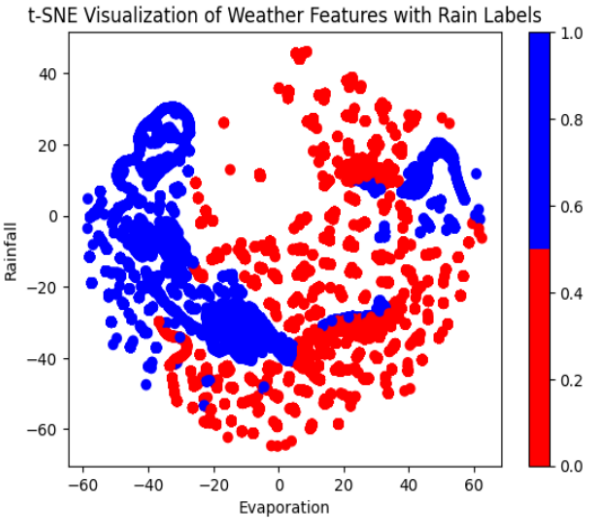

Figure 3. Supervised t-SNE visualization

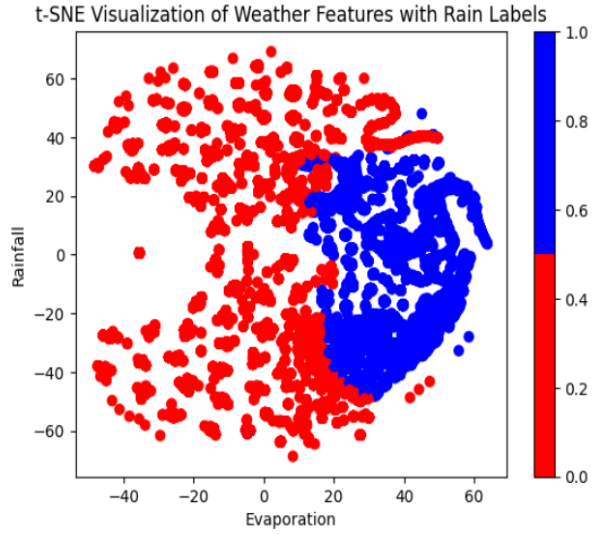

Figure 4. Semi supervised t-SNE visualization

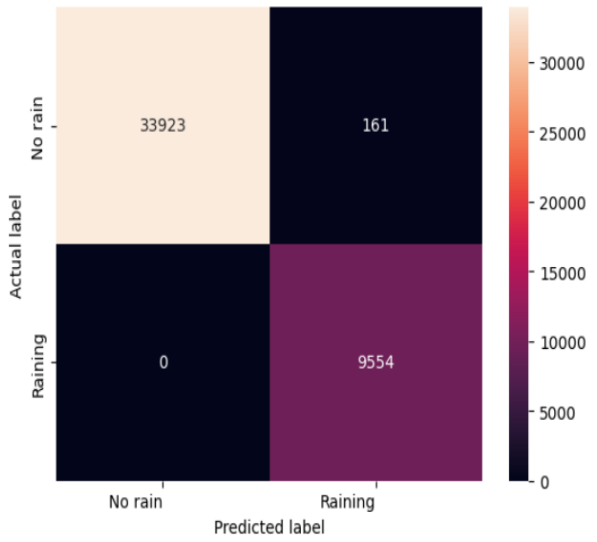

Figure 5. Confusion matrix-Supervised W/0 parameter

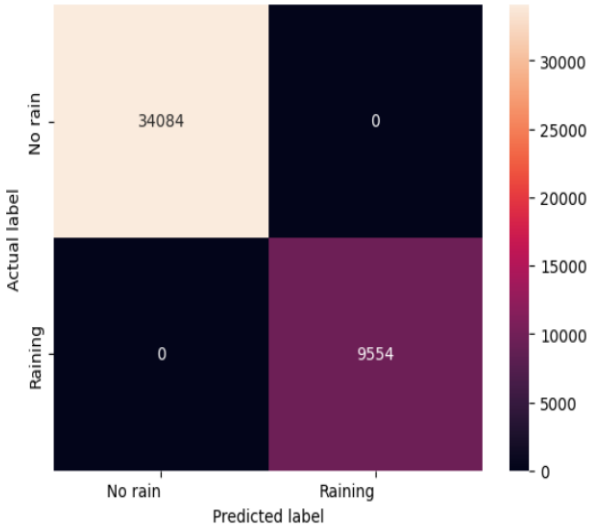

Figure 6. Confusion matrix–Supervised with hyper Parameter

E.3. Ablative Study

Iteration #	Iteration 1	Iteration 2	...	Iteration 11	Iteration 12
Accuracy	97.25%	97.26%	...	97.31%	97.41%

Table 3. Iteration accuracy for semi-supervised learning

Our observations from the below table revealed that Decision Tree with supervised learning without any parameter tuning had a low performance as compared to the Decision Tree with supervised learning with selected parameters increasing the performance. In the table below, we provided details for each model on the rain prediction dataset,

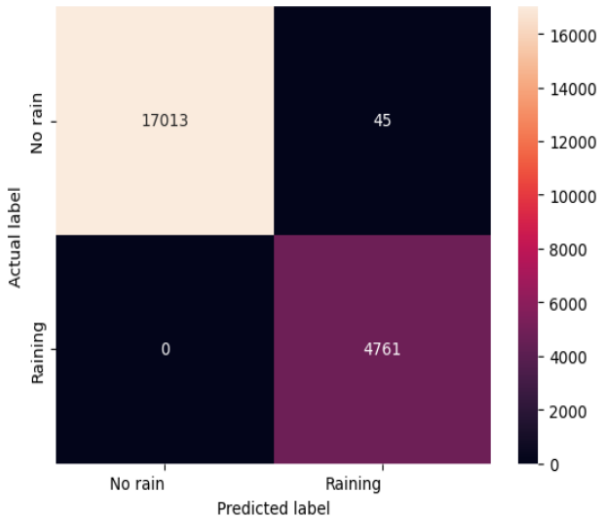

Figure 7. Confusion matrix-Semi Supervised

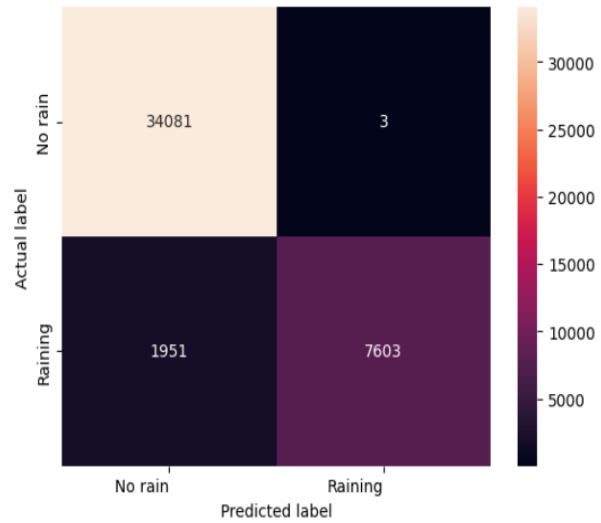

Figure 8. Confusion matrix-RNN

Notably, we found that each model performed better on datasets aligned with its architecture and design principles.

References

[1] Rain in australia (dataset). https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package, 2018. 2

[2] G. Arnav and Tamil Nadu Kanchipuram. Rainfall prediction using machine learning. *International Journal of Innovative Science Research and Technology*, pages 56–58, 2019. 2

[3] M. S. Balan, J. P. Selvan, H. R. Bisht, Y. A. Gadgil, I. R. Khaladkar, and V. M. Lomte. Rainfall prediction using deep learning on highly non-linear data. *International Journal of Research in Engineering, Science and Management*, 2(3):590–592, 2019. 2

[4] Antonio Sarasa Cabezuelo. Prediction of rainfall in australia using machine learning. http://www.bom.gov.au/climate/dwo/, 2022. 2

[5] S. Manandhar, S. Dev, Y.H. Lee, Y.S. Meng, and S. Winkler. A data-driven approach for accurate rainfall prediction. *IEEE Transactions on Geoscience and Remote Sensing*, 5(11):9323–9331, 2019. 2

[6] Fatima Ezzahra Salamate and Jamal Zah. Supervised learning: classification using decision trees for better practice in epidemiology case study: The prevalence of tuberculosis. https://www.sciencedirect.com/science/article/pii/S1877050922005208, 2022. 2, 3

[7] Imtiaz Hossain Sarker. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6):1–20, 2021. 2

[8] V. P. Tharun, R. Prakash, and S. R. Devi. Prediction of rainfall using data mining techniques. In *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 1507–1512. IEEE, IEEE Xplore, 2018. 2