Rainfall Prediction using Machine Learning

Shabeg Singh Gill
IIIT, Delhi
shabeg19388@iiitd.ac.in

Hardik Garg
IIIT, Delhi
hardik19040@iiitd.ac.in

Aniket Verma
IIIT, Delhi
aniket19233@iiitd.ac.in

Rahul Sethi
IIIT, Delhi
rahul19266@iiitd.ac.in

November 25, 2021

Abstract

Water is an essential part of our lives. Rainfall plays an important role at the micro level by fulfilling our survival needs and at the macro level as a part of a country's GDP. Rainfall can also help prevent natural disasters such as forest fires.

Rainfall Prediction is one of the necessary techniques to predict the climate of any country. In this paper, we present a comparison of different ML models to predict whether it would rain tomorrow or not, given features such as temperature, pressure, humidity and sunlight for the current day.

Github- https://github.com/RahulSethi070801/ML_Project

1 Introduction

There are a plethora of rainfall issues around the globe, floods are an important issue leading to the devastation of life and property. Saving human life and property becomes a challenging task at the time of floods and heavy rainfall. Unsuccessful rainfall predictions can lead to huge losses, hence, it becomes a challenging problem to solve. Lack of rainfall can also lead to forest fire due to dry climates which in turn leads to huge loss of life and property.

In this project, we work on the Australia Rain dataset from kaggle ([1]) and predict whether there will be rainfall tomorrow based on today's weather parameters. A case study of different ML models is presented along with reasoning, explaining the differences in the working of these models.

2 Literature Survey

2.1 Rainfall Prediction using ML and NN

2.1.1 Methodology

In [2], they used two techniques to predict the rainfall. The first approach is LASSO regression and the second approach is Neural Network approach(ANN). This system compares both the models and then accordingly gives the result with the best algorithm. Prediction is not done on both the algorithms to reduce complexity. Firstly, the system finds the most accurate algorithm between machine learning and neural network by training on both the algorithms and accordingly does the prediction on the algorithm with better accuracy.

The accuracy is presented in the form of various metrics and the result is obtained in the form of graphs. After performing the comparison, the conclusion of the system is that LASSO regression process is more accurate than Artificial Neural Network process.

2.1.2 Limitation

There was a high percentage of error while calculating the accuracy through different methods of evaluation(metrics). This system is very complex which makes the computational costs very high.

2.2 Prediction of Rainfall Using Machine Learning Techniques

2.2.1 Methodology

In [3], methodologies such as Artificial Neural Network are used. Extraction procedures/algorithms produce the output by classification of the data according to the categories.

In this method, multiple regressions are used to predict the values with the help of descriptive variables and is a statistical method. It is having a linear relationship between the descriptive variables and the output values. The number of observations is indicated by n. Multiple meteorological parameters are necessary to predict the rain fall.

The proposed method predicts the rainfall for the Indian rainfall dataset using multiple linear regressions and provides improved results in terms of accuracy, MSE and correlation.

2.2.2 Limitation

Assumptions made by the multiple linear regressions are that there is a linear relationship between both the descriptive and the independent variables. Also, the highly correlated variables are considered independent variables.

3 Dataset

3.1 EDA

The dataset used is the Australian Rain dataset on kaggle ([1]). It contains data for the last 10 years across different regions of Australia. There are 24 columns in the dataset, as shown by Table 1.

Location	WindDir9am(dir)	Date
MinTemp(°C)	WindDir3pm(dir)	Cloud9am(oktas)
MaxTemp(°C)	WindSpeed9am(k/h)	Cloud3pm(oktas)
Rainfall(in mm)	WindSpeed3pm(k/h)	Temp9am(°C)
Evaporation(mm)	Humidity9am(%)	Temp3pm(°C)
Sunshine(hrs)	Humidity3pm(%)	RainToday(bool)
WindGustDir(dir)	Pressure9am(hpa)	Risk$_M Mcont$)
WindGustSpeed(k/h)	Pressure3pm(hPa)	RainTomorrow(bool)

Table 1: Features of the dataset

Initially, the data is skewed, having unequal number of yes and no samples(Figure 1). Some of the fields have a large number of NaN values as well(Figure 2). All these are handled in the preprocessing stage. There are 142193 data points in total, thus making it a large dataset.

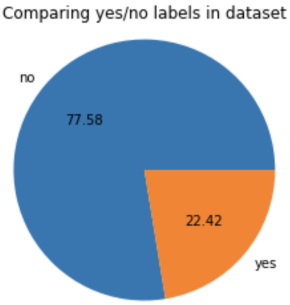

Figure 1: Skewness of the dataset

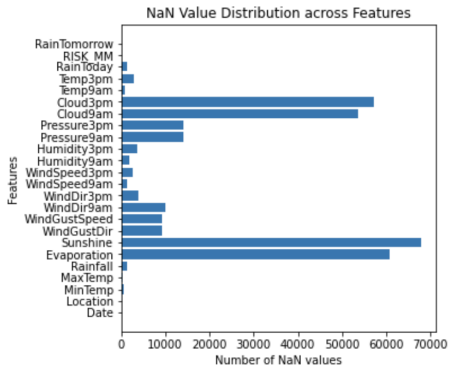

Figure 2: NAN values distribution across features

We also plot the correlation matrix of the dataset and

observe the correlations between a number of parameters as shown in Figure 3.

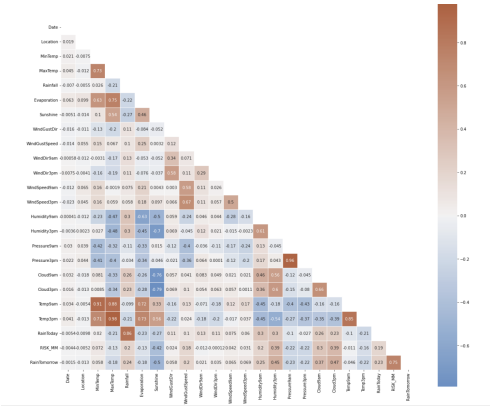

Figure 3: Correlation Matrix

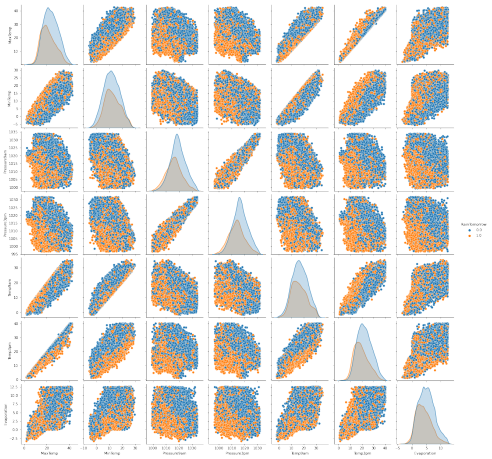

Figure 4: Pair-plot (pairwise correlation)

3.2 Dataset Preprocessing

The preprocessing pipeline shown in Figure 4 is followed:-

- Dataset is oversampled/undersampled through sklearn library.
- Categorical values are converted to continuous ones using LabelEncoder.
- NaN values are replaced by mode.
- Values outside IQR are removed as outliers.
- Data points are scaled for optimal training.

Figure 5: Preprocessing Pipeline

4 Methodology

All the ML models follow the standard pipeline:-

- Data preprocessing (as shown in Section 3.2).
- Select K=10 best features using ANOVA-F criterion.
- Split the dataset into train and test sets.
- Initialize the model.
- Run grid search with cross validation to find best possible hyperparameters.
- Fit the model using best hyperparameters found.
- Generate classification report.

5 Results and Analysis

5.1 Oversampling

Saga solver with L1 loss is used for logistic regression due to the larger dataset size and better convergence. Hinge loss (max(0,1-t.y)) is used for SGDClassifier to penalize misclassified points more. Decision Trees and Random Forests work well with 8 as the max depth to ensure low variance and higher bias than prior linear models. Boosting algorithm XGBoost performs the best so far due to boosting characteristics.

Model	Accuracy	F1 Score	ROC AUC
Logistic Regression	0.78	0.77	0.86
SGDClassifier	0.78	0.77	0.86
Gaussian Naive Bayes	0.75	0.74	0.82
Decision Trees	0.81	0.81	0.89
Random Forest	0.82	0.81	0.90
XGBoost	0.89	0.89	0.96

Table 2: Oversampled Results on Training Set

Model	Accuracy	F1 Score	ROC AUC
Logistic Regression	0.78	0.77	0.86
SGDClassifier	0.79	0.78	0.86
Gaussian Naive Bayes	0.75	0.75	0.82
Decision Trees	0.80	0.80	0.88
Random Forest	0.81	0.80	0.89
XGBoost	0.86	0.81	0.93

Table 3: Oversampled Results on Testing Set

Model	Hyperparameters
Logistic Regression	(C=0.1, solver="saga", penalty="l1")
SGDClassifier	(alpha=0.001, loss="hinge", warm_start=True)
Gaussian Naive Bayes	-
Decision Trees	(criterion="entropy", max_depth=8)
Random Forest	(n_estimators=500, max_depth=8)
XGBoost	default parameters

Table 4: Hyperparameters on different models

Rainfall	Sunshine
Humidity9am	Humidity3pm
Pressure9am	Pressure3pm
Cloud9am	Cloud3pm
RainToday	Temp3pm

Table 5: Top 10 features (by ANOVA-F value)

Gaussian Naive Bayes shows the least score out of all. This is because of close proximity of the probabilities of yes and no labels which leads to a greater number of misclassifications for a larger number of data points. Logistic regression outperforms Gaussian Naive Bayes due to better convergence on account of L1 loss function supported by saga solver. The inverse regularization constant is 0.1 which reduces variance and prevents gradient overshooting, thus avoiding overfitting.

SGD classifier produces similar results as Logistic regression, however, it uses hinge loss which is better than L1 loss function because it penalizes misclassifications more, leading to optimal training (similar to L2 loss). We also use warm start as true to ensure that for each iteration, weights from previous iterations are also considered.

Decision Trees outperform all other models mentioned above due to their tendency to overfit. If we let the model completely overfit the dataset (training score=1.0), then testing score is also very high. However, such a thing was not used and it was made sure that the train and test scores are similar to ensure low variance. For this, the tree depth was constrained to 8.

Random Forests outperform decision trees because it is an ensemble model which averages out the error of multiple decision trees classifiers and also leverages the concept of bootstrapping. In both decision trees and random forests, “entropy” is chosen to split features rather than “gini” index because “entropy” has a maximum value of 1 whereas ”gini” index has a maximum value of 0.5, therefore, ”entropy” can differentiate between features to a greater extent, although it is a bit computationally expensive.

The final classifier is XGBoost(which is currently being fine-tuned). This outperforms all prior models because it is an ensemble model using boosting techniques. This puts more weight on misclassified instances and the ones which are classified correctly will have their weights decreased for the next iteration. The dataset has no noise and outliers, making boosting an ideal choice.

5.2 Undersampling

Model	Accuracy	F1 Score	ROC AUC
Logistic Regression	0.78	0.76	0.85
SGDClassifier	0.78	0.76	0.85
Gaussian Naive Bayes	0.75	0.73	0.81
Decision Trees	0.82	0.80	0.89
Random Forest	0.83	0.81	0.90
XGBoost	0.88	0.93	0.93
AdaBoost	0.79	0.72	0.77
SVM	0.90	0.89	0.88
KNN	-	-	-

Table 6: Undersampled Results on Training Set

lbfgs solver with L2 loss is used for logistic regression due to smaller dataset size and better convergence. Log loss is used for SGDClassifier to penalize misclassified points further.

Decision Trees and Random Forests work well with 8 as the max depth to ensure lower variance in comparison to prior linear models.
Here, the criteria that has been used for feature selection is the gini criteria.

Model	Accuracy	F1 Score	ROC AUC
Logistic Regression	0.77	0.75	0.85
SGDClassifier	0.77	0.75	0.85
Gaussian Naive Bayes	0.74	0.73	0.81
Decision Trees	0.79	0.77	0.85
Random Forest	0.80	0.79	0.88
XGBoost	0.86	0.92	0.87
AdaBoost	0.78	0.70	0.76
SVM	0.86	0.91	0.86
KNN	0.80	0.74	0.79

Table 7: Undersampled Results on Testing Set

Boosting algorithm XGBoost performs the best so far due to boosting characteristics. SVM makes correct predictions for all training points unlike the testing points. This is happening because of overfitting of the model.

For Neural Networks, it is observed that adam is the better optimizer compared to sgd due to exponentially weighted averages. Increase in the depth of the network results in better metrics, however, the training time also grows, therefore, this is a tradeoff which needs to be considered.

Model	Hyperparameters
Logistic Regression	(C=1)
SGDClassifier	(alpha=0.001, loss="hinge", warm_start=True)
Gaussian Naive Bayes	-
Decision Trees	(criterion="gini", max_depth=8)
Random Forest	(n_estimators=500, max_depth=8)
XGBoost	(default parameters)
AdaBoost	($n_estimators = 500$)
SVM	(kernel = "linear", gamma ="auto", C=2)
KNN	(leaf_size=20, n_neighbors=10, p=1, weights='distance')

Table 8: Hyperparameters on different models

Model	Accuracy	F1 Score	ROC AUC
Neural Network 1	0.84	0.83	0.93
Neural Network 2	0.84	0.83	0.92
Neural Network 3	0.82	0.80	0.90
Neural Network 4	0.82	0.81	0.90
Neural Network 5	0.87	0.86	0.94
Neural Network 6	0.81	0.79	0.89

Table 9: Undersampled Results on Training Set

Model	Accuracy	F1 Score	ROC AUC
Neural Network 1	0.84	0.83	0.92
Neural Network 2	0.84	0.83	0.92
Neural Network 3	0.83	0.81	0.90
Neural Network 4	0.83	0.82	0.90
Neural Network 5	0.84	0.83	0.92
Neural Network 6	0.82	0.81	0.89

Table 10: Undersampled Results on Testing Set

Model	Hyperparameters
Neural Network 1	(solver="adam",activation="relu") [32,16,8]
Neural Network 2	(solver="adam",activation="tanh") [32,16,8]
Neural Network 3	(solver="sgd",activation="relu") [32,16,8]
Neural Network 4	(solver="sgd",activation="tanh") [32,16,8]
Neural Network 5	(solver="adam",activation="tanh") [64,32,16,8]
Neural Network 6	(solver="adam",activation="tanh") [4,2]

Table 11: Hyperparameters on different models

6 Conclusion

Based on the data analysis and application of various models, we come to the conclusion that ensemble Machine Learning models such as boosting perform well on the given dataset.

This is because it works on improving the weak components of the ensemble model to improve the bias and variance. Neural Networks slightly outperform ML models but they still remain comparable.

Out of all the given features, factors such as pressure, humidity, cloud cover, sunshine, temperature and rainfall today play the most important role in determining if it will rain tomorrow or not. The undersampling and/or over-

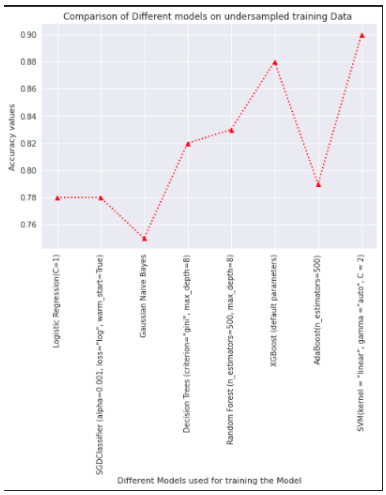

Figure 6: Comparison of Different models on undersampled training data

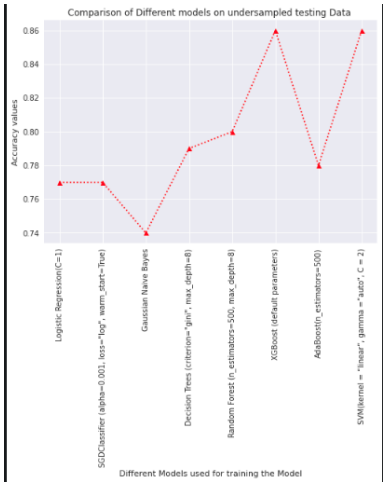

Figure 7: Comparison of Different models on undersampled testing data

sampling datasets show similar results. For further work, some of the correlated features can be dropped further and the accuracy can be compared. Also, a ratio can be devised using these parameters which can be checked for a strong correlation with rainfall for the next day. We can also extend this model to a multiclass classification with levels indicating the severity of rainfall.

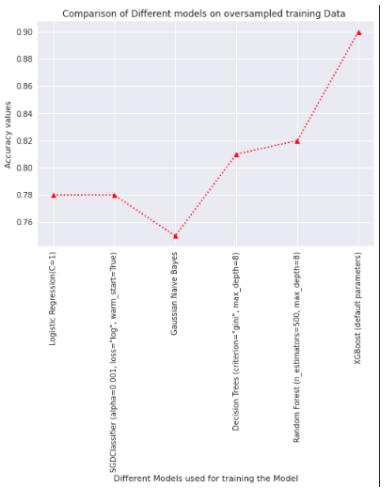

Figure 8: Comparison of Different models on oversampled training data

Member contribution:-

Aniket Verma - GNB, RF, preprocessing, ML pipeline, analysis, motivation, SVM, XGBoost.
Hardik Garg - DT, XGBoost, dataset, visualization, preprocessing, analysis, conclusion, report, Neural Network.
Shabeg Singh Gill - Logistic Regression, GNB, report, preprocessing, analysis, visualisation, results, SVM, Adaboost.
Rahul Sethi -SGD,DT,preprocessing,EDA,report,ML Pipeline,conclusion, SVM, KNN, Adaboost.

References

[1] https://www.kaggle.com/jsphyg/weather-dataset-rattle-package/version/2

[2] https://www.ijrte.org/wp-content/uploads/papers/v9i1/A2747059120.pdf

[3] http://ijream.org/papers/IJREAMV07I02SJ008.pdf