DENGUE PREDICTION USING MACHINE LEARNING

CSE445

Section 11

TEAM AKATSUKI

PRESENTED BY

KHALID HASAN
SYEDA NUSAIBA SHARIFEEN
RUHAMA FABIHA RAHMAN
S M RIAD
TIHUM KABIR

PRESENTED TO

MS. SARNALI BASAK
ADJUNCT FACULTY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NORTH SOUTH UNIVERSITY

ASSIGNED TASK: DATA AUGMENTATION

Dengue Case Prediction Using Machine Learning

Khalid Hasan, Syeda Nusaiba Sharifeen, Ruhama Fabiha Rahman, S M Riad, Tihum Kabir
Department of Computer Science and Engineering
North South University, Dhaka, Bangladesh
khalid.hasan1@northsouth.edu, syeda.sharifeen@northsouth.edu, ruhama.rahman@northsouth.edu,
sm.riad1@northsouth.edu, fazle.tihum@northsouth.edu

***Abstract*—This project aims to develop a machine-learning model to forecast dengue cases across various districts in Bangladesh. By utilizing historical data, environmental factors like temperature, humidity and rainfall, and demographic information, this paper predicts the number of dengue cases. The project emphasizes on data preprocessing and feature selection to ensure model accuracy, and tackles the challenges faced such as missing values or invariability in dengue patterns. The model provides early outbreak detection and risk assessments, allowing for timely public health interventions.The findings aim to support strategic planning in combating dengue outbreaks effectively.**

***Index Terms*—dengue prediction, machine learning, dataset, forecasting, public health, environment, demographic**

I. Introduction

This project focuses on predicting dengue outbreaks using machine learning. Dengue is a significant public health concern in tropical regions, including Bangladesh. Early prediction of outbreaks can help authorities take preventative measures to control the spread. In this project, we explore weather conditions and historical dengue cases to develop a model capable of forecasting future outbreaks. Additionally, the project highlights the significance of identifying seasonal trends and high-risk periods, ensuring that the predictions align with real-world patterns.

The remainder of this paper is tailored to provide a comprehensive understanding of the methods and analyses conducted in this study. It begins with an explanation of the data collection process, followed by detailed preprocessing techniques and feature selection strategies. Subsequently, the paper evaluates the performance of multiple machine learning models and discusses the challenges faced during their implementation. A focused analysis of the XGBoost model is presented, including its superior performance and interpretability using SHAP analysis. The paper concludes with insights derived from the results, highlighting their implications and potential avenues for future improvements.

II. Methodology

A. Project Workflow

This project focuses on analyzing real-world data to better understand dengue cases in Bangladesh. It began with gathering and combining monthly datasets into a comprehensive one, which required translation and extensive cleaning to remove impurities. Missing values were addressed using effective strategies, and data was split for training and testing with optimal configurations. The analysis primarily focused on a couple of months with a significant spike in cases, training models on selective data to identify patterns. Various algorithms were evaluated, and after testing different approaches, the dataset for all 12 months in the year was opted for. After using this approach, one model demonstrated superior performance in capturing trends. The findings highlight the challenges and unpredictability of the dataset. Figure 1 shows the project workflow in detail.

B. Data Collection

The dengue data was obtained directly from the Management Information System (MIS) of the Directorate General of Health Services (DGHS) of Bangladesh. We successfully acquired access to the required dataset after formally approaching the Director of MIS and securing the necessary permissions.

C. Dataset Visualization

The dataset comprises of 49 columns, The first column shows the names of districts from which the data was collected, the second column indicates the year of data collection and columns 3 to 14 (fig: 2) (fig: 3) represent temperatures, Each temperature represents average monthly temperatures for the respective months. Columns 15 to 26 (fig: 3) (fig: 4) represent represent the average rainfall for each respective month. Columns 27 to 38 (fig: 4) (fig: 5)represent average humidity for each respective month. the last 12 columns (fig: 5) represent the number of dengue cases for each month. The environmental factors used to show the number of dengue cases helps us visualize the dataset better. This is the case due to the strong impact of these environmental factors on the reproduction and habitat of the dengue-causing aedes mosquitoes.

District	Year	Temperature_1	Temperature_2	Temperature_3	Temperature_4	Temperature_5	Temperature_6	Temperature_7	Temperature_8
Bagerhat	2019	20.0	22.68	26.655	29.47	31.44	30.825	30.225	29.81
Bagerhat	2020	19.385	21.44	26.865	29.515	29.675	29.74	29.93	29.705
Bagerhat	2021	20.64	23.015	28.825	30.685	30.495	29.525	29.705	29.815
Bagerhat	2022	18.30134409	20.11919643	27.01370968	29.8	28.7141129	28.78069444	28.47473118	28.05107527
Bagerhat	2023	18.97930108	22.85089286	25.2625	28.90277778	29.26518817	29.75875	28.7358871	28.38938172
Bagerhat	2024	21.7	22.5	25.8	30.6	32.5	31.7	27.6	27.1
Bandarban	2019	21.055	23.13	26.145	28.665	30.065	29.98	28.635	29.505
Bandarban	2020	20.22	21.455	26.305	28.255	29.47	29.38	28.99	28.99
Bandarban	2021	21.39	22.795	27.825	29.44	29.81	28.615	28.785	28.71
Bandarban	2022	19.28669355	19.87276786	25.71182796	28.15416667	27.45913978	26.73847222	27.7563172	27.63844086

Fig. 2. Visualizing Dataset, columns: 1 - 10

Temperature_9	Temperature_10	Temperature_11	Temperature_12	Rainfall_1	Rainfall_2	Rainfall_3	Rainfall_4	Rainfall_5	Rainfall_6
29.36	28.305	25.235	20.53	0.0	100.0	41.0	101.0	82.0	286.0
29.955	29.225	25.385	20.83	54.0	3.0	8.0	103.0	245.0	429.0
29.31	29.13	24.97	21.345	0.0	0.0	0.0	0.0	187.0	430.0
27.63194444	26.35376344	22.97402778	20.08736559	15.1	35.4	20.0	29.3	240.7	333.9
28.10458333	26.68790323	23.48333333	19.72526882	0.3	7.0	87.4	66.1	143.3	289.2
29.0	29.9	nan	nan	0.0	0.0	0.0	0.0	0.01	0.01
29.095	28.73	26.175	21.385	0.0	36.0	15.0	102.0	215.0	267.0
29.46	29.565	25.86	21.795	52.0	5.0	0.0	147.0	340.0	350.0
29.47	29.465	26.54	22.67	0.0	0.0	0.0	32.0	99.0	375.0
26.75930556	26.80846774	24.74847222	21.32325269	14.2	34.4	15.7	38.7	259.3	331.3

Fig. 3. Visualizing Dataset, columns: 11 - 20

Rainfall_7	Rainfall_8	Rainfall_9	Rainfall_10	Rainfall_11	Rainfall_12	Humidity_1	Humidity_2	Humidity_3	Humidity_4
301.0	389.0	290.0	97.0	265.0	11.0	71.52	71.75	72.61	76.13
446.0	401.0	227.0	326.0	12.0	0.0	69.23	70.36	70.71	74.93
867.0	338.0	408.0	251.0	2.0	60.0	74.87	74.89	76.35	77.4
200.4	267.9	394.2	448.7	0.5	2.0	76.56182796	71.5	61.58736559	73.23194444
264.7	416.5	273.4	186.5	161.1	8.6	76.43010753	68.57142857	70.59274194	72.08055556
22.43	60.61	0.46	8.91	nan	nan	47.0	55.0	50.0	64.0
1137.0	355.0	271.0	170.0	133.0	5.0	78.87	77.54	74.48	77.37
433.0	213.0	289.0	324.0	34.0	0.0	77.74	78.5	74.39	76.83
478.0	568.0	231.0	146.0	0.0	73.0	68.68	69.04	61.94	73.47
213.4	231.4	320.3	170.2	1.3	2.3	73.47983871	63.23214286	60.68413978	74.78611111

Fig. 4. Visualizing Dataset, columns: 21 - 30

Humidity_5	Humidity_6	Humidity_7	Humidity_8	Humidity_9	Humidity_10	Humidity_11	Humidity_12	January	February
77.87	83.83	86.94	86.97	88.27	85.81	83.23	80.13	nan	nan
78.84	81.7	89.0	83.23	85.0	82.23	78.43	73.97	0.0	0.0
79.65	84.13	87.58	86.77	88.1	86.87	85.43	81.06	0.0	0.0
80.08467742	82.30694444	82.95564516	85.24731183	87.11805556	83.58064516	73.425	74.88037634	0.0	0.0
75.48924731	81.36527778	84.27284946	86.71370968	87.54166667	81.27956989	78.30555556	80.40725806	1.0	0.0
63.0	69.0	88.0	90.0	76.0	76.0	nan	nan	0.0	0.0
79.77	83.53	88.19	84.84	87.8	86.97	85.13	84.13	nan	nan
79.23	82.87	88.68	85.71	87.23	86.23	85.3	84.45	21.0	29.0
75.87	80.53	83.23	81.23	85.33	83.65	80.87	82.65	0.0	0.0
82.20698925	87.73333333	86.04301075	85.64516129	90.43333333	84.43010753	73.95138889	74.30107527	0.0	0.0

Fig. 5. Visualizing Dataset, columns: 31 - 40

March	April	May	June	July	August	September	October	November	December
nan	nan	nan	nan	nan	1126.0	114.0	34.0	16.0	8.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	4.0	10.0	3.0	3.0	0.0
0.0	0.0	1.0	0.0	1.0	1.0	15.0	55.0	81.0	28.0
0.0	0.0	1.0	5.0	118.0	331.0	453.0	469.0	308.0	34.0
0.0	0.0	0.0	1.0	9.0	6.0	27.0	30.0	nan	nan
nan	nan	nan	nan	nan	246.0	52.0	80.0	58.0	30.0
31.0	30.0	31.0	30.0	31.0	31.0	30.0	31.0	30.0	31.0
0.0	0.0	0.0	0.0	0.0	1.0	0.0	6.0	6.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	32.0	205.0	98.0

Fig. 6. Visualizing Dataset, columns: 41 - 50

D. Monthly Trend of Dengue Cases

Figure (fig: 7) illustrates the monthly trend of dengue cases, showing a very low number of cases during the initial months of the year, followed by a sudden spike in August. This increase in cases during July, August, and September is primarily attributed to the rainy season. Rain creates puddles of stagnant water, which serve as ideal breeding grounds for Aedes mosquitoes, the primary carriers of dengue. Additionally, warm temperatures combined with high humidity during these months accelerate mosquito breeding rates, thereby increasing their population. These conditions make July to September particularly favorable for the spread of dengue.

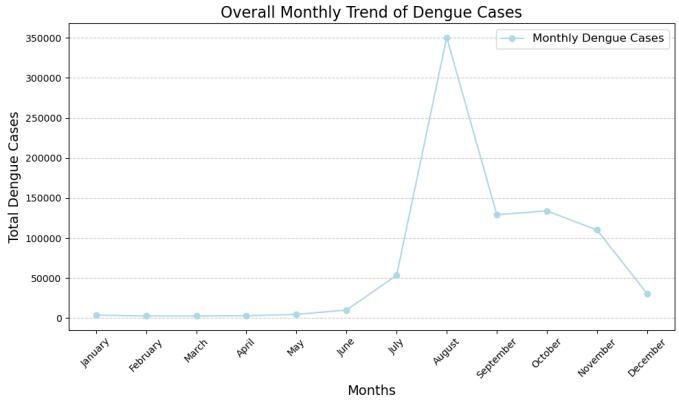


Fig. 7. Monthly Trend of Dengue Cases

E. ML Begins

As we can see from the monthly trend of Dengue cases, dengue cases spikes in the months of July to November, so at first we tried to focus in these months

F. Cross Validation

1) K-fold cross validation: We did 10-fold cross-validation, and K-Nearest Neighbors(KNN) gave the best result. (fig: 8)

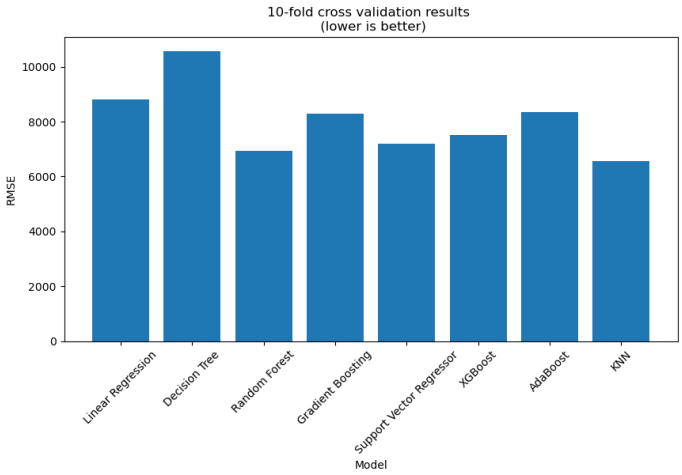


Fig. 8. Cross Validation Score

G. Finding the best Combination of Handling Missing values, Test/Train split ratio and Random State

Our models were meticulously trained using the optimal combination of strategies for handling missing values, test/train split ratios, and random states to ensure the best performance. We systematically evaluated every possible combination of these techniques and selected the configuration that yielded the best results. Based on this analysis, the models were finalized and trained accordingly.

Missing Value Handling Strategies:

- **Constant (0):** Missing values were replaced with 0 using `SimpleImputer(strategy='constant', fill_value=0)`.

- **Mean:** Missing values were replaced with the column mean using `SimpleImputer(strategy='mean')`.
- **Median:** Missing values were replaced with the column median using `SimpleImputer(strategy='median')`.
- **Mode:** Missing values were replaced with the most frequent value using `SimpleImputer(strategy='most_frequent')`.
- **Forward Fill (ffill):** Missing values were propagated forward using the last valid observation.
- **Backward Fill (bfill):** Missing values were propagated backward using the next valid observation.

Test Sizes and Random States:

- **Test Sizes:** 0.20 and 0.25.
- **Random States:** 0 and 42.

By combining these techniques and testing all possible configurations, we ensured the models were trained under optimal conditions, achieving superior results. Look at an example for K-Nearest Neighbors. (fig: 9)

```
All combinations evaluated:
      Strategy  Test_Size  Random_State           MSE        R2
0   constant_0       0.20             0  6.562599e+06  0.204210
12        mode       0.20             0  6.562599e+06  0.204210
8       median       0.20             0  6.563073e+06  0.203603
4         mean       0.20             0  6.665512e+06  0.206950
16       ffill       0.20             0  6.691363e+06  0.203929
20       bfill       0.20             0  6.722467e+06  0.200858
18       ffill       0.25             0  1.865352e+07  0.088852
10      median       0.25             0  1.878573e+07  0.077793
2   constant_0       0.25             0  1.878929e+07  0.077967
14        mode       0.25             0  1.878929e+07  0.077967
6         mean       0.25             0  1.891375e+07  0.076987
22       bfill       0.25             0  1.892880e+07  0.078054
19       ffill       0.25            42  1.086920e+08  0.093053
11      median       0.25            42  1.089190e+08  0.091962
3   constant_0       0.25            42  1.089281e+08  0.091958
15        mode       0.25            42  1.089282e+08  0.091957
23       bfill       0.25            42  1.089787e+08  0.091943
7         mean       0.25            42  1.090393e+08  0.091746
21       bfill       0.20            42  1.339520e+08  0.092464
9       median       0.20            42  1.340892e+08  0.091088
1   constant_0       0.20            42  1.341021e+08  0.091072
13        mode       0.20            42  1.341021e+08  0.091072
5         mean       0.20            42  1.352399e+08  0.083856
17       ffill       0.20            42  1.356672e+08  0.079665

Best combination:
{'Strategy': 'constant_0', 'Test_Size': 0.2, 'Random_State': 0, 'MSE': 6562598.952727271, 'R2': 0.2042102700376898 8}
```

Fig. 9. Best Strategy

H. Feature Selection

For our analysis, we selected the following features: rainfall, humidity, temperature, and dengue cases from the previous years. Environmental factors such as rainfall, humidity, and temperature were included because Aedes mosquitoes, the primary vector for dengue, thrive under specific environmental conditions. These factors directly influence mosquito breeding and survival rates, which are critical in determining the spread of dengue.

Additionally, past dengue cases were incorporated as a feature to capture the temporal and historical patterns of outbreaks. By combining environmental and historical data, the model is better equipped to identify trends and predict future dengue cases effectively.

I. Models and their Score/result

1) LinearRegression: Final model evaluation
R square value: -0.478947905338313
Mean Squared Error: 12431281.791924564
Root Mean Squared Error: 3525.8022905325483
Coefficient of Variation of RMSE: 2.4292814378736374
Theil's U: -10.312591932373346 (fig: 10)

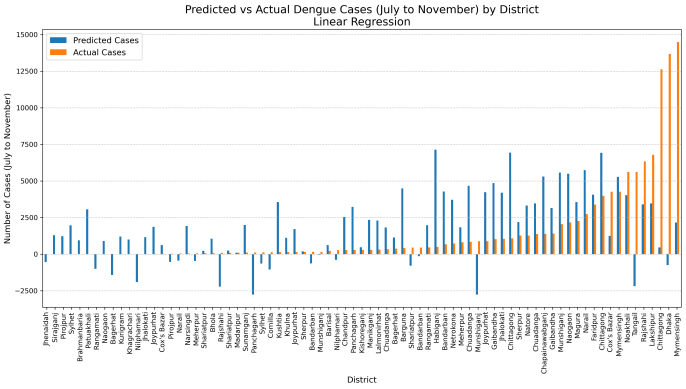


Fig. 10. Performance Evaluation of Linear Regression

2) Decision Tree Regressor: Final model evaluation
R square value: 0.04186935679619985
Mean Squared Error: 7901367.558441559
Root Mean Squared Error: 2810.93713171276
Coefficient of Variation of RMSE: 2.0078308624558905
Theil's U: -18.179250725842653 (fig: 11)

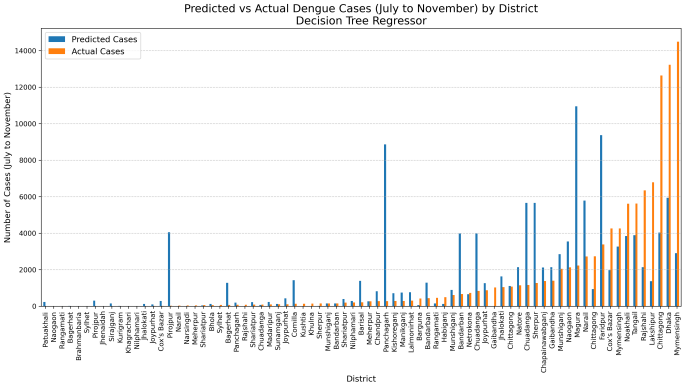


Fig. 11. Performance Evaluation of Decision Tree Regressor

3) Random Forest Regressor: Final model evaluation
R square value: 0.20870511935627134
Mean Squared Error: 6525531.505988311
Root Mean Squared Error: 2554.5119897914574
Coefficient of Variation of RMSE: 1.8246683477021328
Theil's U: -5.6149739704992365 (fig: 12)

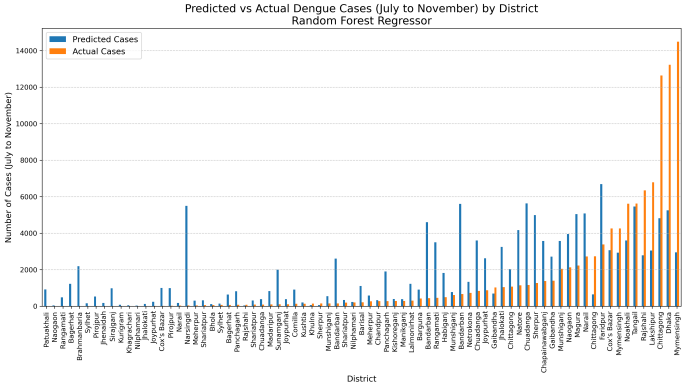


Fig. 12. Performance Evaluation of Random Forest Regressor

4) K Nearest Neighbors Regressor: Final model evaluation
R square value: 0.20421027003768988
Mean Squared Error: 6562598.952727271
Root Mean Squared Error: 2561.7570050118475
Coefficient of Variation of RMSE: 1.829843406579952
Theil's U: -11.96124535424423 (fig: 13)

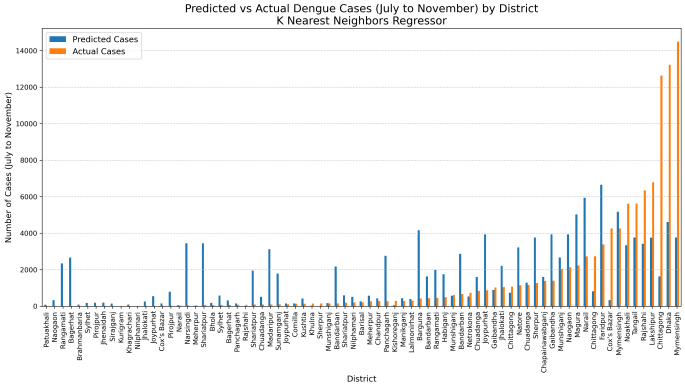


Fig. 13. Performance Evaluation of K Nearest Neighbors Regressor

5) XGBoost Regressor: Final model evaluation
R square value: 0.2193138940371907
Mean Squared Error: 6562049.237321808
Root Mean Squared Error: 2561.6497101129594
Coefficient of Variation of RMSE: 1.7649793091976973
Theil's U: -8.523863921975645 (fig: 14)

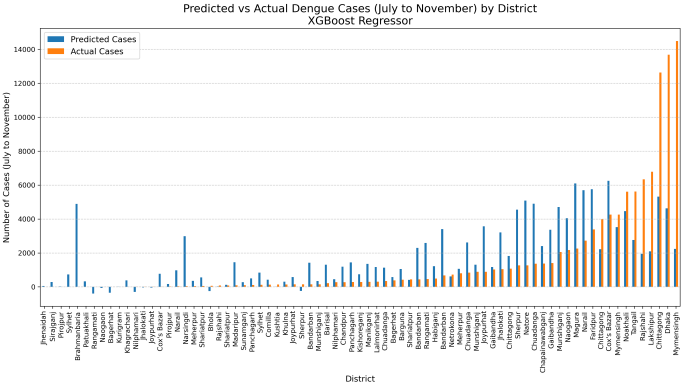


Fig. 14. Performance Evaluation of XGBoost Regressor

6) Gradient Boosting Regressor: Final model evaluation
R square value: 0.1524035461191734
Mean Squared Error: 7124463.495973003
Root Mean Squared Error: 2669.169064704033
Coefficient of Variation of RMSE: 1.8390602561134126
Theil's U: -6.500812880108979 (fig: 15)

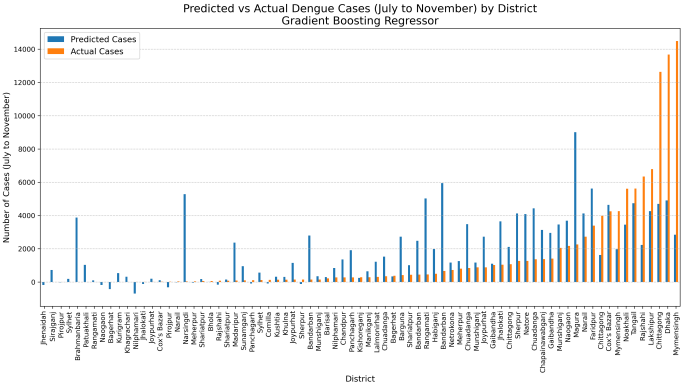


Fig. 15. Performance Evaluation of Gradient Boosting Regressor

7) AdaBoost Regressor: Final model evaluation
R square value: 0.19291319463698842
Mean Squared Error: 6651169.918828467
Root Mean Squared Error: 2578.986219200961
Coefficient of Variation of RMSE: 1.837020711179223
Theil's U: -2.515026755944233 (fig: 16)

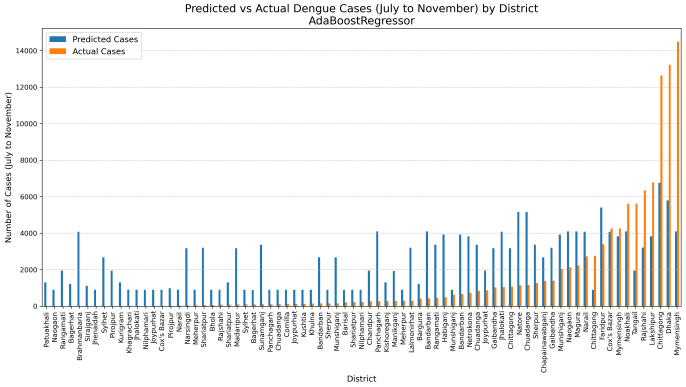


Fig. 16. Performance Evaluation of AdaBoost Regressor

J. Model Comparison

The models are evaluated using five key metrics: R-Square, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Coefficient of Variation of RMSE (CVRMSE), and Theil's U. The following charts illustrate the performance of each model on these metrics.

1) R-Square Value: The R-Square value measures the proportion of the variance in the dependent variable that is predictable from the independent variables.

Fig. 17. R-Square Value Comparison Across Models

2) Mean Squared Error (MSE): MSE measures the average squared difference between predicted and actual values. A lower MSE indicates better accuracy.

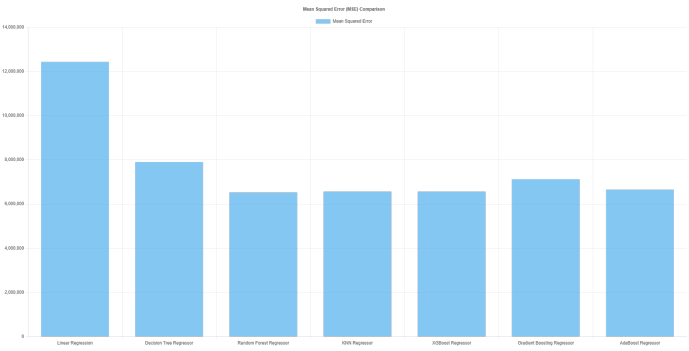


Fig. 18. Mean Squared Error (MSE) Comparison Across Models

3) Root Mean Squared Error (RMSE): RMSE represents the square root of the average squared errors. It provides a measure of the model's predictive accuracy in the same units as the dependent variable.

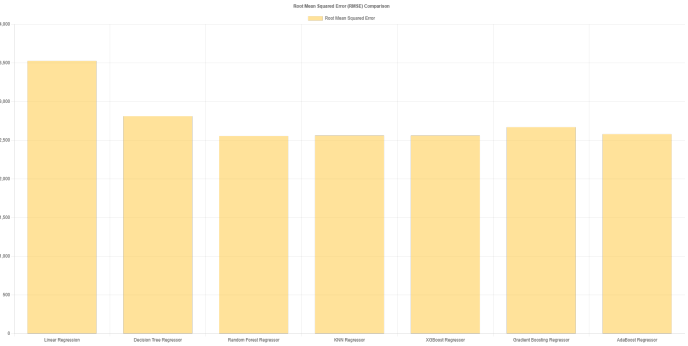


Fig. 19. Root Mean Squared Error (RMSE) Comparison Across Models

4) Coefficient of Variation of RMSE (CVRMSE): CVRMSE indicates the relative magnitude of RMSE compared to the mean of the observed data, providing a standardized measure of error.

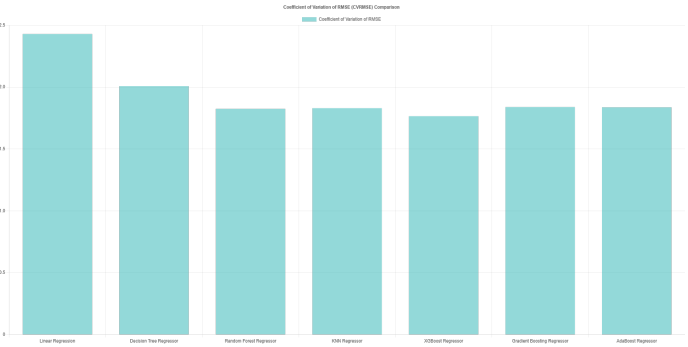


Fig. 20. Coefficient of Variation of RMSE (CVRMSE) Comparison Across Models

5) Theil's U: Theil's U measures the model's predictive accuracy compared to a naive forecast. A lower value indicates better forecasting performance.

Fig. 21. Theil's U Comparison Across Models

K. Improving Model Performance by Expanding Dataset

The initial results from the models were unsatisfactory, with performance metrics indicating that the models were far from being suitable for practical use. The primary goal was to predict dengue cases for the months of July to November, as these months typically see a spike in dengue cases. However, due to the dataset's limited size, the models could not produce accurate predictions.

To address this issue, we expanded the scope of the dataset to include all 12 months instead of focusing solely on July to November. This approach increased the amount of data available for model training, improving the models' ability to learn patterns and make predictions.

After implementing this change, we observed significant improvements in the results. Among the models tested, the **XGBoost Regressor** stood out, delivering much better performance compared to the other models. This suggests that with a larger dataset covering all months, the models, especially XGBoost, are better equipped to predict dengue cases effectively.

L. Model Selection and Focus on XGBoost

Using the expanded dataset with 12 months of data, we tested various models to improve prediction accuracy. However, except for the XGBoost model, none of the models produced significant results or were suitable for practical application. Consequently, we decided to focus on the XGBoost model, conducting various analyses and optimizations to enhance its performance and achieve the best possible results.

1) Evaluation Metrics of XGBoost Model: The XGBoost model demonstrated excellent performance based on the following evaluation metrics:

- **R-Square Value:** 0.9999994073961121
- **Mean Squared Error (MSE):** 5.086795817279584
- **Root Mean Squared Error (RMSE):** 2.2553926082346694
- **Coefficient of Variation of RMSE (CVRMSE):** 0.0014754027443913238
- **Theil's U Statistic:** 0.00034129305741046874

2) Actual vs Predicted Values: The scatter plot in Figure 22 compares the actual dengue cases with the predicted values generated by the model. The points align closely with the line of perfect prediction, indicating high accuracy in the XGBoost model's predictions.

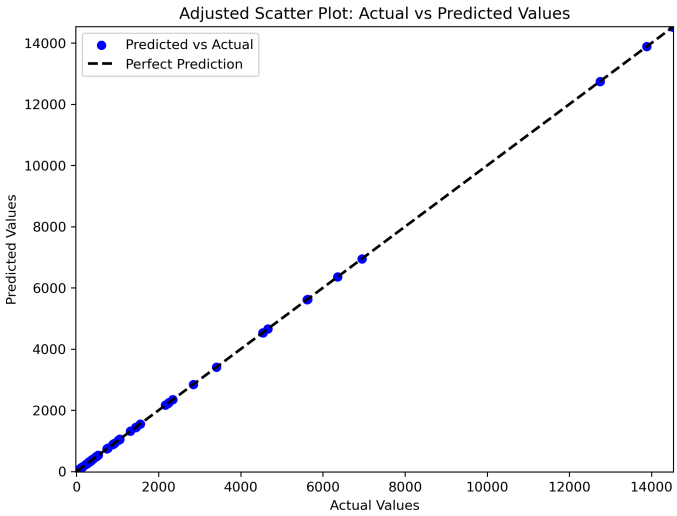


Fig. 22. Scatter Plot: Actual vs Predicted Values

3) Residuals Plot: Figure 23 shows the residuals plot for the XGBoost model. The residuals are distributed around zero, with no clear pattern, suggesting that the model effectively captures the relationship between the features and the target variable. However, some outliers are observed, which may indicate areas for further improvement.

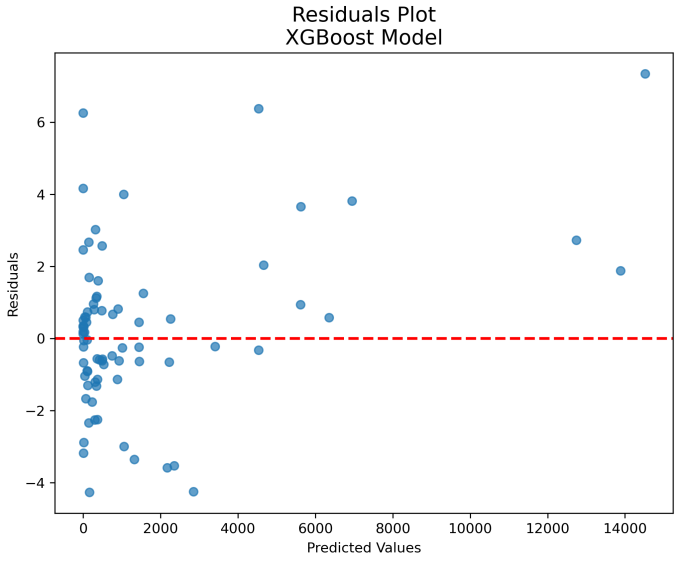


Fig. 23. Residuals Plot for XGBoost Model

4) Learning Curve: The learning curve in Figure 24 illustrates the training and validation errors as the size of the training dataset increases. The training error remains consistently low, while the validation error decreases initially and stabilizes, highlighting the model's ability to generalize well with sufficient training data.

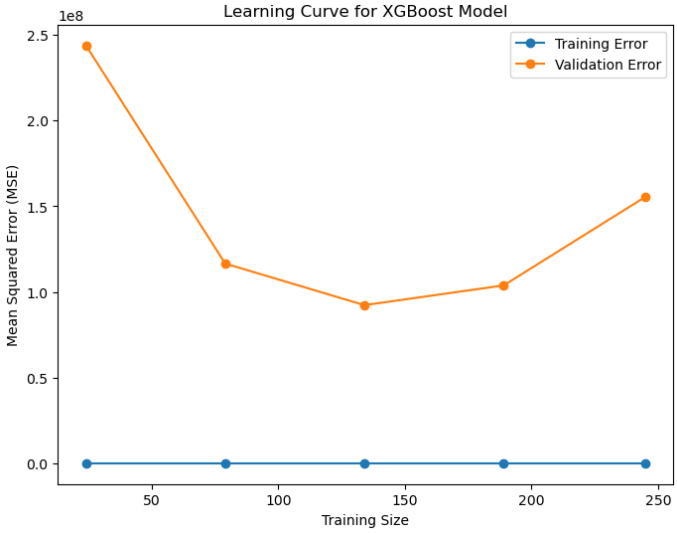


Fig. 24. Learning Curve for XGBoost Model

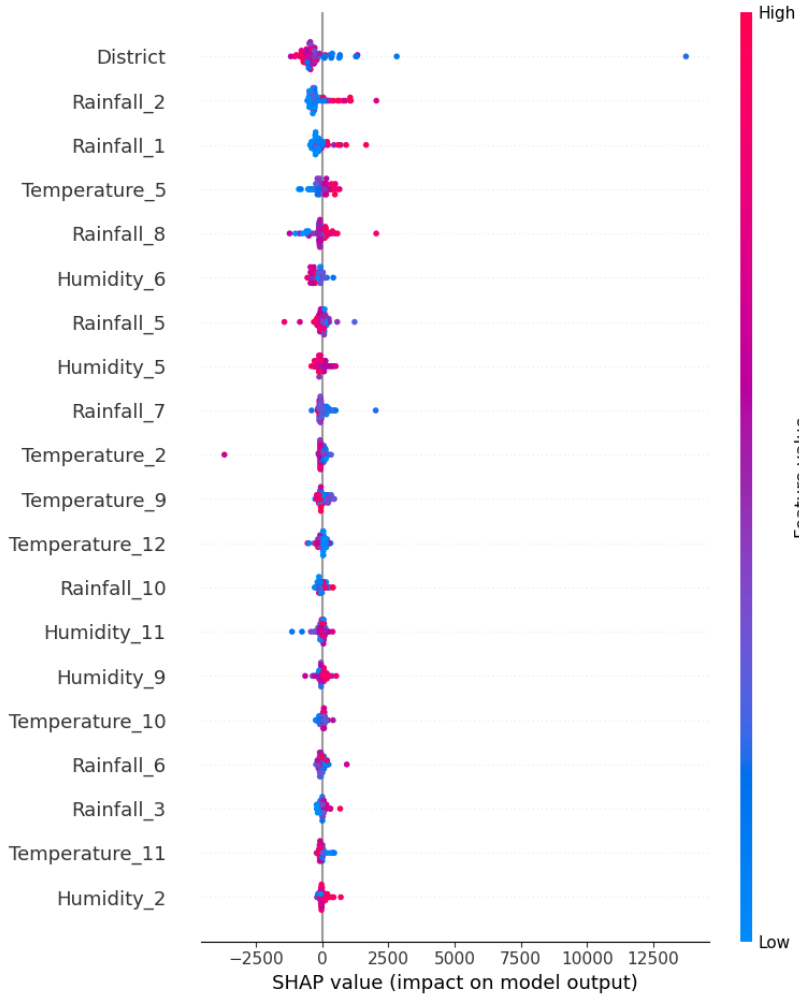


Fig. 25. SHAP Summary Plot

5) SHAP Analysis for Feature Importance: SHAP (SHapley Additive exPlanations) analysis was used to interpret the XGBoost model by evaluating feature importance and their impact on predictions. The SHAP summary plot and waterfall plot comprehensively understand how features influence the model's predictions.

a) SHAP Summary Plot: The SHAP summary plot, shown in Figure 25, ranks features based on their impact on the model's output. Each dot represents a SHAP value for a specific feature and data point. The dot's position indicates the impact's magnitude and direction, while the color represents the feature's value (high or low). Features such as `District`, `Rainfall_2`, and `Temperature_5` exhibit the highest impact on the model's predictions.

b) SHAP Waterfall Plot: The SHAP waterfall plot, presented in Figure 26, explains a single prediction by breaking down the contribution of each feature to the model's output. Negative contributions (blue) indicate features that decrease the prediction, while positive contributions (pink) indicate features that increase it. Key features such as `District`, `Temperature_5`, and `Rainfall_2` significantly influence the final prediction.

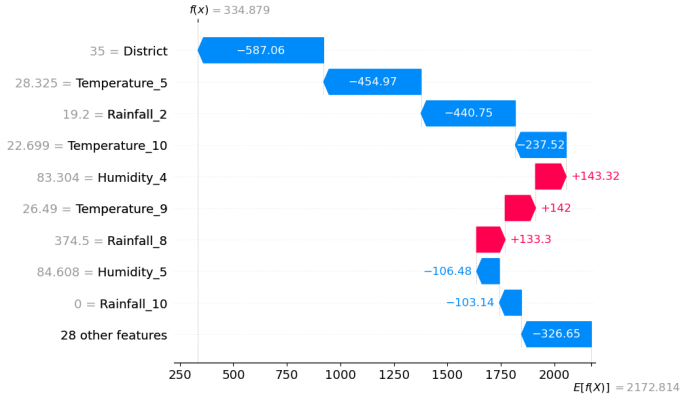


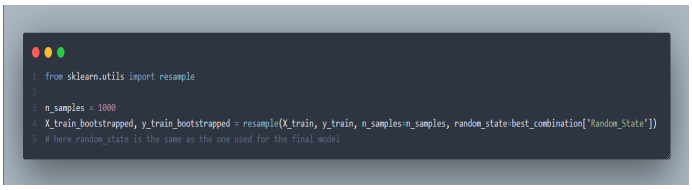
Fig. 26. SHAP Waterfall Plot

III. Updated Work

As suggested by our supervisor, we explored data augmentation techniques to address the small size of our dataset. Specifically, we looked into **SMOTE** and **Bootstrapping**. However, since our project focuses on a regression problem, SMOTE was not suitable, as it is primarily used for classification tasks.

Instead, we decided to use **Bootstrapping**, particularly the resampling technique. This method leverages the `resample` function from `sklearn.utils`, which is a utility designed for resampling arrays or sparse matrices. The `resample` function simplifies the bootstrapping process by directly taking the input data and the desired number of samples, and it also allows setting a `random_state` for reproducibility.

For our experiment, we generated **1000 bootstrapped samples** (Figure 27). Unfortunately, despite augmenting and scaling the dataset, we did not achieve the desired results. While the **K-Nearest Neighbors (KNN)** model showed a slightly improved performance, the improvement was insignificant(Table V-E). This suggests that the augmentation methods, while useful for increasing data size, did not substantially enhance model performance in our case.



```
from sklearn.utils import resample

n_samples = 1000
X_train_bootstrapped, y_train_bootstrapped = resample(X_train, y_train, n_samples=n_samples, random_state=best_combination["Random_State"])
# here random_state is the same as the one used for the final model
```

Fig. 27. Bootstrapping

IV. Conclusion

This study demonstrates the potential of machine learning models, particularly XGBoost, in predicting dengue cases using environmental, historical, and demographic data. While initial attempts with limited datasets yielded suboptimal results, expanding the dataset to cover all months significantly improved model performance. To further address the challenges posed by a small sample size, we employed **bootstrapping** as a data augmentation technique. Bootstrapping helped increase the number of samples by resampling from the original dataset, but despite these efforts, the improvements in model performance were modest. While the K-Nearest Neighbors (KNN) model showed a slight performance boost, the overall results were not as significant as hoped.

Despite the progress, challenges such as data quality and variability remain. Future work can focus on obtaining more comprehensive datasets and exploring advanced modeling techniques to further improve prediction accuracy and support public health decision-making.

The project is available on GitHub at the following link: https://github.com/khalidhasananik/CSE445-Project.git

V. Group Members

A. Khalid Hasan, Id: 2111736642, Email: khalid.hassan1@northsouth.edu

B. Syeda Nusaiba Sharifeen, Id: 2132576642, Email: syeda.sharifeen@northsouth.edu

C. Ruhama Fabiha Rahman Id: 2031430042, Email: ruhama.rahman@northsouth.edu

D. S M Riad, Id: 2112094642, Email: sm.riad1@northsouth.edu

E. Tihum Kabir, Id: 2131035642, Email: fazle.tihum@northsouth.edu

Model	Original R-Square	Original MSE	Bootstrapping R-Square	Bootstrapping MSE
Linear Regressor	-0.4789	12431281.79	-0.4890	12515499.10
AdaBoost Regressor	0.1929	6651169.92	-0.2086	9960126.97
Decision Tree Regressor	0.0419	7901367.56	-0.3367	11023210.51
K-Nearest Neighbors (KNN)	0.2042	6562598.95	0.1966	6625098.73

TABLE I
COMPARISON OF REGRESSION MODELS: R-SQUARE AND MEAN SQUARED ERROR (MSE) BEFORE AND AFTER BOOTSTRAPPING

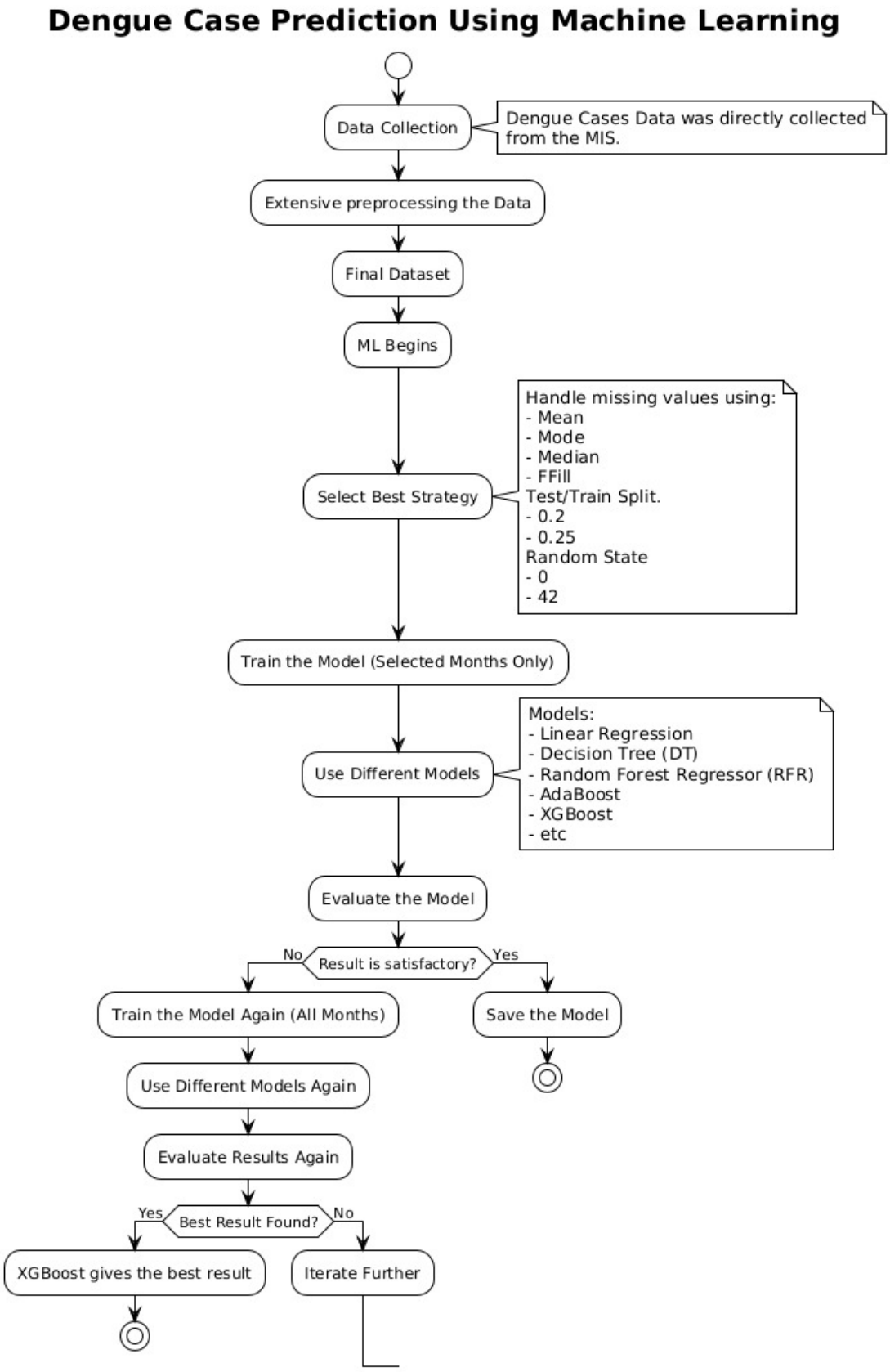


Fig. 1. Project Workflow