Predicting Hospital Stay Duration Using Synthetic Healthcare Data

1st Caleb Burton
Department of Electrical and Computer Engineering
University of North Carolina at Charlotte
Charlotte, NC, USA
cburto34@charlotte.edu

2nd William Miller
Department of Electrical and Computer Engineering
University of North Carolina at Charlotte
Charlotte, NC, USA
wmille40@charlotte.edu

3rd Robert Thomas
Department of Electrical and Computer Engineering
University of North Carolina at Charlotte
Charlotte, NC, USA
rthoma97@charlotte.edu

GitHub Repository: https://github.com/wmiller0906/Predicting-Hospital-Stay-Duration-Using-Healthcare-Data.git

CONTENTS

***Abstract*—This project aims to build a machine learning model using linear regression to predict the length of a patient's hospital stay based on demographic and medical information. By analyzing patient records, admission details, and hospital resource usage, we seek to improve hospital efficiency by forecasting bed availability and potential capacity constraints. This prediction model will allow hospitals to better allocate resources, reduce congestion, and improve patient care planning.**

***Index Terms*—machine learning, hospital stay prediction, linear regression, healthcare analytics, feature selection**

I. INTRODUCTION

When new patients are admitted into a busy hospital, administrators have to predict how long they are likely to stay so the hospital can keep rooms available for patients with scheduled surgeries and other procedures. When a patient occupies a room longer than anticipated and the room is booked for another patient, the hospital administrators must then make a difficult decision to move one of the patients. With no space left, the hospital often groups patients in one room. The whole process can cause stress on troubled patients who are already experiencing illness.

Take an example of an elderly patient being admitted with pneumonia. The hospital predicts a recovery within 10 days and anticipates the patient's dismissal at that time. The room is also booked for another patient who will be recovering from surgery in two weeks. However, the patient with pneumonia experiences worse symptoms, and their stay is extended. Now the hospital must either move one of the two patients to a different room. If the hospital had more accurately predicted the stay of the first patient, they could have planned accordingly and put a different patient in this room.

Many factors can be used to predict a patient's length of stay (LoS). More factors than a hospital attendant can take into consideration. Machine learning can be used to take all factors into account and make the best possible decision for allocating hospital beds and other resources.

II. Related Work

This study from Toho University Ohashi Medical Center [1] introduces a deep learning model to predict hospitalization costs and stay durations for heart failure patients using electronic health record data. It outperforms traditional regression models by handling complex variable interactions, improving accuracy. Despite promising results, limitations include missing data biases and the need for broader validation. A flowchart of their model demonstrates their use of convolution and normalization in their linear regression process in figure 1.

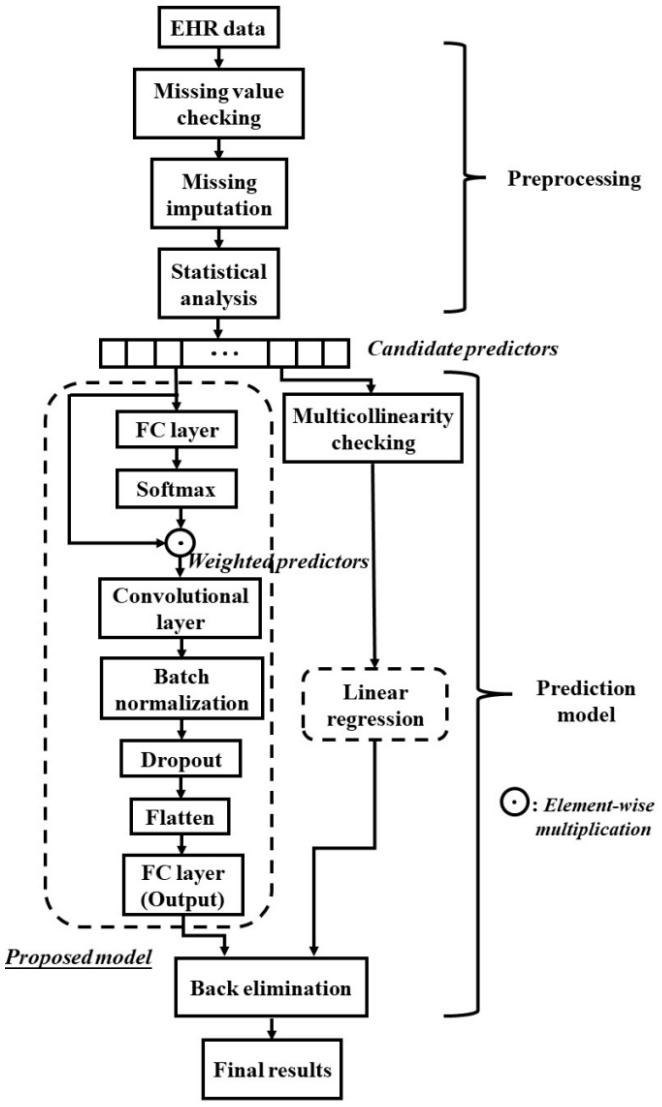

Fig. 1. Prediction Model — Toho University

This document from the American Health Care Association [2] outlines the methodology for calculating the LoS) in long-term care (LTC) facilities using MDS 3.0 data. It defines LoS as the number of days between admission and discharge, with adjustments for interruptions in service. The report provides metrics such as median LoS and the percentage of stays under 7, 14, 20, and 45 days. It explains how expected and risk-adjusted LoS are calculated using logistic regression and national benchmarks. Additionally, it details the clinical characteristics used in risk adjustment to provide a more accurate measure of resource utilization. The findings help interpret LoS in the context of other quality indicators like rehospitalization rates and functional improvement.

This paper from PLOS Digital Health [3] provides a systematic review of hospital LoS prediction methods, highlighting the need for a unified framework to improve generalizability across different hospital environments. It examines various approaches, including statistical models, machine learning techniques, and operational research-based methods, discussing their strengths and limitations. The review identifies challenges such as data preprocessing inconsistencies and model tuning issues that restrict applicability beyond individual hospitals. The authors propose a standardized framework to enable direct comparisons between LoS prediction models and suggest further research into fuzzy systems and model interpretability to enhance predictive accuracy. They include a diagram showing the taxonomy of LoS prediction models in figure 2.

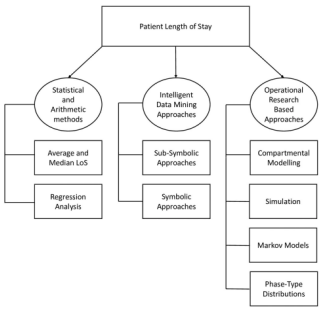

Fig. 2. Prediction Model — PLOS Digital Health

III. Methodology

A. Dataset

1) Data Features: The dataset used for this project was provided by Microsoft, hosted on their GitHub, as a collection of synthetic hospital data designed to demonstrate how LoS predictions can be made using machine learning for applications in healthcare analytics and planning. It includes 100,000 observations across 27 features. The features present include continuous values, such as vital signs and laboratory measurements, and categorical features, such as the presence or absence of a specific clinical diagnosis. Of these 27 features, 21 were used to develop the model. Continuous, categorical, and unused features are listed in Table I, Table II, and Table III, respectively. While admission date and discharge date were not directly used in training the model, they were used in data preprocessing to develop a LoS column as the target variable.

Prior to model development, features were visualized to examine any potential trends that may arise during training. These visualizations were generated as plots and are shown in Figure 3 and Figure 4.

TABLE I
CONTINUOUS FEATURES

Readmission Count (In last 30 days)
Hematocrit (g/dL)
Neutrophil Presence (cells/μL)
Sodium Level (mmol/L)
Blood Glucose Levels (mg/dL)
Blood Urea Nitrogen (mg/dL)
Creatinine (mg/dL)
BMI (kh/m2)
Pulse (bpm)
Respiration Rate (breaths/min)

TABLE II
CATEGORICAL FEATURES

End Stage Renal Failure (Receiving dialysis)
Asthma
Iron Deficiency
Pneumonia
Substance Use Disorder
Major Psychological Disorder
Depression
Other Psychological Disorders
Fibrosis
Malnutrition
Blood Disorder

TABLE III
UNUSED FEATURES

Electronic ID
Gender
Secondary Diagnosis
Facility ID
Admission/Discharge Date

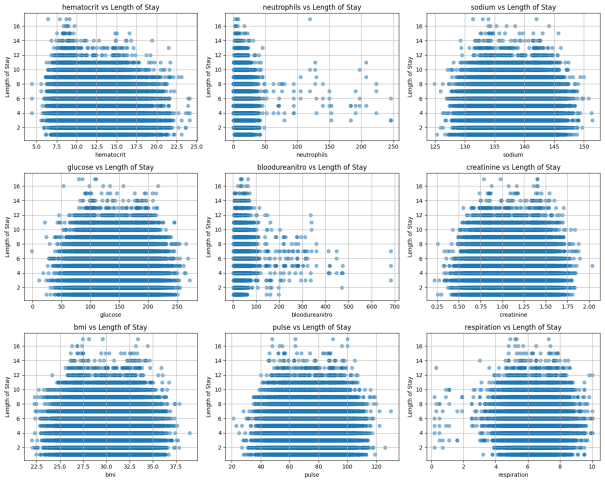

Fig. 3. Visualization of Continuous Features

2) Partitioning and Preprocessing: Dealing with an outside dataset required "cleaning" where date formats and data types were not efficiently listed for python integration. The given dates in the dataset listed each day as 'mm/dd/yyyy'. The program took in these values and converted them into a 'yyyy-mm-dd' format to standardize them before converting them into a numerical value. This was a vital step due to the length of stay being a created column; calculated by finding the difference between the check-in and check-out dates. The dates were observed with an additional day on leap years to ensure accurate day count.

The input variable of "rcount", which represents the number of this patient's readmissions preceding this check-in, needs to be listed as an integer for accurate categorization. The dataset originally included "5+" as a signifier of this patient having over 4 readmissions. This character is read as a string by python. The program converted any instance of "5+" into "5", allowing for any number to be read as an integer. The cleaned dataset was split into 80% for training and 20% for a validation set to form and test the training model.

Continuous variables such as blood pressure and lab results were normalized and standardized to zero mean and unit variance so that no single scale dominated model fitting.

B. Model and Training Approach

1) Single-Variable Regression: The most fundamental application of a linear regression model is training and testing each individual feature in isolation, using one predictor at a time, using equation 1 below.

Where x denotes the single standardized feature and θ_1 is its weight. By examining each feature in isolation, a clear measure of how much a given feature shifts the expected LoS when applied to the multivariate model. Although these models do not account for interactions between variables, they can quickly rank which factors may have the most effect on the predicted LoS by observing how well the regression model matches the actual results. An example of this single-variable model versus the data trend can be found in Figure 6.

2) Multivariate Regression: Bringing together all input features, including vital signs, lab values, diagnosis flags, and prior admissions found in equation 2 below.

Each coefficient, θ, is attached to its respective variable x, appropriately weighing each variable to minimize the prediction error of the regression line. Using the "sklearn" library in python, normalized and standardized fit models were trained and tested to predict the next patient length of stay given its features. The plots of the standardized and normalized multivariate regressions can be seen in Figure 8 and Figure 9.

3) Selective-Variable Regression: While pursuing the best regression fit for data, some variables are not beneficial towards the prediction. To find and eliminate any features that produce more noise than progress, p-values were assigned to each feature using the previously designed regression model. The p-values were designed to be higher when the more that a given feature is considered non-beneficial. A threshold of '0.05' for each p-value was set. The program identified any features that are above this threshold to exclude from the new selective regression model. The program identified the following features as unbeneficial to use for the new model: sodium, glucose, creatinine, and BMI. All other features are included as inputs for the selective-variable regression model.

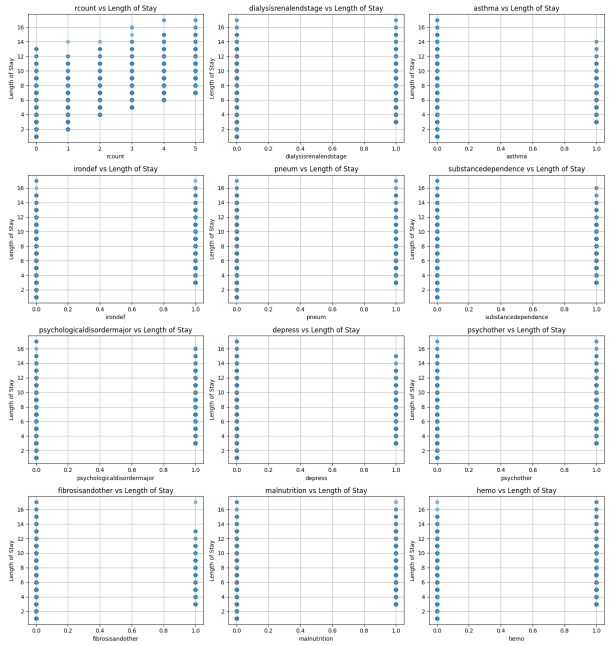

Fig. 4. Visualization of Categorical Features

$$\hat{y} = \theta_0 + \theta_1 x \tag{1}$$

Fig. 5. Single-Variable Linear Regression Equation

The final plot for the selective variable regression can be seen in Figure 10.

4) Feature Interaction Regression: Another extension to the regression model that was tested for overall improvement is pairing features together and using their products as inputs for a new model. The program was written to test every combination of feature products. The R^2 and MAE metrics were compared for each combination. The following 5 feature products were used as features in in the new regression model: $(rcount \times sodium)$ $(rcount \times bmi)$, $(rcount \times pulse)$, $(rcount \times respiration)$, and $(rcount \times creatinine)$. Arrays were made to make each of these products act as feature columns. The new inputs were included in addition to the existing features. Standardized and normalized training models were produced, tested, and plotted. The resultant plot of the standardized feature interaction linear regression model is shown in Figure 11.

IV. Results and Analysis

To verify the performance of a model, MAE and R^2 metrics were collected and a scatter plot was generated to visualize how the predictions on training and test data compared to the true values for the target variable.

1) Univariate Models: Of the univariate models, the best performance came from the model training on readmission count (rcount). The metrics of each of these models is collected in Table IV and Table V in descending order, with the best performance starting the list.

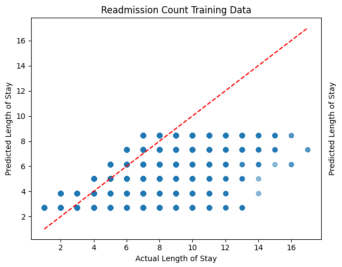

Fig. 6. Example of univariate regression

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_p x_p \tag{2}$$

Fig. 7. Multivariate Linear Regression Equation

TABLE IV
MAE of each Univariate Model

Model Feature	**MAE**
Readmission Count	1.2296
Blood Urea Nitrogen	1.9048
Creatinine	1.9148
Glucose	1.9149
BMI	1.9142
Sodium	1.9151
Neutrophils	1.9152
Pulse	1.9152
Respiration	1.9173
Hematocrit	1.9202

TABLE V
R^2 of each Univariate Model

Model Feature	**R^2**
Readmission Count	0.5622
Blood Urea Nitrogen	0.0219
Respiration	0.0007
Hematocrit	0.0055
Pulse	<0.0001
Neutrophils	<0.0001
Creatinine	<0.0001
Glucose	<0.0001
BMI	<0.0001
Sodium	<0.0001

The univariate model trained using the feature readmission count had the best performance with a R^2 of 0.5622 and MAE of 1.2296. No other univariate model had performance comparable, with the next best R^2 being 96.1% lower than the readmission count model's R^2. This significance over other features shows that readmission count is the most predictive feature for determining LoS. Figure 12 is the performance plot of the readmission count model's predictions vs actual LoS.

2) Standard Multivariate Model: The multivariate regression model was trained and tested using the normalized and

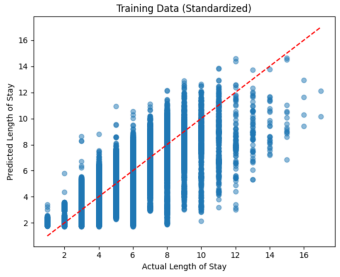

Fig. 8. Standardized Multivariate model plot

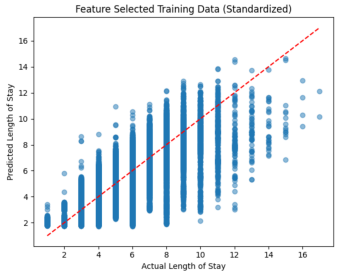

Fig. 9. Normalized Multivariate model plot

standardized data. Performance for both methods of data preprocessing was identical in regards to MAE, R^2, and the prediction performance scatter plot. Table VI collects the metrics of the multivariate model and Figure 13 is the scatter plot of the standardized model's performance.

TABLE VI
PERFORMANCE METRICS OF MULTIVARIATE MODEL

Metric	Standardized	Normalized
MAE	0.8999	0.8999
R^2	0.7498	0.7498

With a R^2 score of 0.75, this model is able to provide some accurate predictions of LoS based on the provided features and displays a substantial improvement over the best univariate model which had a R^2 score of 0.56. This improvement over the univariate models shows a degree of interaction between features which will be explored further in part D. In order to improve the R^2 score further, feature selection and feature interaction terms will be incorporated into a multivariate model and analyzed.

3) Multivariate Model with Feature Selection: After gathering the p-values, features with a p-value greater than 0.05 were excluded. A multivariate model was then trained with the remaining features using standardized and normalized data. The results for both processed datasets were identical. The metrics from this multivariate model are found in Table VII and the scatter plot of the standardized model is found in Figure 13.

The performance of this multivariate model using feature selection was identical to the multivariate model without feature selection, both achieving an R^2 score of 0.75. Due

Fig. 10. Selective Feature Regression Plot

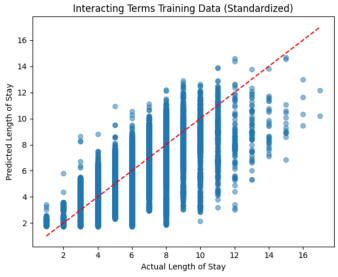

Fig. 11. Standardized Feature-Interaction model plot

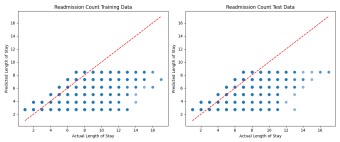

Fig. 12. Model Performance of Readmission Count on Training and Test Data

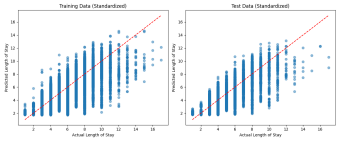

Fig. 13. Scatter Plot of Standardized Multivariate Model's Prediction vs Actual LoS

TABLE VII
PERFORMANCE METRICS OF MULTIVARIATE MODEL (WITH FEATURE SELECTION)

Metric	Standardized	Normalized
MAE	0.8999	0.8999
R^2	0.7498	0.7498

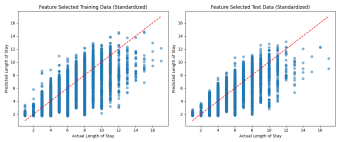

Fig. 14. Scatter Plot of Standardized Multivariate Model's Prediction vs Actual LoS (with Feature Selection)

to feature selection's lack of improvements to the model's R^2 score, it will not be incorporated into the multivariate model.

4) Multivariate Model with Feature Interaction Terms: As seen from the improvement of the multivariate model over the best performing univariate model, a degree of interaction exists between the features in predicting LoS. In order to take advantage of this trait, feature interaction terms were created and then used to develop univariate models. The interaction terms from the top five univariate models' R^2 scores were appended to the multivariate model's features. Table VIII lists the top five interaction terms and their respective R^2 score.

TABLE VIII
R^2 SCORES OF INTERACTION TERM MODELS

Interaction Term	R^2 Score
rcount * sodium	0.5616
rcount * BMI	0.5583
rcount * respiration	0.5537
rcount * pulse	0.5441
rcount * creatinine	0.5350

The results for both processed datasets were identical. The metrics of this model approach are found in Table IX and the scatter plot of the standardized model is in Figure 15.

TABLE IX
PERFORMANCE METRICS OF MULTIVARIATE MODEL (WITH INTERACTION TERMS

Metric	Standardized	Normalized
MAE	0.9001	0.9001
R^2	0.7497	0.7497

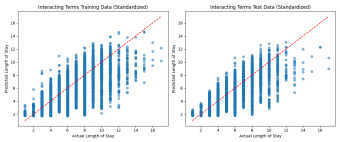

Fig. 15. Scatter Plot of Standardized Multivariate Model's Prediction vs Actual LoS (with Interaction Terms

The performance of this model was slightly worse than the multivariate model without feature interaction terms. Even though the interaction terms had comparable R^2 scores to the best performing individual feature, their addition marginally hindered performance. This may be due to additional noise present in the model from adding these interaction terms. For future work, it would be beneficial to analyze p-values in addition to R^2 scores to determine which interaction terms to add. Due to interaction terms failing to improve the multivariate model's performance, they were not added.

5) Random Forest Regression Model: The final action to check for improvements to a model predicting LoS was to look outside of linear regression algorithms. For proof of concept of better models, Random Forest Regression was used. The metrics of this trial are recorded in Table X and the scatter plot of the model trained on standardized data is Figure 16.

Using random forest regression, there was a substantial improvement in performance. R^2 score had a 23.7% improvement from 0.75 to 0.93. Random forest regression is an

TABLE X
PERFORMANCE METRICS OF MULTIVATE MODEL (RANDOM FOREST REGRESSION

Metric	Standardized	Normalized
MAE	0.3832	0.3816
R^2	0.9269	0.9276

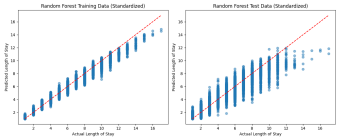

Fig. 16. Scatter Plot of Standardized Multivariate Model's Prediction vs Actual LoS (Random Forest Regression

algorithm that has several advantages over linear regression. One such advantage is that linear regression assumes a linear relationship between the features and target variable, but this may not be the case. Random forest regression handles non-linear relationships more efficiently and can make predictions without this handicap. Another advantage is that random forest regression automatically handles feature interactions, looking for relationships between features and applying them instead of relying on the developer to select interaction terms. A final advantage of random forest regression is its ability to resist the presence of noise in training the model. Due to relying on an average of the predictions along multiple trees, it can make more accurate predictions with less of an impact from outliers and unhelpful data. Random forest regression cannot be added to our model because it is no longer considered linear regression, but it does show that for future work, there exists algorithms besides linear regression that can predict LoS with greater accuracy.

V. CONCLUSIONS

Developing this model started by analyzing univariate linear regressions, which revealed readmission count as the most predictive single feature with a R^2 score of 0.5622, roughly 2467% greater than the next highest R^2 score for univariate regressions. This significant performance over other features suggests that past hospitalizations are a strong predictor of future length of stay (LoS).

Building upon this, the standard multivariate linear regression model achieved an R^2 score of 0.7498, showing that in union multiple features provide a much more accurate prediction of LoS. While feature selection and interaction terms were explored to look for improvements in the multivariate model, neither managed to provide a greater R^2 score. However, feature selection did reveal that not all variables had a statistically significant linear relationship with LoS and their removal did not lead to a decrease in performance.

Finally, to explore further areas of improvement in developing a LoS prediction model, random forest regression was employed. It achieved a significantly higher R^2 of 0.9276, an improvement of 23.7% over the best model produced in this project. Due to random forest regression's ability to capture

non-linear relationships, this reveals linear regression may be limited in modeling LoS using real-world healthcare data.

While the multivariate linear regression model's performance was acceptable for this experiment, the results suggest that other more advanced algorithms can be used to produce a model capable of substantial performance gains, suitable for deployment into healthcare analytics and planning.

AI Assistance Disclosure

Portions of this report, including conceptual clarification, methodology refinement, and editorial improvements, were developed with assistance from ChatGPT, an AI language model by OpenAI. All final decisions, analyses, and interpretations were made by the authors.

References

[1] X. Zhou, X. Zhu, and K. Nakamura, "Prediction of hospitalization cost and length of stay for patients with heart failure using deep learning," *2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech)*, pp. 158–161, 03 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9754924

[2] "Length of stay calculation." [Online]. Available: https://www.ahcancal.org/Data-and-Research/LTC-Trend-Tracker/Documents/Length%20of%20Stay%20Calculation.pdf

[3] K. Stone, R. Zwiggelaar, P. Jones, and N. Mac Parthaláin, "A systematic review of the prediction of hospital length of stay: Towards a unified framework," *PLOS Digital Health*, vol. 1, p. e0000017, 04 2022. [Online]. Available: https://journals.plos.org/digitalhealth/article?id=10.1371/journal.pdig.0000017

[4] A. Choudhury, "Hospital length of stay dataset microsoft," Kaggle.com, 2022. [Online]. Available: https://www.kaggle.com/datasets/aayushchou/hospital-length-of-stay-dataset-microsoft

[5] Microsoft, "Home," Github.io, 2025. [Online]. Available: https://microsoft.github.io/r-server-hospital-length-of-stay/index.html

[6] A. S. Chadha, "What do normalization and standardization mean? when to normalize data and when to standardize data?" Medium, 07 2021. [Online]. Available: https://shorturl.at/5RFKt

[7] GeeksforGeeks, "Random forest regression in python," GeeksforGeeks, 06 2019. [Online]. Available: https://www.geeksforgeeks.org/random-forest-regression-in-python/#