



Optimization of healthcare process management using machine learning

Andreas Avgoustis

Committee of Three

Themis Exarchos **Associate Professor**
Aristidis G. Vrahatis **Assistant Professor**
Panagiotis Vlamos **Professor**

December 15, 2023

Abstract

Healthcare management plays a crucial role in ensuring the efficient delivery of healthcare services. This responsibility encompasses strategic planning, resource management, and regulatory compliance to enhance patient care outcomes. In this paper, we delve into the multifaceted nature of healthcare management, highlighting the expertise required to optimize processes within dynamic healthcare environments. Furthermore, we explore the potential of machine learning in addressing operational challenges within healthcare. By examining various machine learning algorithms, we identify their advantages and limitations, proposing a structured method of application. Through this analysis, we aim to illuminate how machine learning can minimize patient waiting times and optimize overall healthcare operations. By integrating advanced technologies and strategic approaches, healthcare organizations can harness the transformative power of machine learning to improve efficiency and patient care delivery.

Acknowledgments

I would like to extend my deepest gratitude to my professor, Themis Exarchos, whose mentorship, expertise, and unwavering support have been invaluable throughout this journey. Their guidance, encouragement, and constructive feedback have not only shaped my understanding of the subject matter but have also inspired me to strive for excellence in all aspects of my academic pursuits.

I am also indebted to Aristidis G. Vrahatis and Panagiotis Vlamos, whose insights, wisdom, and encouragement have enriched my learning experience and propelled me forward in my academic and professional endeavors.

Furthermore, I am grateful to my family and friends for their unwavering support, understanding, and encouragement during challenging times. Their love, encouragement, and belief in me have been a constant source of strength and motivation.

Contents

1	Introduction	8
2	Background and Related Work	10
3	Proposed Methodology	14
3.1	Dataset	15
3.2	Processing Data	20
3.3	Model and Algorithm Structure	22
4	Experimental Evaluation	27
4.1	Libraries and Frameworks	27
4.2	Experimental Setup	28
4.3	Experimental Results	29
5	Proposed Application	34
6	Conclusion and Future Work	37
A	Appendix : Coding Details of Methodology	44

List of Figures

1	Artificial Intelligence	8
2	Machine Learning Usage	10
3	Basic architecture of convolutional neural networks	11
4	Internet of Things proposed system architecture	13
5	Scheme of the real-time forecasting system	14
6	Architectural diagram of proposed methodology	15
7	Patient Waiting for appointment examination process	16
8	Linear Regression Usage	23
9	Decision Tree Usage	24
10	XGBoost Usage	25
11	Architecture of Recurrent Neural Network	27
12	Mean Absolute Error Results (a) Facility 1 (F1). (b) Facility 2 (F2). (c) Facility 3 (F3). (d) Facility 4 (F4).	31
13	R-squared Results (a) Facility 1 (F1). (b) Facility 2 (F2). (c) Facility 3 (F3). (d) Facility 4 (F4).	33
14	Proposed Application Use Case	35
15	Architecture of Proposed Application	35
16	Deployment Roadmap	36

List of Tables

1	Dataset Features	17
2	Mean Absolute Error Performance	30
3	R-Squared Performance	33

List of Equations

1	Linear Regression	23
2	XGBoost Regression	25
3	Reccurent Neural Network	26
4	Mean Absolute Error	29
5	R-Squared	32

1 Introduction

Healthcare management, a critical facet in the realm of healthcare, assumes the responsibility of orchestrating the effective delivery of healthcare services. This multifaceted role encompasses strategic planning, resource management, and the seamless coordination of various components within a healthcare organization, all aimed at ensuring the delivery of high-quality patient care. At the forefront of healthcare management are dedicated professionals who wield their expertise to optimize processes, allocate resources judiciously, and navigate the intricate landscape of regulatory compliance. Their unwavering focus is on enhancing the overall effectiveness and efficiency of healthcare delivery, addressing challenges head-on, and striving to improve patient outcomes within the dynamic and complex healthcare environment [1].

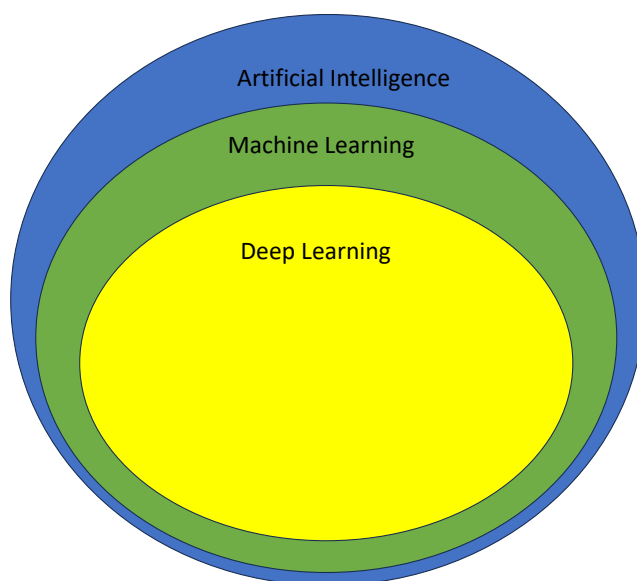


Figure 1: Artificial Intelligence

In the ever-evolving panorama of healthcare management, the advent of Artificial Intelligence (AI) heralds unprecedented opportunities. These cutting-edge technologies offer innovative solutions to streamline operations, optimize resource utilization, and elevate the overall quality of patient care. AI applications, including predictive analytics, have the potential to revolutionize healthcare management by forecasting patient admission rates and wait times. This foresight enables proactive resource allocation and the optimization of staffing levels, contributing significantly to oper-

ational efficiency [31].

Machine learning algorithms, a subset of AI, play a pivotal role in enhancing clinical decision support systems and operational tools. These algorithms empower healthcare professionals with valuable insights for personalized treatment plans, ultimately improving clinical outcomes. The integration of process automation through AI-driven technologies alleviates administrative burdens, allowing healthcare staff to redirect their focus towards the core of their profession—patient care.

For practical utility, the best clinical operations tools should operate in real-time, require minimal initial assumptions, automatically identify key features responsible for failures and successes, and naturally adapt to frequent changes in the environment. This necessity for all-encompassing analysis and adaptivity naturally leads to the exploration of machine learning techniques.

For these advancements to be practically utilitarian, the best clinical operations tools should operate in real-time, demanding minimal initial assumptions. They should seamlessly identify key features responsible for both failures and successes, adapting naturally to the frequent changes in the healthcare environment. This quest for comprehensive analysis and adaptability naturally directs attention toward the exploration and implementation of various machine learning techniques.

As the healthcare industry journeys further into the era of AI, it is imperative to not only recognize its potential but also to foster an environment that encourages continual exploration and refinement of these technologies. The symbiosis between healthcare and AI holds the promise of redefining standards, enhancing patient outcomes, and ultimately contributing to the evolution of healthcare management on a global scale.

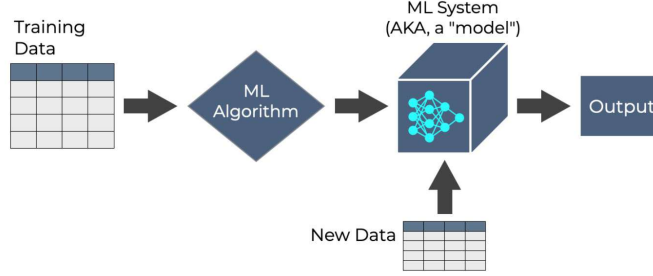


Figure 2: Machine Learning Usage

However, the practical advantages of machine learning extend beyond diagnostics and treatment planning. Waiting times for patients[1], a critical aspect of the healthcare experience, can be significantly addressed through the application of machine learning in healthcare operations management. This article aims to delve into the specific suitability of machine learning in solving operational problems, exploring the advantages and limitations of various machine learning algorithms. Additionally, we present a structured method of application and discuss the limitations of usage, shedding light on how machine learning can be a transformative force in minimizing patient waiting times and optimizing overall healthcare operations.

The subsequent sections of this article are organized as follows: Section 2 delves into key recent studies about healthcare operations, highlighting machine learning advantages and deep learning studies in healthcare management. Section 3 elucidates the proposed methodology and regression analysis for patient wait time. Section 4 details the pre-processing techniques applied to datasets and the experiments conducted, presenting and discussing the respective results. Section 5 outlines the proposed application structure and the technologies that will be employed. Finally, the article concludes in Section 6.

2 Background and Related Work

In the field of healthcare operations management, a notable study [13] has focused on harnessing the potential of machine learning technologies to address specific medical

challenges. The study presented an innovative artificial intelligence solution utilizing Convolutional Neural Network (CNN) algorithm for malaria diagnosis. Leveraging a substantial dataset comprising 24,958 malaria microscopy images sourced from the NIH National Library of Medicine[17], the machine learning model underwent deep learning training. Subsequently, 2,600 images were selected for the final testing phase of the proposed diagnostic architecture. Empirical results from the study showcased the effectiveness of the CNN diagnostic model. Precision metrics of 0.97 for uninfected cells and 0.99 for parasite cells, coupled with recall rates of 0.99 for uninfected cells and 0.97 for parasite cells, underscored the robustness of the model. The F1-scores further emphasized the diagnostic performance, registering at 0.98 for both uninfected and parasite cells. These findings [13] collectively suggest the significant advantages of machine learning-based diagnostic methods over traditional manual approaches. The study highlighted improvements in healthcare operational capabilities, encompassing diagnostic quality, reduced processing costs, shorter lead times, and increased productivity.

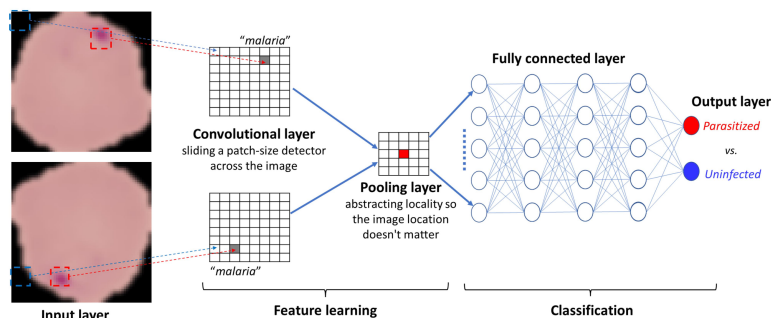


Figure 3: Basic architecture of convolutional neural networks

The application of machine learning techniques in the context of hospital operations remains an area of limited exploration. In this study[4], we introduce a prediction pipeline designed for the dynamic environment of a UK teaching hospital's emergency department (ED), utilizing live electronic health records for patients.

The primary objective is to generate short-term, probabilistic forecasts of emergency admissions, addressing a critical aspect of hospital management. The prediction pipeline employs a series of XGBoost classifiers applied to a dataset comprising 109,465 ED visits. The resulting AUROC scores ranged from 0.82 to 0.90, contingent upon the elapsed visit time at the point of prediction. Patient-level probabilities of admission are aggregated to forecast both the number of admissions among current ED patients and the total emergency admissions within specified time-windows, incorporating patients yet to arrive. The efficacy of the pipeline is evidenced by a MAE of 4.0 admissions, surpassing a benchmark metric with an MAE of 6.5. Furthermore, models developed during the challenging operational conditions of the Covid-19 pandemic, based on 104,504 later visits, demonstrated AUROCs of 0.68–0.90 and an MAE of 4.2 (30%) compared to a benchmark’s 4.9 (33%). A significant aspect of this work[4] involves addressing the challenges inherent in designing and implementing models for real-time use. This encompasses considerations such as temporal framing, data preparation, and the dynamic nature of operational conditions. The study not only contributes valuable insights into the potential of machine learning in hospital operations but also sheds light on practical strategies for overcoming the intricacies associated with real-time model deployment.

The advent and maturation of the Internet of Things (IoT) have significantly transformed healthcare practices, marking extraordinary data processing capabilities and massive data storage facilitated by the integration of the Industrial Internet of Things (IIoT)[17]. However, the transition to IoT-enabled healthcare brings forth security challenges, particularly concerning the storage of sensitive medical data in cloud servers. In response to these challenges, this work proposes an IoT-based deep learning-based[7] privacy preservation and data analytics system. The architecture involves a wearable IoT module collecting user data, which is then securely processed in the cloud server. A Convolutional Neural Network (CNN) is employed to analyze health-related information while ensuring the privacy of users by segregating and separating sensitive data. The proposed system introduces a secure access control module based on user attributes, contributing to enhanced security in the IoT-Healthcare system. The CNN classifier achieves commendable performance metrics with precision, recall, and F1 score reaching 95%. Performance improvements are observed with an increase in the size of the training set, and the introduction of data augmentation enhances system performance. Notably, an accuracy rate of around 98% is achieved with an increased user count. In conclusion, the proposed IoT-based deep learning system[7] exhibits strong potential in addressing security concerns, ensuring privacy, and maintaining high data integrity in the context of healthcare data analytics. The experimental results underscore the robustness of the

system, positioning it as a promising solution for secure and effective healthcare data management.

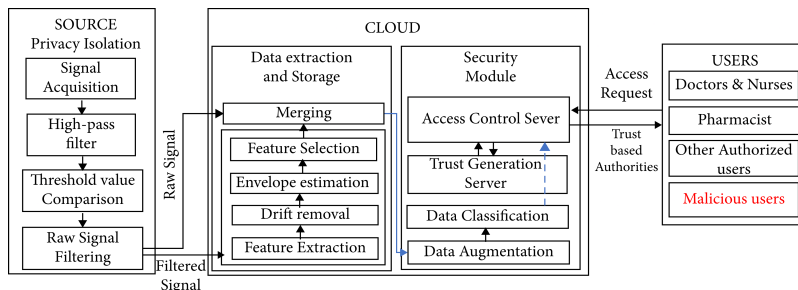


Figure 4: Internet of Things proposed system architecture

In the evolving landscape of healthcare, the delivery of high-quality and cost-effective care to patients necessitates the integration of modern and powerful technologies within healthcare institutions. Despite significant strides in the computerization and digitization of medical practices, the development of efficient and robust management tools has faced challenges. This is particularly attributed to the extreme complexity and variability inherent in healthcare operations [2], surpassing the capabilities of conventional management approaches. One crucial avenue for addressing these challenges lies in the application of machine learning algorithms. Known for their scalability and adaptability to intricate patterns, machine learning algorithms offer a promising solution. Two notable advantages, the ability to construct robust models from numerous weakly predictive features and the capacity to discern crucial factors within complex feature sets, make machine learning particularly well-suited to tackle the intricate nature of healthcare operations. The primary focus of this work[1] is to investigate this symbiotic relationship between machine learning and healthcare operations, specifically concentrating on two major operational challenges: predicting operational events and identifying key workflow drivers. Through practical examples, it aims to illustrate how machine learning methodologies can enhance human comprehension and management of healthcare operations, ultimately leading to more efficient healthcare delivery. In essence, this work[1] endeavors to bridge the existing gap in healthcare management tools by harnessing the potential of machine learning. By doing so, it aspires to contribute to the creation of a more streamlined and adaptive healthcare operational framework, better equipped to meet the dynamic demands of the healthcare ecosystem.

The efficient management of Emergency Departments (EDs) hinges on accurate esti-

mations of waiting times, a crucial aspect in resource allocation and the anticipation of potential overcrowding. However, the intricate and dynamic nature of ED operations poses a significant challenge to waiting time prediction[1]. In this study[18], employing predictive analytics on two extensive datasets derived from real-world EDs. The machine learning models evaluated include Lasso, Random Forest, Support Vector Regression, Artificial Neural Network, and the Ensemble Method. Notably, to enhance prediction accuracy, novel queue-based variables are introduced[18] as additional predictors, capturing the current state of the ED. These findings reveal that the Ensemble Method emerges as the most effective in predicting waiting times. Balancing accuracy and computational efficiency, Random Forest proves to be a reasonable trade-off. These results carry significant practical implications for EDs and hospitals, suggesting the feasibility of implementing a real-time performance monitoring system. Such a system, empowered by machine learning techniques, holds the potential to support operational decision-making within EDs, optimizing resource utilization and enhancing overall efficiency.

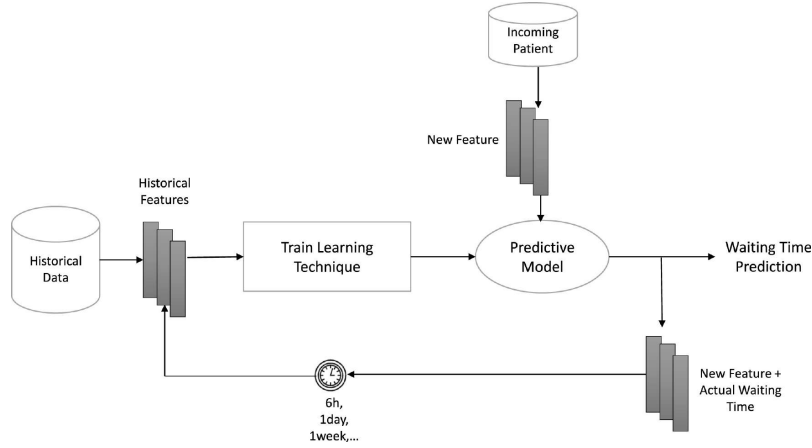


Figure 5: Scheme of the real-time forecasting system

3 Proposed Methodology

Our work proposes the use of various machine learning and deep learning regression algorithms trained on schedule time series data from various hospital facilities[1].

Each hospital facility contains data from arrival and schedule time of patients that came. The key purpose is to convey the machine learning and deep learning algorithms with the capability to effectively discern schedule arrival and actual arrival of doctor appointment from hospitals and the possibility of creating application to help patients for these appointments and doctors organizing them. A bird's-eye view of the key pillars of this work is presented in Figure 6

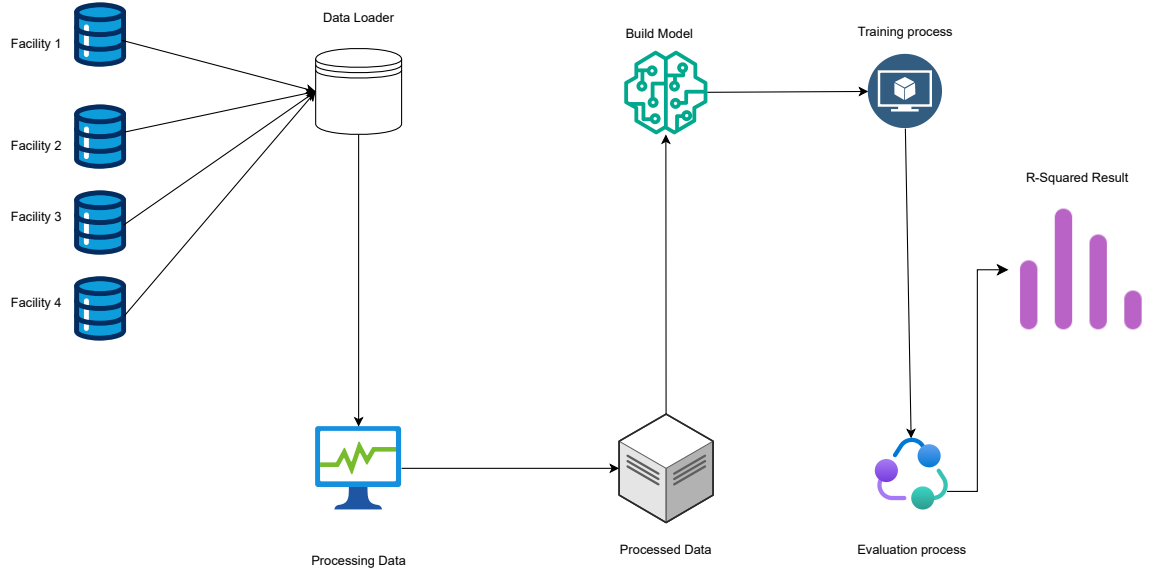


Figure 6: Architectural diagram of proposed methodology

3.1 Dataset

Hospital Facilities are containing data[2] for various purposes. Each facility has its own methods of scheduling patients. Predicting patient wait times is a major problem and cause of patient dissatisfaction and missed care opportunities as it being

shown in Figure 7. The ability to accurately predict current wait times serves two important goals: if displayed to the patients, it permits them to plan their time; and when displayed to the managers, it allows them to intervene if necessary and possible.

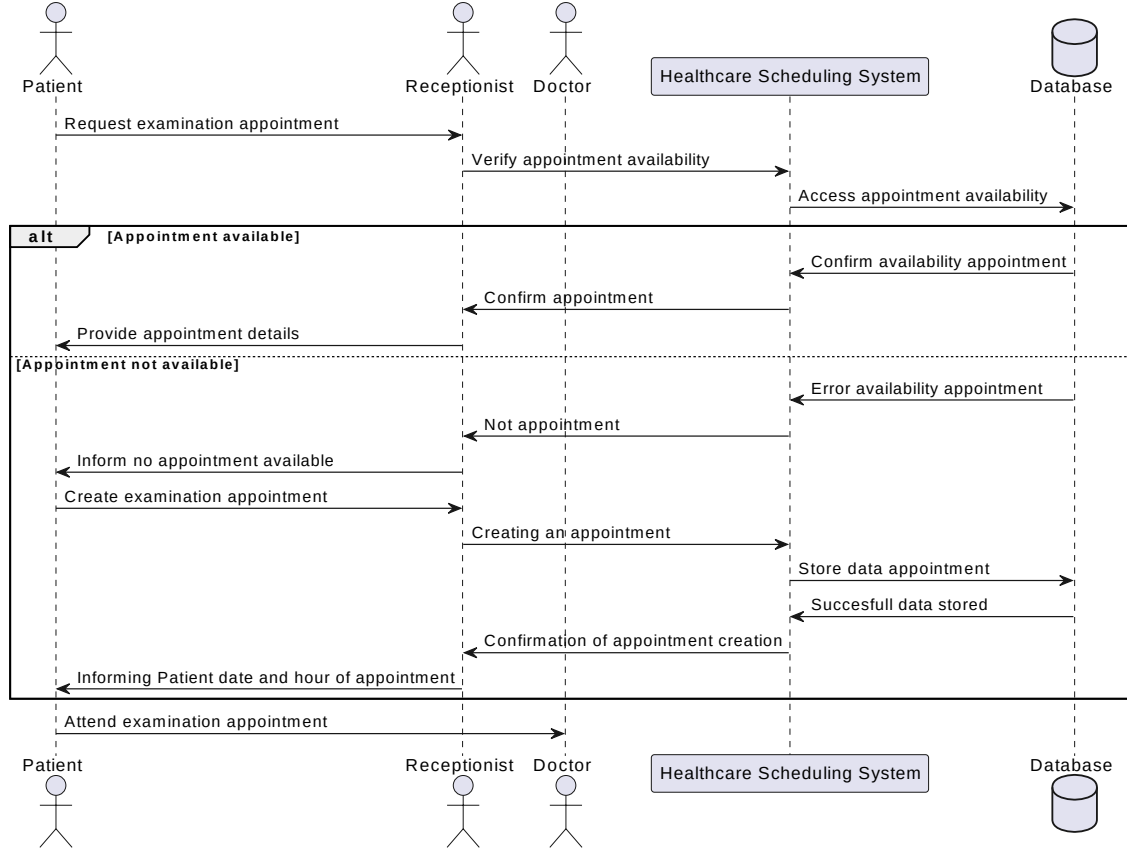


Figure 7: Patient Waiting for appointment examination process

Each facility contains data from patient arrival time, scheduled exam begin time, and actual begin time. There are some differences in our case of facilities. We have data collected from three facilities[1] that have scheduled appointments and from one walk-in facility. Each dataset contains the waiting time of patients appointment and 66 more features as you see in Table 1. In the next section, we will discuss the processing of data before using it in our models.

Table 1: Dataset Features

Feature	Description
x_ArrivalDTTM	Arrival time of patient.
x_ScheduledDTTM	Scheduled time of patient's exam.
x_BeginDTTM	Begin time of patient's exam.
SumHowEarlyWaiting	Sum of how early the patients in line are for their appointment.
AvgHowEarlyWaiting	Average of how early the patients in line are for their appointment.
LineCount0Strict	Number of patients in line with scheduled times after the current time.
SumWaits	Sum of the wait times for patients in line.
LineCount0/1/2/3/4	Number of patients in line measured when a patient arrives, 15, 30, 45 & 60 minutes before.
FlowCount2/4	Number of patients starting exams in the 30- and 60-minute window before the patient arrived.
SchFlowCount2/4	Number of patients scheduled in the 30- and 60-minute window before the patient arrived.
FutFlowCount2/4	Number of patients scheduled in the 30- and 60-minute window after the patient arrived.
DelayCount	Number of delayed exams up to the current time of day.
DelayCountLastHour	Number of delayed exams in the last hour.
mintime	Minimum wait time for the day.
maxtime	Maximum wait time for the day.
AheadCount	Number of patients scheduled before the current patient for the day.
ThoracicCount	Number of patients waiting for a thoracic exam.

Feature	Description
PediatricCount	Number of patients waiting for a pediatric exam.
NeuroCount	Number of patients waiting for a neuro exam.
AbdominalCount	Number of patients waiting for an abdominal exam.
VascularCount	Number of patients waiting for a vascular exam.
CardiacCount	Number of patients waiting for a cardiac exam.
MSKCount	Number of patients waiting for a musculoskeletal exam.
NumScannersUsedToday	Number of scanners in the facility that have been used on that day.
SumInProgress	Sum of the length of time exams have been in progress.
BeforeSlot	Time since the last appointment slot.
AfterSlot	Time until the next appointment slot.
Median5	Median delay/wait time for the 5 most recent customers.
MostRecent1/2/3/4/5	Delay/wait time for the most recent patient, 2nd, 3rd, 4th & 5th most recent patients.
StartTime	Hour of arrival, and 2nd, 3rd & 4th powers of the hour of arrival to account for nonlinear trends.
IsLast	Last scheduled patient for the day.
IsFirst	First scheduled patient for the day.
NoneInProgress	No exams in progress.
NoneCompleted	No exams completed that day.
NoneInLine	No patients in line.
SumWaitByTaskTypeLine	Sum of waits of patients in line by exam type.

Feature	Description
AvgWaitByTaskTypeLine	Average waits of patients in line by exam type.
SumTimeToCompleteInProgress	The sum of the expected times to complete of the exams in progress
DelayedInLine	The number of patients in line who are delayed.
SumDelayWaitingByExamCode	Sum of delays of patients in line by exam type.
SumDelayWaitingInLine	Sum of delays/waits of patients in line.
SumDelayInProgress	Sum of delays/waits of exams in progress.
ExpectedDelayNextExam	Expected delay of the next scheduled exam.
AvgAgePeopleWaiting	Average age of the patients in line.
DayOfWeek	The day of the week the exam is scheduled.
Month	The month of the year the exam is scheduled.
DayOfYear	The day of the year the exam is scheduled.
InProgressSize	Number of exams in progress for the facility.
AvgWaitLastK1Customers	Average wait for the last 2, 4, and 8 customers.
NumCompletedToday	Number of exams completed up to the current day.
NumCompletedInLastW1/W2/W3	Number of exams completed in the last 30, 60, and 120 minutes.
NumCustomersInLastW1/W2/W3	Number of customers who have arrived in the last 30, 60 & 120 minutes.
AvgWaitLastW1/W2/W3	Average delay/wait time last 30, 60, and 120 minutes.

Feature	Description
AvgDelayForDay	Average delay/wait for patients for that day.
OutpatientWaitingCount	Number of outpatients waiting in line.
MalesWaitingCount	Number of male patients waiting in line.
NumAddOnsToday	Number of people who have been added to the schedule for today.
NumAddOnsLastW2	Number of people who have been added to the schedule in the last 60 minutes.
NumScheduledNextSlot	Number of patients scheduled in the next slot.
NumScheduledNextW2	Number of people scheduled in the next 60 minutes.
SumTimeToCompleteNextSlot	Expected time to completion of exams in the next slot.
SumTimeToCompleteNextW2	Expected time to completion of exams scheduled in the next hour.
WithContrastCountWaiting	Number of patients waiting for an exam with contrast.
WithAndWithoutContrastCountWaiting	Number of patients waiting for an exam with and without contrast.
WithContrastCountInProgress	Number of exams in progress with contrast.
WithAndWithoutContrastCountInProgress	Number of exams in progress with and without contrast.

3.2 Processing Data

After acquiring raw data from four distinct facilities, a careful examination revealed variations among the features of each facility. The significance of each feature emerged as crucial in the development of machine learning models tailored for regression analysis. The data processing phase became imperative, involving the transformation of regular data into a numeric format. Noteworthy steps included the removal of superfluous features or rows that could adversely impact regression

outcomes. Additionally, innovative features were engineered based on the existing dataset. This meticulous data processing approach aims to enhance the quality and suitability of the dataset for subsequent regression analysis in the context of the conference theme.

Our data processing contains the following steps:

- fixing rows that contains null or NAN values
- changing columns from raw data to clear numeric values
- creating new columns based on remaining features
- changing scale of the numeric features

In our initial step, a comprehensive analysis of each row in the datasets was conducted to identify and address potential null values. Subsequently, we adopted a strategy of imputing missing data for all facilities by utilizing the mean of the preceding value. This approach led to successful data completion across every row and column in the dataset.

Moving on to the subsequent step, it became evident that certain columns were incompatible with the regression model in use. To rectify this issue, we undertook a transformation of these columns by converting their feature types into numeric. This systematic adjustment ensured that all columns within the dataset were represented in a numeric format, aligning with the requirements of the regression model.

Continuing our processing analysis, we employed feature engineering methods to derive new features based on the existing dataset. Code Listing 1.3 on appendixA illustrates this process, wherein columns such as 'line count' were used to create a new feature ('total line count') based on the summarization of line counts. Additionally, the average of 'flow count2' and 'flow count4' was computed using 'flow count' columns. Moreover, a new feature was generated by multiplying the 'wait' feature by 'early waiting.' Finally, to enhance interpretability and compatibility with our regression models, all numeric features underwent scaling and transformation.

In conclusion, our data preprocessing endeavors were guided by a meticulous approach aimed at ensuring the completeness, compatibility, and informativeness of the dataset for regression analysis. The initial step involved a comprehensive analysis and imputation strategy for handling null values, resulting in a robust dataset filled with meaningful information. Recognizing the need for alignment with the regression model, we transformed specific columns into numeric format, fostering consistency throughout the dataset. The subsequent phase of feature engineering introduced new dimensions to the dataset, further enhancing its utility.

By creating features based on existing attributes and employing scaling techniques, we have not only optimized the dataset for interpretability but also fortified its readiness for regression modeling. This comprehensive methodology underscores the importance of thoughtful data preprocessing in paving the way for robust and reliable regression analyses in diverse contexts.

3.3 Model and Algorithm Structure

In pursuit of effective model training, the requisite data preprocessing detailed in the preceding Section 3.2 was imperative. The development of regression models, as delineated in our proposed methodology, posed a multifaceted challenge. Within our analytical framework for predicting facility data, we judiciously employed four distinct regression algorithms, each contributing uniquely to our investigative pursuits:

- Linear regression algorithm[19] a cornerstone in statistical modeling, was strategically harnessed to capture and interpret linear relationships within the processed facility data.
- Incorporating decision tree principles[18] this regression algorithm facilitated the delineation of complex data patterns, offering a tree-like structure for insightful predictions.
- Leveraging XGBoost Regression[21] this regression algorithm emerged as a robust tool for addressing non-linearity and enhancing predictive accuracy through ensemble techniques.
- Recurrent Neural Network[22] that proved instrumental in capturing temporal dependencies within the facility data, utilizing sequential learning to refine predictive capabilities.

This methodical selection of regression algorithms reflects a holistic approach, considering both classical and advanced techniques, in our endeavor to analyze and predict facility data. The ensuing sections provide a detailed exposition of the methodologies employed, the intricacies of model training, and the overarching insights gained from the amalgamation of these diverse regression models. Linear Regression, as a cornerstone in statistical modeling, played a fundamental role in our analytical framework. Its simplicity and interpretability allowed us to establish a baseline understanding of linear relationships within the processed facility data. By strategically harnessing Linear Regression, we gained valuable insights into the direct impact of

individual features on the target variable. The model's coefficients provided quantifiable measures, aiding in the transparent interpretation of linear trends observed in the dataset. Despite its simplicity as you can see in the following equation 1, Linear Regression proved to be an essential component, forming the bedrock upon which we built our comprehensive ensemble of regression models.

$$y = mx + b \quad (1)$$

Where:

y is the predicted value,
 x is the independent variable,
 m is the slope (coefficient),
 b is the intercept.

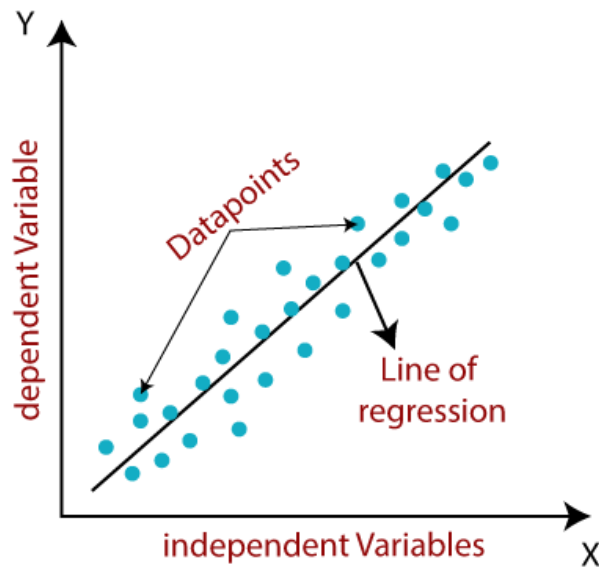


Figure 8: Linear Regression Usage

Moving beyond the foundational role of Linear Regression, Decision Tree Regression emerged as a pivotal component in our analytical arsenal. By incorporating decision tree principles, this regression algorithm adeptly addressed the inherent complexities present in the facility data. The model facilitated the delineation of intricate data patterns through a tree-like structure, enabling us to uncover nuanced relationships that might defy linear characterization.

Decision Tree Regression's ability to capture nonlinearities and interactions among variables was particularly advantageous. The interpretability of the resulting tree structure allowed us to gain actionable insights into the decision-making processes inherent in the facility dataset. As a versatile tool, Decision Tree Regression complemented the simplicity of Linear Regression, contributing to a more nuanced understanding of the multifaceted patterns governing our predictive analytics framework.

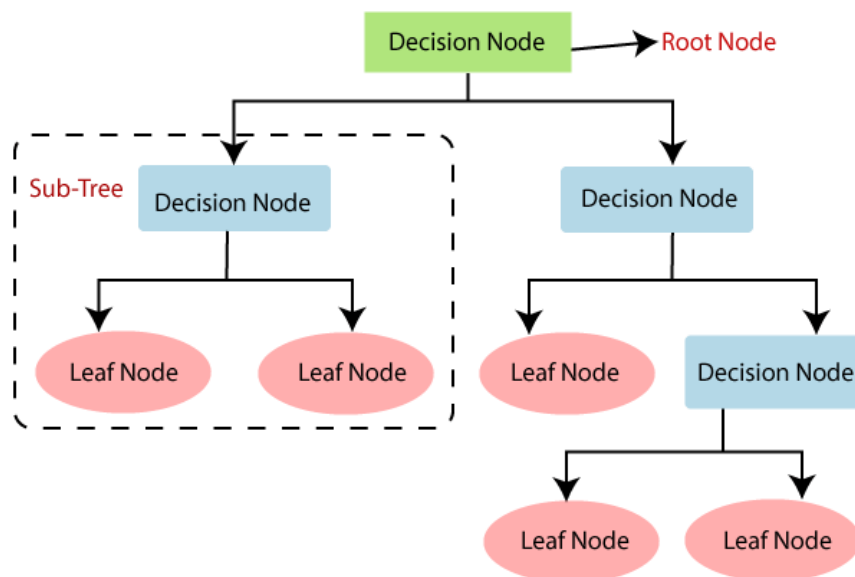


Figure 9: Decision Tree Usage

Transitioning to the XGBoost Regression model, our analytical journey gains enhanced robustness and predictive accuracy. Leveraging the powerful XGBoost framework, this regression algorithm stands out as a formidable tool in addressing non-linearity and optimizing ensemble learning. XGBoost, an abbreviation for eX-

treme Gradient Boosting, combines the strengths of multiple decision trees to create a highly adaptive and powerful predictive model.

The essence of XGBoost lies in its iterative training process as you can see in the following equation2, where weak learners (shallow decision trees) are successively added to the model, and each subsequent tree corrects the errors of the ensemble. This process is guided by gradients, emphasizing accurate predictions for previously misclassified instances. The overall prediction is then obtained by summing the contributions of all individual trees.

$$\hat{y} = \sum_{k=1}^K \eta \cdot h_k(x) \quad (2)$$

In our specific use case, XGBoost Regression demonstrated exceptional capabilities in capturing intricate patterns within the facility data. Its ensemble nature addressed overfitting and underfitting challenges, making it particularly adept at handling complex relationships and achieving high predictive accuracy.

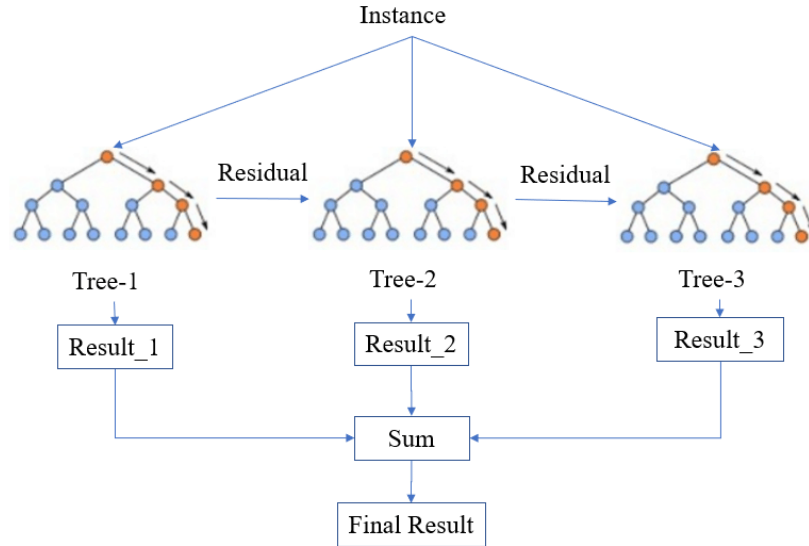


Figure 10: XGBoost Usage

Shifting our focus to the Recurrent Neural Network (RNN), we venture into the realm of deep learning methodologies, where temporal dependencies within the facility data take center stage. At the forefront of advanced modeling techniques, the RNN proves instrumental in capturing the intricate nuances of time-series data through sequential learning and memory retention as you can see in the following equation 3. The unique strength of the RNN lies in its ability to recognize patterns evolving over time, a crucial aspect in our time-series prediction task. The model achieves this by incorporating feedback loops that allow information from previous time steps to influence predictions at subsequent steps. This sequential learning enables the RNN to discern temporal dependencies within the facility data, making it a natural fit for applications where historical context is paramount.

$$\hat{y}_t = \sigma(W_{yh} \cdot h_t + b_y) \quad (3)$$

Where:

y_t is the predicted output at time,

h_t is the hidden state of the LSTM unit at time,

W_{yh} is the weight matrix connecting the hidden state to the output

b_y is the bias term.

σ is the activation function

In our specific use case, the RNN's architecture, featuring layers of Long Short-Term Memory (LSTM) units, enables it to capture and store information over extended sequences. This proves particularly beneficial in scenarios where facility data exhibits temporal dynamics and evolving patterns. The ensuing sections of our analysis delve into the intricate details of RNN methodology, highlighting considerations such as input shape, epochs, and batch size. Additionally, the evaluation of the RNN's performance provides insights into its ability to refine predictive capabilities within the ensemble. As an indispensable component within our diverse regression model ensemble, the Recurrent Neural Network contributes a layer of sophistication, addressing temporal dependencies and enriching the overall predictive capacity of our analytical framework.

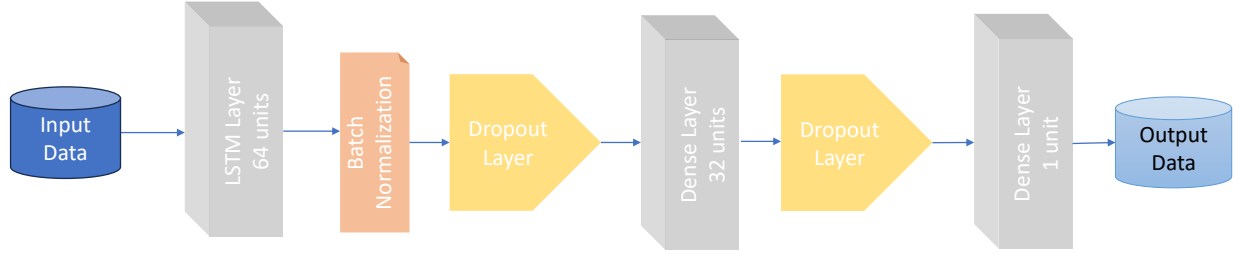


Figure 11: Architecture of Recurrent Neural Network

4 Experimental Evaluation

This section details the setup that was used for performing the experimental evaluation of the proposed methodology, as well as the results received.

4.1 Libraries and Frameworks

Building upon the robust foundation of Python[23], our methodology stands as a testament to the deliberate integration of various programming elements, ensuring a comprehensive and nuanced approach to problem-solving. The dynamic synergy of the selected libraries and frameworks not only facilitated the seamless implementation of our methodology but also played a pivotal role in enhancing its efficiency across diverse programming facets. The utilization of Python revealed, as detailed in AppendixA, a requisite dependency on specific frameworks and libraries crucial for the development of our methodology and the subsequent attainment of results. Each selected library and framework played a distinct yet indispensable role in various

facets of programming, contributing collectively to the efficacy of our approach. TensorFlow[33], a cornerstone of our framework selection, empowered our methodology with a powerful suite of tools for machine learning tasks. The integration of TensorFlow provided a robust backbone, allowing for the exploration and deployment of intricate machine learning models. Complementing TensorFlow, Keras[32] enriched our methodology by offering a high-level neural networks API. This facilitated a streamlined development process, allowing for the creation and fine-tuning of intricate neural network architectures with remarkable ease.

Numpy[24], an indispensable library for numerical computations, played a foundational role in handling complex mathematical operations. Its efficiency in array manipulation and numerical processing significantly contributed to the computational robustness of our methodology. Matplotlib [25], selected for data visualization, became an instrumental component in our methodology, providing an intuitive and visually informative representation of complex data structures. This not only aided in understanding the intricacies of our models but also facilitated effective communication of results. Further fortifying our approach, Scikit-learn[26] emerged as a valuable asset, offering a versatile toolkit for machine learning algorithms and tools. Its extensive array of functionalities, ranging from classification and regression to clustering and dimensionality reduction, broadened the scope of our methodology and its applicability to diverse problem domains.

This strategic amalgamation of diverse programming elements underscores the adaptability and versatility of our methodology. The deliberate selection of these libraries and frameworks reflects a keen awareness of the multifaceted nature of contemporary problem-solving, ensuring that our methodology remains at the forefront of innovation and efficacy in the ever-evolving landscape of computational research.

4.2 Experimental Setup

In the experimental phase of our investigation, we leveraged the comprehensive dataset sourced from all four facilities delineated in Section 3. This dataset furnishes intricate details concerning patient arrival and departure times in relation to their scheduled appointments. For the training aspect of our methodology, as delineated in Section 4.1, we harnessed the robust capabilities of the TensorFlow and Keras frameworks to engineer our regression models. Specifically, our focus lay on constructing a Recurrent Neural Network (RNN), predicated upon the Long Short-Term Memory (LSTM) regression model architecture elucidated in Section 3.3 which entailed the utilization of three layers for both training and validation purposes.

The computational infrastructure employed for this endeavor comprised a high-

performance computing system featuring an i7-12700H CPU boasting 14 cores and 16 gigabytes of RAM, supplemented by a GeForce RTX 3050 Ti GPU furnished with a 4-gigabyte memory card. Notably, in addition to local training procedures, we extended our computational framework to encompass web-based training modalities via the Saturn Cloud platform. This expansion afforded us access to an augmented computational environment equipped with 8 CPU cores and 64 gigabytes of RAM, ultimately yielding commendable outcomes in terms of both R-squared values, as delineated in Table 3 and Mean Absolute Error, as presented in Table 2. The discernible enhancement in performance underscores the efficacy of our approach and the scalability of our model across diverse computational platforms.

4.3 Experimental Results

The performance of the regression models in predicting patient arrival and departure times across various facilities as being explained in Figure 12 was assessed using the Mean Absolute Error MAE equation4 metric. MAE measures the average absolute difference between the predicted and actual values, providing insights into the accuracy of the models' predictions. Table 2 presents the MAE values for each regression model across different facilities.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i| \quad (4)$$

In the equation :

MAE : Mean Absolute Error

n : number of observations

$\hat{y}_i$: predicted value for observation i

y_i : actual value for observation i

Across all facilities, the MAE values for the Linear Regression model ranged from 1.72 to 11.46 units of time. These results indicate varying levels of deviation between the predicted and actual values, with higher MAE values suggesting less accurate predictions. In contrast, the Decision Tree models exhibited MAE values ranging from 0.59 to 5.87 across the facilities. These models generally demonstrated lower MAE values compared to Linear Regression, indicating potentially better accuracy in capturing patient scheduling dynamics.

Similarly, the Boosted Tree models showed MAE values ranging from 0.31 to 4.36, consistently demonstrating superior accuracy compared to Linear Regression and

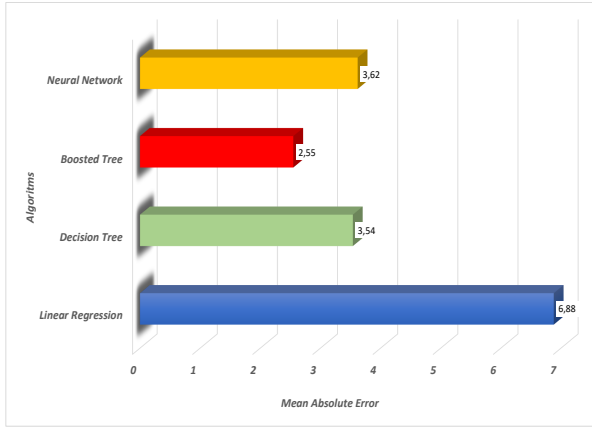
Decision Tree models. The lower MAE values suggest that the Boosted Tree models were more effective in predicting patient arrival and departure times across all facilities.

Meanwhile, the Neural Network models exhibited MAE values ranging from 0.96 to 7.13. Although generally lower than Linear Regression, these values were slightly higher compared to Decision Tree and Boosted Tree models, indicating a moderate level of accuracy in predicting patient scheduling dynamics.

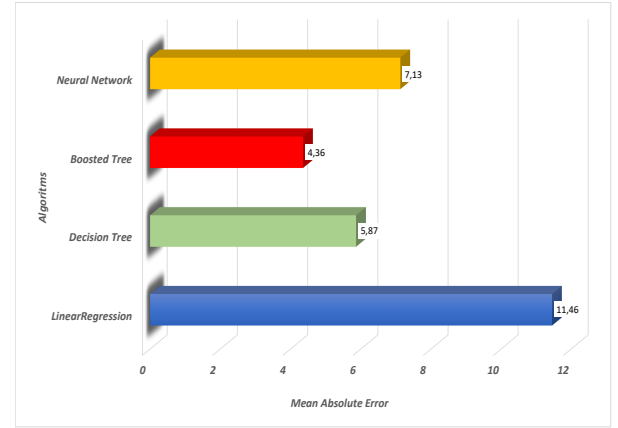
Notably, Facility 4 consistently exhibited the lowest MAE across all models, suggesting superior predictive performance in this facility. These results underscore the effectiveness of decision tree and boosted tree models in capturing the complex dynamics of patient scheduling across diverse healthcare facilities.

Table 2: Mean Absolute Error Performance

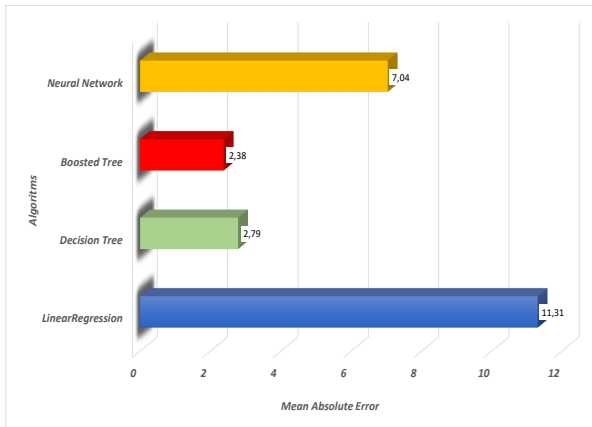
Algorithms	Facility 1	Facility 2	Facility 3	Facility 4
Linear Regression	6.88	11.46	11.31	1.72
Decision Tree	3.54	5.87	2.79	0.59
Boosted Tree	2.55	4.36	2.38	0.31
Neural Network	3.62	7.13	7.04	0.96



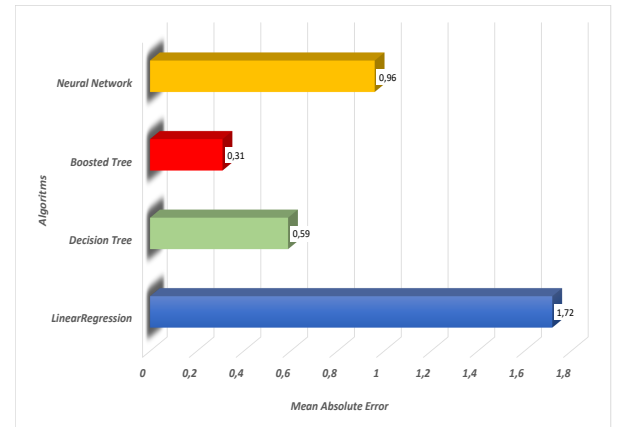
(a)



(b)



(c)



(d)

Figure 12: Mean Absolute Error Results (a) Facility 1 (F1). (b) Facility 2 (F2). (c) Facility 3 (F3). (d) Facility 4 (F4).

The analysis of R-squared equation 5 results provides insights into the performance and explanatory capability of various regression models employed to predict patient arrival and departure times across different facilities.

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (5)$$

In the equation :

n : number of observations

$\hat{y}_i$: predicted value for observation i

y_i : actual value for observation i

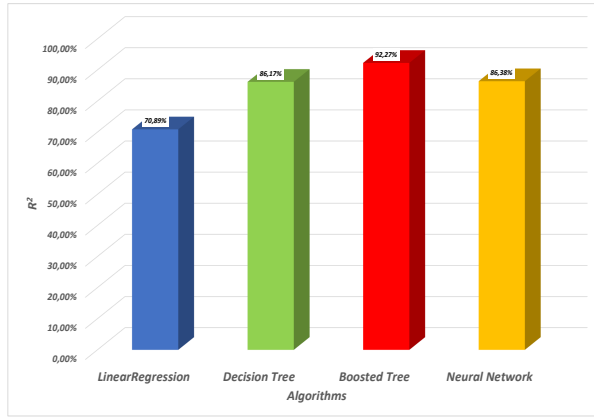
$\bar{y}$: mean of the actual values

Notably, the Boosted Tree models consistently exhibit the highest R-squared values across all facilities, indicating their superior ability to capture and explain the variance in patient scheduling dynamics. With R-squared values ranging from 92.27% to 98.16%, the Boosted Tree models demonstrate a remarkable capacity to account for the variability in arrival and departure times, showcasing their efficacy in modeling complex relationships within the dataset. Following closely, the Decision Tree models also present robust performance, albeit with slightly lower but still substantial R-squared values ranging from 80.67% to 96.19%. This suggests that Decision Tree models effectively capture a significant portion of the variance in patient scheduling patterns, making them viable alternatives for predictive modeling in healthcare scheduling contexts. Conversely, Linear Regression and Neural Network models exhibit comparatively lower R-squared values across the facilities, indicating a relatively weaker explanatory power in capturing the underlying patterns within the data. Despite this, their R-squared values, ranging from 69.11% to 88.09% for Linear Regression and 79.04% to 94.26% for Neural Network models, still demonstrate a notable capability to explain a substantial proportion of the variability in patient arrival and departure times.

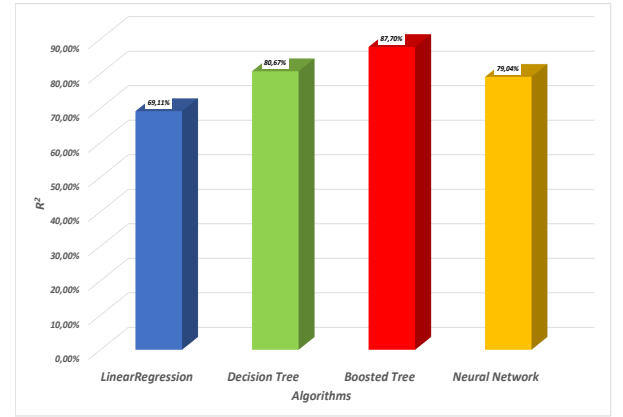
These findings underscore the importance of selecting appropriate regression techniques tailored to the complexities of the dataset and the specific predictive objectives, with Boosted Tree models emerging as particularly promising candidates for accurate modeling of patient scheduling dynamics.

Table 3: R-Squared Performance

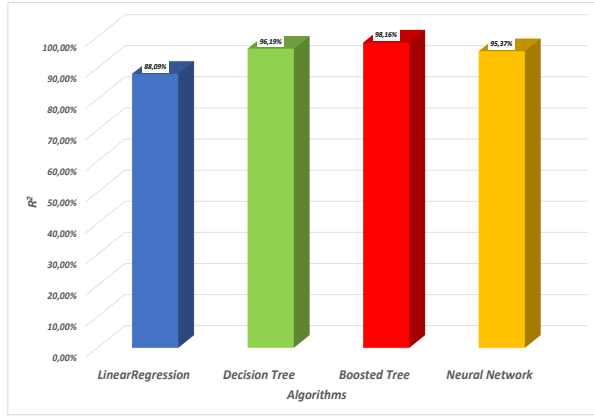
Algorithms	Facility 1	Facility 2	Facility 3	Facility 4
Linear Regression	70.89%	69.11%	88.09%	84.33%
Decision Tree	86.17%	80.67%	96.19%	94.64%
Boosted Tree	92.27%	87.70%	98.16%	97.39%
Neural Network	86.38%	79.04%	95.37%	94.26%



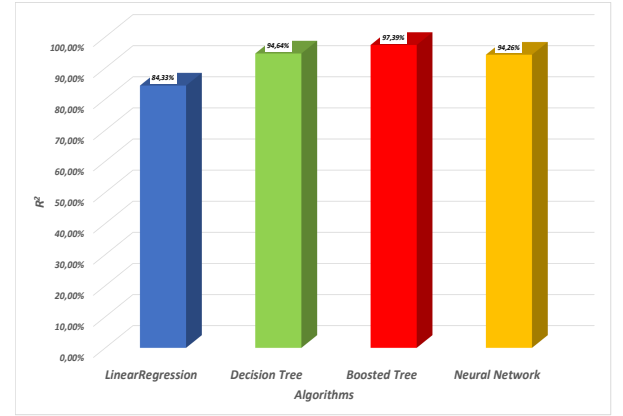
(a)



(b)



(c)



(d)

Figure 13: R-squared Results (a) Facility 1 (F1). (b) Facility 2 (F2). (c) Facility 3 (F3). (d) Facility 4 (F4).

5 Proposed Application

The results highlight the exceptional performance of Boosted Tree models in capturing and explaining the variance in patient scheduling dynamics across all facilities. Leveraging advanced machine learning technologies, these models offer a powerful tool for accurate predictive modeling in healthcare scheduling. To translate these insights into actionable solutions, we propose the development of a predictive scheduling software application.

Our proposed application aims to revolutionize healthcare scheduling processes by leveraging advanced machine learning algorithms to accurately predict patient arrival and departure times. This software aims to facilitate seamless appointment scheduling and management for both patients and doctors. By providing a user-friendly interface and powerful features, the platform enables patients to schedule appointments with their preferred doctors and allows doctors to efficiently organize their schedules. We prioritize user-centric design and functionality to meet the needs of both patients and healthcare providers. The following key features have been meticulously crafted to enhance the scheduling experience, streamline workflows, and foster improved patient care delivery:

- Patient Interface
- Doctor Interface
- Calendar Integration
- Customizable Profiles
- Secure Data Storage

The patient interface of our application offers a streamlined experience for scheduling appointments with healthcare providers. Patients can easily search for doctors based on specialty, location, availability, and other criteria. Once they find a suitable provider, they can view the available appointment slots and book appointments at their convenience. Automated appointment reminders via email or SMS help reduce no-show rates, ensuring patients attend their scheduled appointments. Additionally, patients can securely access and update their medical history within the platform, empowering healthcare providers with essential information for delivering personalized care.

For healthcare providers, our platform offers comprehensive appointment management tools to organize their schedules effectively. Doctors can view their appointment calendars, manage their availability, and accept or reschedule appointments as

needed. Access to patient profiles and medical histories allows doctors to prepare for appointments and provide tailored care. The platform also includes built-in communication tools, enabling seamless interactions between doctors and patients for quick consultations and follow-ups.

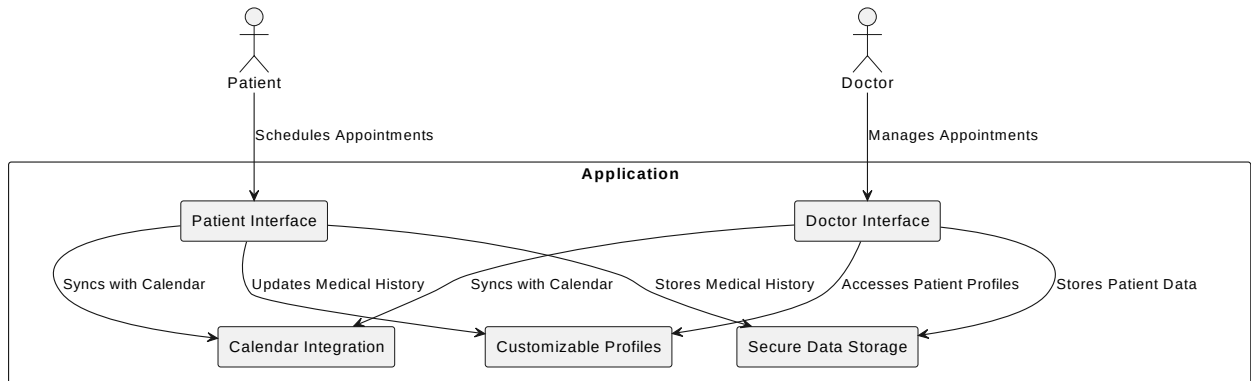


Figure 14: Proposed Application Use Case

To streamline scheduling processes, our platform integrates seamlessly with popular calendar applications such as Google Calendar or Outlook. This ensures that doctors' schedules are synchronized across platforms, minimizing scheduling conflicts and providing a unified view of appointments.

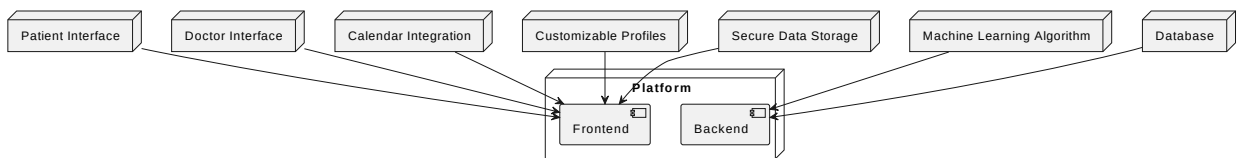


Figure 15: Architecture of Proposed Application

Both patients and doctors can create customizable profiles within the platform, providing essential information such as contact details, specialties, availability, and practice locations. These profiles help patients make informed decisions when selecting healthcare providers and enable doctors to showcase their expertise and credentials. Data security and privacy are paramount in healthcare applications. Our platform implements robust security measures to ensure the confidentiality and integrity of patient data.

Our development roadmap consists of iterative phases, starting from the core platform development to deployment, testing, and continuous improvement. By following this structured approach, we aim to deliver a high-quality and reliable patient-doctor appointment scheduling platform that meets the needs of both patients and healthcare providers.

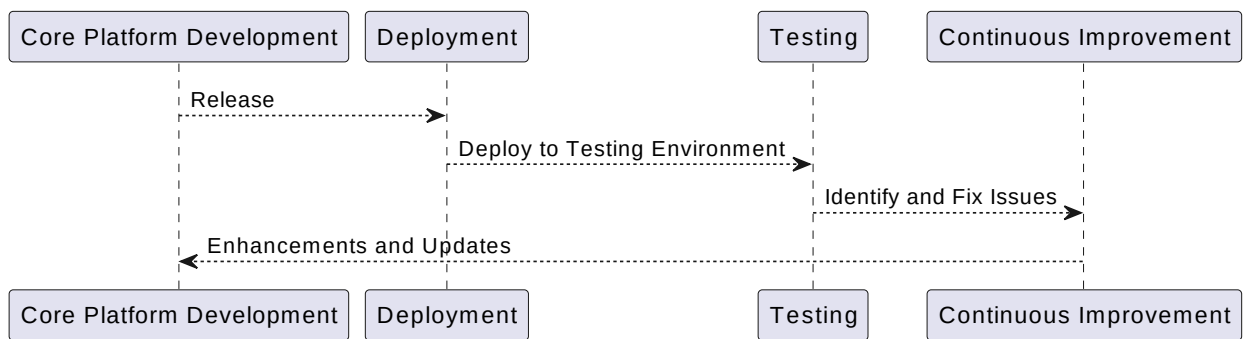


Figure 16: Deployment Roadmap

6 Conclusion and Future Work

In conclusion, this article has explored the specific suitability of machine learning in addressing operational challenges within healthcare settings. By examining the advantages and limitations of various machine learning algorithms, we have provided insights into how these technologies can be effectively applied to minimize patient waiting times and optimize overall healthcare operations.

Moving forward, future work will involve further research and development to refine the structured method of application outlined in this article. This includes addressing the limitations of machine learning usage in healthcare, such as data privacy concerns and interpretability issues. Additionally, efforts will focus on implementing and evaluating machine learning solutions in real-world healthcare environments to assess their impact on operational efficiency and patient outcomes.

Ultimately, by continuing to leverage machine learning as a transformative force in healthcare operations, we can strive towards creating more efficient and patient-centric healthcare systems. Through ongoing collaboration between researchers, healthcare practitioners, and technology experts, we can unlock the full potential of machine learning to improve healthcare delivery and enhance patient experiences.

References

- [1] Pianykh, O.S., Guitron, S., Parke, D., Zhang, C., Pandharipande, P., Brink, J. and Rosenthal, D., 2020. Improving healthcare operations management with machine learning. *Nature Machine Intelligence*, **2**(5), pp.266-273.
- [2] Kurdi, B., E. Alquqa, H. Alzoubi, M. Alshurideh, and S. Al-Hawary. "The effect of process quality improvement and lean practices on competitive performance in the UAE healthcare industry." *Uncertain Supply Chain Management* 11, no. 1 (2023): 261-266.
- [3] Rejeb, Abderahman, Karim Rejeb, Horst Treiblmaier, Andrea Appolloni, Salem Alghamdi, Yaser Alhasawi, and Mohammad Iranmanesh. "The Internet of Things (IoT) in healthcare: Taking stock and moving forward." *Internet of Things* (2023): 100721.
- [4] King, Zella, Joseph Farrington, Martin Utley, Enoch Kung, Samer Elkhodair, Steve Harris, Richard Sekula, Jonathan Gillham, Kezhi Li, and Sonya Crowe. "Machine learning for real-time aggregated prediction of hospital admission for emergency patients." *NPJ Digital Medicine* 5, no. 1 (2022): 104.
- [5] Alanazi, Abdullah. "Using machine learning for healthcare challenges and opportunities." *Informatics in Medicine Unlocked* 30 (2022): 100924.
- [6] Cardoso, M. Jorge, Wenqi Li, Richard Brown, Nic Ma, Eric Kerfoot, Yiheng Wang, Benjamin Murrey et al. "Monai: An open-source framework for deep learning in healthcare." *arXiv preprint arXiv:2211.02701* (2022).
- [7] Thilagam, K., A. Beno, M. Vanitha Lakshmi, C. Bazil Wilfred, Santhi M. George, M. Karthikeyan, Vijayakumar Peroumal, C. Ramesh, and Prabakaran Karunakaran. "Secure IoT Healthcare Architecture with Deep Learning-Based Access Control System." *Journal of Nanomaterials* 2022 (2022).
- [8] Jimma, Bahiru Legesse. "Artificial intelligence in healthcare: A bibliometric analysis." *Telematics and Informatics Reports* (2023): 100041.
- [9] Zini, Marco, and Carlo Carcasci. "Machine learning-based monitoring method for the electricity consumption of a healthcare facility in Italy." *Energy* 262 (2023): 125576.

- [10] Pallathadka, Harikumar, Malik Mustafa, Domenic T. Sanchez, Guna Sekhar Sajja, Sanjeev Gour, and Mohd Naved. "Impact of machine learning on management, healthcare and agriculture." *Materials Today: Proceedings* 80 (2023): 2803-2806.
- [11] Bastani, Hamsa, Dennis J. Zhang, and Heng Zhang. "Applied machine learning in operations management." *Innovative Technology at the Interface of Finance and Operations: Volume I* (2022): 189-222.
- [12] Javaid, Mohd, Abid Haleem, Ravi Pratap Singh, Rajiv Suman, and Shanay Rab. "Significance of machine learning in healthcare: Features, pillars and applications." *International Journal of Intelligent Networks* 3 (2022): 58-73.
- [13] Cho, Young Sik, and Paul C. Hong. "Applying Machine Learning to Healthcare Operations Management: CNN-Based Model for Malaria Diagnosis." In *Healthcare*, vol. 11, no. 12, p. 1779. MDPI, 2023.
- [14] Shehadeh, Ali, Odey Alshboul, Rabia Emhamed Al Mamlook, and Ola Hamedat. "Machine learning models for predicting the residual value of heavy construction equipment: An evaluation of modified decision tree, LightGBM, and XGBoost regression." *Automation in Construction* 129 (2021): 103827.
- [15] Rahman, Md Mostafizur, and Sumiya Akter Nisher. "Predicting Average Localization Error of Underwater Wireless Sensors via Decision Tree Regression and Gradient Boosted Regression." In *Proceedings of International Conference on Information and Communication Technology for Development: ICICTD 2022*, pp. 29-41. Singapore: Springer Nature Singapore, 2023.
- [16] Ozcan, Mert, and Serhat Peker. "A classification and regression tree algorithm for heart disease modeling and prediction." *Healthcare Analytics* 3 (2023): 100130.
- [17] Terauchi Y, Takamoto I, Kubota N, Matsui J, Suzuki R, Komeda K, Hara A, Toyoda Y, Miwa I, Aizawa S, Tsutsumi S, Tsubamoto Y, Hashimoto S, Eto K, Nakamura A, Noda M, Tobe K, Aburatani H, Nagai R, Kadowaki T. Glucokinase and IRS-2 are required for compensatory beta cell hyperplasia in response to high-fat diet-induced insulin resistance. *J Clin Invest* [Internet]. 2007 Jan 2 [cited 2007 Jan 5];117(1):246-57. Available from: <http://www.jci.org.ezproxy-v.musc.edu/cgi/content/full/117/1/246> doi:10.1172/JCI17645

- [18] Elisabetta Benevento, Davide Aloini, Nunzia Squicciarini, Towards a real-time prediction of waiting times in emergency departments: A comparative analysis of machine learning techniques, *International Journal of Forecasting*, Volume 39, Issue 1, 2023, Pages 192-208, ISSN 0169-2070, <https://doi.org/10.1016/j.ijforecast.2021.10.006>.
- [19] Ozcan, M., Peker, S. (2023). A classification and regression tree algorithm for heart disease modeling and prediction. *Healthcare Analytics*, 3, 100130.
- [20] James, G., Witten, D., Hastie, T., Tibshirani, R., Taylor, J. (2023). Linear regression. In *An introduction to statistical learning: With applications in python* (pp. 69-134). Cham: Springer International Publishing.
- [21] Nguyen, H., Cao, M. T., Tran, X. L., Tran, T. H., Hoang, N. D. (2023). A novel whale optimization algorithm optimized XGBoost regression for estimating bearing capacity of concrete piles. *Neural Computing and Applications*, 35(5), 3825-3852.
- [22] Alam, M. S., Mohamed, F. B., Selamat, A., Hossain, A. B. (2023). A review of recurrent neural network based camera localization for indoor environments. *IEEE Access*.
- [23] Van Rossum G, Drake FL. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace; 2009.
- [24] Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. *Nature* 585, 357–362 (2020). DOI: 10.1038/s41586-020-2649-2.
- [25] J. D. Hunter, "Matplotlib: A 2D Graphics Environment", *Computing in Science Engineering*, vol. 9, no. 3, pp. 90-95, 2007
- [26] Scikit-learn: Machine Learning in Python, Pedregosa et al., *JMLR* 12, pp. 2825-2830, 2011.
- [27] Venkatesh, Dr A. Narasima. "Reimagining the future of healthcare industry through internet of medical things (IoMT), artificial intelligence (AI), machine learning (ML), big data, mobile apps and advanced sensors." *Artificial Intelligence (AI), Machine Learning (ML), Big Data, Mobile Apps and Advanced Sensors* (October 28, 2019) (2019).

- [28] Alugubelli, Raghunandan. "Exploratory study of artificial intelligence in health-care." *International Journal of Innovations in Engineering Research and Technology* 3, no. 1 (2016): 1-10.
- [29] Prashanth, B. U. V., Mohammed Riyaz Ahmed, and Manjunath R. Kounte. "Design and implementation of DA FIR filter for bio-inspired computing architecture." *International Journal of Electrical and Computer Engineering* 11, no. 2 (2021): 1709.
- [30] Wei, Yuanyuan, Julian Jang-Jaccard, Wen Xu, Fariza Sabrina, Seyit Camtepe, and Mikael Boulic. "LSTM-autoencoder-based anomaly detection for indoor air quality time-series data." *IEEE Sensors Journal* 23, no. 4 (2023): 3787-3800.
- [31] McLaughlin, Daniel B. *Healthcare operations management*. AUPHA, 2008.
- [32] Gulli, Antonio, and Sujit Pal. *Deep learning with Keras*. Packt Publishing Ltd, 2017.
- [33] Singh, Pramod, Avinash Manure, Pramod Singh, and Avinash Manure. "Introduction to tensorflow 2.0." *Learn TensorFlow 2.0: Implement Machine Learning and Deep Learning Models with Python* (2020): 1-24.
- [34] Bisong, Ekaba, and Ekaba Bisong. "Matplotlib and seaborn." *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners* (2019): 151-165.
- [35] Singh, Bikesh Kumar, and G. R. Sinha. "Machine Learning in Healthcare: Fundamentals and Recent Applications." (2022).
- [36] Panesar, Arjun. *Machine learning and AI for healthcare*. Coventry, UK: Apress, 2019.
- [37] Jain, LakhmiC. "Artificial Intelligence and Machine Learning for Healthcare." (2023).
- [38] Agrawal, Rashmi, Jyotir Moy Chatterjee, Abhishek Kumar, Pramod Singh Rathore, and Dac-Nhuong Le, eds. *Machine Learning for Healthcare: Handling and Managing Data*. CRC Press, 2020.
- [39] Singh, Krishna Kant, Mohamed Elhoseny, Akansha Singh, and Ahmed A. Elngar, eds. *Machine learning and the internet of medical things in healthcare*. Academic Press, 2021.

- [40] Mangla, Monika, Subhash K. Shinde, Vaishali Mehta, Nonita Sharma, and Sachi Nandan Mohanty, eds. Handbook of Research on Machine Learning: Foundations and Applications. CRC Press, 2022.
- [41] Watt, Jeremy, Reza Borhani, and Aggelos K. Katsaggelos. Machine learning refined: Foundations, algorithms, and applications. Cambridge University Press, 2020.
- [42] Holzinger, Andreas. Machine learning for health informatics. Springer International Publishing, 2016.
- [43] Shaikh, T.A., Hakak, S., Rasool, T., Wasid, M. (Eds.). (2023). Machine Learning and Artificial Intelligence in Healthcare Systems: Tools and Techniques (1st ed.). CRC Press. <https://doi.org/10.1201/9781003265436>
- [44] Santosh, K. C., and Loveleen Gaur. Artificial intelligence and machine learning in public healthcare: Opportunities and societal impact. Springer Nature, 2022.
- [45] Malik, Hasmat, Nuzhat Fatema, and Jafar A. Alzubi, eds. AI and machine learning paradigms for health monitoring system: intelligent data analytics. Vol. 86. Springer Nature, 2021.
- [46] Chen, Yen-Wei, and Lakhmi C. Jain. "Deep learning in healthcare." Paradigms and applications. Springer, Heidelberg (2020).
- [47] Natarajan, Prashant, John C. Frenzel, and Detlev H. Smaltz. Demystifying big data and machine learning for healthcare. CRC Press, 2017.
- [48] Bhoi, Akash Kumar, Pradeep Kumar Mallick, Chuan-Ming Liu, and Valentina Emilia Balas, eds. Bio-inspired neurocomputing. Vol. 310. Berlin/Heidelberg, Germany: Springer, 2021.
- [49] Gong, Maoguo, Linqiang Pan, Tao Song, and Gexiang Zhang, eds. Bio-inspired Computing—Theories and Applications. Singapore: Springer, 2016.
- [50] Deisenroth, Marc Peter, A. Aldo Faisal, and Cheng Soon Ong. Mathematics for machine learning. Cambridge University Press, 2020.
- [51] Lee, Cheng Few, and John C. Lee, eds. Handbook Of Financial Econometrics, Mathematics, Statistics, And Machine Learning (In 4 Volumes). World Scientific, 2020.

- [52] Clarke, Bertrand, Ernest Fokoue, and Hao Helen Zhang. Principles and theory for data mining and machine learning. New York: Springer, 2009.
- [53] Chong, Edwin KP, Wu-Sheng Lu, and Stanislaw H. Zak. An Introduction to Optimization: With Applications to Machine Learning. John Wiley and Sons, 2023.
- [54] Heng, Liao, and Bill McColl, eds. Mathematics for future computing and communications. Cambridge University Press, 2021.
- [55] Schmidt, Gunther. Relational mathematics. Vol. 132. Cambridge University Press, 2011.
- [56] Brunton, Steven L., and J. Nathan Kutz. Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press, 2019.
- [57] Müller, Andreas C., and Sarah Guido. Introduction to machine learning with Python: a guide for data scientists. " O'Reilly Media, Inc.", 2016.
- [58] Rubinstein, Reuven Y., and Dirk P. Kroese. The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning. Vol. 133. New York: Springer, 2004.
- [59] Khang, Alex, Geeta Rana, R. K. Tailor, and Vugar Abdullayev, eds. "Data-Centric AI Solutions and Emerging Technologies in the Healthcare Ecosystem." (2023).
- [60] Khang, Alex, Shashi Kant Gupta, Sita Rani, and Dimitrios A. Karras, eds. Smart Cities: IoT Technologies, big data solutions, cloud platforms, and cybersecurity techniques. CRC Press, 2023.

A Appendix : Coding Details of Methodology

```
1 from data_loader.load_data import load_data
2 from data_processing.feature_engineering import feature_engineering
3 from data_processing.feature_engineering_f4 import
  feature_engineering_f4
4 from model_builder.build_model import build_model
5 from model_evaluation.evaluate_model import evaluate_model
6 import os
7
8
9 def process_dataframe(sheet_name, feature_engineering_func):
10     # Get the directory of the current script
11     script_directory = os.path.dirname(os.path.realpath(__file__))
12
13     # Construct the full path to the Excel file within the
14     data_loader folder
15     file_path = os.path.join(script_directory,
16                               "data_loader", "WaitData.Published.xlsx")
17     df = load_data(file_path, sheet_name=sheet_name)
18     df = feature_engineering_func(df)
19     best_models, X_train, X_test, y_train, y_test,
20     feature_names = build_model(df)
21     evaluate_model(best_models, X_test, y_test, feature_names)
22
23 def main():
24     process_dataframe("F1", feature_engineering)
25     process_dataframe("F2", feature_engineering)
26     process_dataframe("F3", feature_engineering)
27     process_dataframe("F4", feature_engineering_f4)
28
29 if __name__ == "__main__":
30     main()
```

Listing 1: main.py

```
1 import pandas as pd
2
3 def load_data(file_name, sheet_name):
4     df = pd.read_excel(file_name, sheet_name=sheet_name)
5     if df.isnull().values.any():
6         print("Missing values detected. Handling using mean
7         imputation...")
8         df.fillna(df.mean(), inplace=True)
9     return df
```

Listing 2: data loader.py

```

1 import pandas as pd
2 import numpy as np
3 from sklearn.preprocessing import RobustScaler
4
5 def feature_engineering(df):
6     df['Arrival_Hour'] = df['x_ArrivalDTTM'].dt.hour
7     df['Arrival_DayOfWeek'] = df['x_ArrivalDTTM'].dt.dayofweek
8     df['Scheduled_Arrival_Duration'] = (df['x_ArrivalDTTM'] - df['x_ScheduledDTTM']).dt.total_seconds()
9     df['Total_LineCount'] = df[['LineCount0', 'LineCount1', 'LineCount2', 'LineCount3', 'LineCount4']].sum(axis=1)
10    df['Avg_FlowCount'] = df[['FlowCount2', 'FlowCount4']].mean(axis=1)
11    df['Wait_Multiplied_AvgWait'] = df['Wait'] * df['AvgHowEarlyWaiting']
12    df['IsWeekend'] = (df['Arrival_DayOfWeek'] >= 5).astype(int)
13
14    scaler = RobustScaler()
15    numeric_features = ['SumHowEarlyWaiting', 'AvgHowEarlyWaiting', 'Scheduled_Arrival_Duration', 'Total_LineCount', 'Avg_FlowCount', 'Wait_Multiplied_AvgWait']
16    df[numeric_features] = scaler.fit_transform(df[numeric_features])
17
18    return df

```

Listing 3: feature engineering.py

```

1 import pandas as pd
2 import numpy as np
3 from sklearn.preprocessing import RobustScaler
4
5 def feature_engineering_f4(df):
6     df['Arrival_Hour'] = df['x_ArrivalDTTM'].dt.hour
7     df['Arrival_DayOfWeek'] = df['x_ArrivalDTTM'].dt.dayofweek
8     df['Scheduled_Arrival_Duration'] = (df['x_ArrivalDTTM'] - df['x_ScheduledDTTM']).dt.total_seconds()
9     df['SumCountScanned'] = (df['LineCount0'] + df['LineCount1'] + df['LineCount2'] + df['LineCount3'] + df['LineCount4']) * df['NumScannersUsedToday']
10    df['SumScannersUsed'] = df['SumWaits'] / df['NumScannersUsedToday']
11
12    df['Total_LineCount'] = df[['LineCount0', 'LineCount1', 'LineCount2', 'LineCount3', 'LineCount4']].sum(axis=1)
13    df['Avg_FlowCount'] = df[['FlowCount2', 'FlowCount4']].mean(axis=1)

```

```

14     df['Wait_Multiplied_AvgWait'] = df['Wait'] * df['
    AvgWaitLastK1Customers']
15
16     df['IsWeekend'] = (df['DayOfWeek'] >= 5).astype(int)
17
18     df.replace([np.inf, -np.inf], np.nan, inplace=True)
19     df.fillna(0, inplace=True)
20
21     scaler = RobustScaler()
22     numeric_features = ['Total_LineCount', 'Avg_FlowCount', '
    Wait_Multiplied_AvgWait']
23     df[numeric_features] = scaler.fit_transform(df[numeric_features
    ])
24
25     return df

```

Listing 4: feature engineering f4.py

```

1  import pandas as pd
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from sklearn.model_selection import train_test_split,
    RandomizedSearchCV
5  from sklearn.linear_model import LinearRegression
6  from xgboost import XGBRegressor
7  from sklearn.tree import DecisionTreeRegressor
8  from sklearn.metrics import mean_squared_error, mean_absolute_error,
    r2_score
9  from sklearn.preprocessing import StandardScaler, PolynomialFeatures,
    RobustScaler
10 from tensorflow.keras.models import Sequential
11 from tensorflow.keras.layers import LSTM, Dense, Dropout,
    BatchNormalization
12 from tensorflow.keras.callbacks import EarlyStopping
13 from sklearn.base import BaseEstimator, RegressorMixin
14 from sklearn.utils.validation import check_X_y, check_array
15 from model_builder.rnn import RNN
16
17
18 def build_model(dataframe):
19     X = dataframe.drop(['x_ArrivalDTTM', 'x_ScheduledDTTM', '
    x_BeginDTTM', 'Wait'], axis=1)
20     y = dataframe['Wait']
21     X_train, X_test, y_train, y_test = train_test_split(X, y,
    test_size=0.2, random_state=2)
22
23     scaler = RobustScaler()

```

```

24 X_train_scaled = scaler.fit_transform(X_train)
25 X_test_scaled = scaler.transform(X_test)
26
27 regression_models = [
28     {'model': LinearRegression(), 'param_dist': {'fit_intercept':
: [True, False]}},
29     {'model': XGBRegressor(), 'param_dist': {'max_depth': [3, 5,
7], 'learning_rate': [0.01, 0.1, 0.2], 'n_estimators': [50, 100,
200]}},
30     {'model': DecisionTreeRegressor(), 'param_dist': {'max_depth
': [None, 5, 10, 15], 'min_samples_split': [2, 5, 10], '
min_samples_leaf': [1, 2, 4]}},
31     {'model': RNN(input_shape=(1, X_train.shape[1]), epochs=30,
batch_size=64), 'param_dist': {}}
32 ]
33
34 best_models = {}
35
36 for model_info in regression_models:
37     model_name = model_info['model'].__class__.__name__
38     print(f"Model: {model_name}")
39
40     model = model_info['model']
41     param_dist = model_info['param_dist']
42
43     if param_dist:
44         n_iter = min(np.prod([len(v) for v in param_dist.values
: [10, 100, 200]}}, 10)
45         random_search = RandomizedSearchCV(estimator=model,
param_distributions=param_dist, scoring='r2', cv=10, n_iter=
n_iter)
46         random_search.fit(X_train_scaled, y_train)
47         best_model = random_search.best_estimator_
48     else:
49         best_model = model.fit(X_train_scaled, y_train)
50
51     best_models[model_name] = best_model
52
53     return best_models, X_train_scaled, X_test_scaled, y_train,
y_test, X.columns

```

Listing 5: build model.py

```

1 from sklearn.base import BaseEstimator, RegressorMixin
2 from sklearn.utils.validation import check_X_y, check_array
3 from sklearn.preprocessing import StandardScaler
4 from tensorflow.keras.models import Sequential

```

```

5 from tensorflow.keras.layers import LSTM, Dense, Dropout,
   BatchNormalization
6 from tensorflow.keras.callbacks import EarlyStopping
7
8 class RNN(BaseEstimator, RegressorMixin):
9     def __init__(self, input_shape, epochs=30, batch_size=64):
10         self.input_shape = input_shape
11         self.epochs = epochs
12         self.batch_size = batch_size
13         self.model = self.create_rnn_model()
14         self.scaler = StandardScaler()
15
16     def create_rnn_model(self):
17         model = Sequential()
18         model.add(LSTM(64, activation='relu', input_shape=self.
input_shape))
19         model.add(BatchNormalization())
20         model.add(Dropout(0.5))
21         model.add(Dense(32, activation='relu'))
22         model.add(BatchNormalization())
23         model.add(Dropout(0.5))
24         model.add(Dense(1, activation='linear'))
25         model.compile(loss='mean_squared_error', optimizer='adam')
26         return model
27
28     def fit(self, X, y, validation_data=None):
29         X, y = check_X_y(X, y)
30         X_scaled = self.scaler.fit_transform(X)
31         X_resaped = X_scaled.reshape(X_scaled.shape[0], 1, X_scaled
.shape[1])
32         callbacks = [EarlyStopping(patience=5, restore_best_weights=
True)]
33         self.model.fit(X_resaped, y, epochs=self.epochs, batch_size
=self.batch_size, validation_data=validation_data, callbacks=
callbacks)
34         return self
35
36     def predict(self, X):
37         X = check_array(X)
38         X_scaled = self.scaler.transform(X)
39         X_resaped = X_scaled.reshape(X_scaled.shape[0], 1, X_scaled
.shape[1])
40         return self.model.predict(X_resaped)

```

Listing 6: rnn.py

```

1 import matplotlib
2 import matplotlib.pyplot as plt
3 import numpy as np
4 from sklearn.metrics import mean_squared_error, mean_absolute_error,
    r2_score
5 import matplotlib.backends.backend_pdf as pdf_backend
6
7 def evaluate_model(best_models, X_test, y_test, feature_names):
8     for model_name, best_model in best_models.items():
9         print(f"Evaluating {model_name}")
10
11         if hasattr(best_model, 'feature_importances_'):
12             feature_importances = best_model.feature_importances_
13             important_features_indices = np.argsort(
feature_importances)[::-1]
14             important_features_names = [feature_names[i] for i in
important_features_indices]
15             print("Important Features:")
16             for feature_name in important_features_names:
17                 print(feature_name)
18         else:
19             print("Model does not support feature_importances_")
20             if not hasattr(best_model, 'predict'):
21                 best_model.fit(X_test, y_test)
22
23             y_pred = best_model.predict(X_test)
24             mse = mean_squared_error(y_test, y_pred)
25             mae = mean_absolute_error(y_test, y_pred)
26             r2 = r2_score(y_test, y_pred)
27
28             print("Mean Squared Error:", mse)
29             print("Mean Absolute Error:", mae)
30             print("R-squared:", r2)
31             print("\n")
32             plt.xlabel('Regression Models', fontsize=12)
33             plt.ylabel('R-squared', fontsize=12)
34             plt.title('R-squared Scores for Regression Models', fontsize
=15)
35             plt.xticks(fontsize=10, rotation=45, ha="right")
36             plt.yticks(fontsize=10)
37             plt.legend()
38             plt.tight_layout()
39             plt.show()

```

Listing 7: evaluation model.py