Predicting IT Hardware Service Times and Technician Performance Using Machine Learning

by

Examination Roll: 232217

A project report submitted to the Institute of Information Technology
in partial fulfilment of the requirements for the degree of
Professional Masters in Information Technology

Supervisor: Professor Shamim Al Mamun, PhD

Institute of Information Technology
Jahangirnagar University
Savar, Dhaka-1342
October, 2025

DECLARATION

I hereby declare that this project is founded on the research conducted by our team. Citations are included for any materials sourced from other scholars. This thesis has never been presented, either fully or partially, for any academic degree previously.

———————————

Roll: 232217

CERTIFICATE

The project titled "Predicting IT Hardware Service Times and Technician Performance Using Machine Learning," submitted by Md. Afjal Hossan Shajal, ID: 232217, Session: Summer-2023, has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Professional Master's in Information Technology.

———————————

Professor Shamim Al Mamun, PhD
Supervisor

BOARD OF EXAMINERS

———————————

Dr.M.Shamim Kaiser	Coordinator
Professor, IIT, JU	PMIT Coordination Committee

———————————

Dr.Risala Tasin Khan	Director & Member
Professor, IIT, JU	PMIT Coordination Committee

———————————

Dr.Jesmin Akhter	Member
Professor, IIT, JU	PMIT Coordination Committee

———————————

K M Akkas Ali	Member
Professor, IIT, JU	PMIT Coordination Committee

———————————

Dr.Rashed Mazumder	Member & Associate Professor
Associate Professor, IIT, JU	PMIT Coordination Committee

ACKNOWLEDGEMENTS

ABSTRACT

In the digital world, modern civilization cannot be imagined without IT service management, especially in terms of hardware maintenance. Because most of the people are directly or indirectly connected to IT hardware (ITH). In this study, IT hardware service times and technicians' performances are being predicted using ML for our office. In previous studies, service times and performance in various fields such as ITSM, food delivery, hardware faults, IT help desk ticketing systems, and others have been predicted by many authors. The majority of research has used a few local attributes. Most researchers did not predict the service time and solver name in a single paper. However, there is a light of hope that, in advanced Information Technology (IT), Machine Learning (ML) is playing a very effective and important role in the diagnosis and prediction of these problems (who solved the problem and how many days are needed). In this paper, we used different types of ML algorithms like RF, XGBoost, LightGBM classifier, and GB regression by using our office IT services dataset with various attributes for the problem solver name and delivery days prediction. The accuracy of these three ML classifiers is 96%, and the regression models also achieved an average error (MSE) of 3.26 days for the delivery time. This paper differs from other studies because both targets (delivery times and solver names) are being predicted in the project report. Finally, we attempt to build a user interface utilizing the trained models' .pkl file, and the interface successfully displays these results. The user interface helps IT servicing management to predict targets accurately, faster, and in less time. If the user interface is accessible to all, it will help achieve their business goals for both small and large organisations in Bangladesh and worldwide.

GitHub link: https://github.com/afjalhossanece-creator/ITHS.git

Keywords: ITSM, ITH, ML, RF, XGBoost, LightGBM, GRB.

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
AUC	Area Under the Curve
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DT	Decision Tree
GBR	Gradient Boosting Regressor
HDD	Hard Disk Drive
IT	Information Technology
ITHSM	Hardware Service Management
ITSM	IT Service Management
KNN	K-Nearest Neighbors
LDA	Linear Discriminant Analysis
LightGBM	Light Gradient Boosting Machine
LR	Logistic Regression / Linear Regression
ML	Machine Learning
MSE	Mean Squared Error
NLP	Natural Language Processing

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER I

Introduction

1.1 Background of the Study

In any modern organization today in Bangladesh and around the world, the reliability and performance of Information Technology (IT) hardware are essential for organizational success. For a large corporate office with 4,000 users, various hardware devices, including CPUs, laptops, monitors, printers, UPS, and scanners, create a complex and demanding support ecosystem. The smoothness of an organization's operations, the productivity of its employees, and the satisfaction of its customers depend largely on a reliable IT infrastructure. However, hardware failures are unavoidable. When a piece of equipment breaks down, the organization's productivity is directly affected. Therefore, the speed and effectiveness with which the IT support team can diagnose and solve these problems are extremely important. In a developing country like ours, many IT support teams still use traditional, manual processes. When a hardware problem occurs, a service ticket is created by writing it down in a register book. This ticket then enters a queue, which is manually reviewed before being assigned to a technician. This manual assignment, often done by a helpdesk manager or a senior technician, is based on their personal knowledge of the team's skills and availability. While this process can work, it is often inefficient, especially in large organizations with a high volume of service requests. After this, the user often has to wait without a clear idea of when their problem will be solved. This reactive approach creates operational problems, delays, and a frustrating experience for employees. This research provides a plan to transform IT support from a reactive function into a proactive, data-driven, and strategic unit by using various ML algorithms. This will help the organization to achieve its overall business goals.

1.2 Problem Statement

In a large corporate environment, managing IT hardware support involves facing several challenges, especially when the hardware servicing system is operated manually. Hardware servicing tasks are assigned to technicians based on a serial number in a register book, not based on their skills. In a large corporate office with many users, manually tracking different types of products is time-consuming and leads to complex problems. As a result, problems take longer to be solved, devices are returned to users late, and frustration is experienced by them. This prevents the company from reaching its goals and can lead to financial losses. Various problems were encountered in conducting this research, such as collecting data, identifying and handling missing values, normalizing categorical values (encoding with LabelEncoder), and training the data set according to the model. This research aims to create a system using ML algorithms that can suggest the name of a specialized technician and also predict the estimated time to solve the problem, based on the specific hardware issue.

1.3 Research Objectives

The main objective of this research is to create a system using ML algorithms and an IT hardware servicing dataset to make the IT servicing process faster and easier. First, the manual dataset needs to be pre-processed to make it usable for ML algorithms. The prepared dataset will then be used to train and test ML algorithms (RFC, XGBoost, LightGBM, RF Regressor, and GB Regressor). The training will have two targets (Classification and Regression). One is to predict who will solve the problem, and the other is to predict how many days it might take to solve it. Finally, the results from both targets will be used together to create a user interface that will make the hardware support system modern, accurate, simple, and faster, and it will show us the results of the training. If this user interface can be made available to everyone, then many other offices in Bangladesh, like our corporate office, can provide fast and easy IT support services to their users. This will allow users to focus on their work, and the company can move forward in achieving its goals.

1.4 Scope of the Project

There are multiple project scopes. These scopes are given below:

Using the Manual Hardware Service dataset: This project mostly focuses on the utilization of the manual servicing dataset that was collected at our office.

This dataset includes various features (e.g., RegSerialNO, UI, PM, ReceivedDate, Problem, Solution, SolvedBY, DeliveryDate, etc.) that help us predict the target more accurately.

ML-Based Solution: Here, ML algorithms such as RF, XGBoost, and Light-GBM classifications are applied for solver prediction, and RF Regressor and Gradient Boosting Regressor are also applied for delivery days prediction.

Using Simple Tools and Techniques: In this study, open-source tools and Python-based libraries were used. These tools help me to implement the system without needing costly software with my basic programming knowledge, making the model affordable and suitable for our office and other offices in Bangladesh.

Achieving High Accuracy: After training, the ML algorithms achieved 96% accuracy in solver prediction and a 3.62-day RMSE for delivery time, which is closest to the real values. These results help to make the system faster and improve decision-making for technician workload distribution.

Future Goal and Automation: Finally, the focus is placed on building a user interface that can help automatically assign the solver name and predict delivery timelines. This opens a new door for digital systems in large corporate offices for hardware servicing environments.

1.5 Motivation

In modern offices in Bangladesh, IT hardware such as laptops, CPUs, monitors, UPS, and printers is used every day. When these devices stop working, difficulties are faced by the organization until proper servicing is completed. One of the main problems is to decide which technician should solve the case and how many days will be needed for delivery. Normally, this decision is made manually, and it often causes delays or wrong assignments. Our office IT servicing dataset contains user information, location, product model, problem, solution, solved by, received date, delivery date, and delivered days. This information gives us the chance to apply ML and make future predictions. In this project, algorithms like Random Forest, XGBoost, and LightGBM are used. These models achieved 96% accuracy in predicting the right solver and an RMSE of 4 days for delivery prediction. The motivation behind this work is to make servicing decisions faster, accurate, and reliable. If such prediction systems are used, both office management and customers will save time, reduce costs, and improve satisfaction.

1.6 Limitations

The results are encouraging; it is vital to acknowledge that this project has several limitations. The following lists these limitations:

Single Source of Data: The study is based on collecting the dataset from our corporate office. Accordingly, the study results cannot be generalized to other companies with different IT landscapes.

Limited Hardware Type: In this paper, some specific IT hardware, such as laptops, CPUs, monitors, UPS, printers, projectors, and scanners, was used. More devices, such as cameras, mobile phones, security, or network equipment, need to be included. Otherwise, the system cannot predict services outside the available hardware categories in a large office.

Specific Time Period: The records in the dataset are for a particular time. There would be no way to capture changes in policy regarding hardware that occur at any time of year other than this.

Handling New Problems: This project works on manual or historical data, but it does not give real-time predictions during the actual servicing process. For example, when a new problem comes in, the model cannot predict instantly unless the data is processed and updated.

1.7 Assumptions of the Study

There are numerous assumptions. The following are the assumptions:

Accuracy of Dataset Records: It is assumed that all the information in the dataset, such as receivedate, delivery date, solvedby, and problem type, has been correctly entered by office staff. If the records are not accurate, then the model may produce wrong predictions based on faulty learning.

Historical Patterns Will Continue: This study assumes that future service requests will follow similar patterns to the records in the dataset. The model is trained on historical data, so it is expected that the same trends will help predict upcoming service tasks accurately.

ML Model Generalization: The assumption here is that the ML models—Random Forest, XGBoost, and LightGBM—will generalize well to unseen data. With an expected accuracy of 96%, the study assumes that the models will not overfit the training data and will continue to perform well when applied to new service requests.

Reliability of Feature Selection: This project assumes that the selected fea-

tures, like location, product model, problem, and solution, are enough to explain and predict who will solve the problem and how many days it will take to deliver. No hidden feature is expected to have more influence.

Stable Data Distribution: It is assumed that the distribution of data (e.g., problem types, product models, delivery times) remains relatively stable over time. The study assumes that future service records will follow similar patterns to the past, allowing the ML models to predict service outcomes with high accuracy and strong RMSE.

1.8 Research Outline

This report is organized as follows: **Chapter I**, which gives an insight into the background, problem statement, objectives, and scope of the study. **Chapter II** reviews the literature on related research, in which the definitions of the main terms used in this thesis and the fundamental concepts, structure of ML principles, and many models are discussed. **Chapter III** describes the system model, providing details about the system architecture, dataset analysis, and the overall operational procedures of the model. Next, **Chapter IV** discusses the system tools and algorithms, highlighting the methods for feature extraction and selection, along with the tools employed for these tasks, and offers an understanding of how these methods operate within our model. Following that, **Chapter V** focuses on the simulation and evaluation of the model, assessing the results through the confusion matrix. Lastly, **Chapter VI** discusses prospective future work and presents the concluding remarks.

CHAPTER II

Literature Overview

2.1 Overview of the Chapter

This section presents a comprehensive examination of relevant research on several ML methodologies for IT Hardware Servicing. In this chapter, it has been pointed out that the main summary in earlier studies, where service outcomes ("solved by" who) and delivery times (delivered days) are described, can be easily detected using many ML algorithms like Random Forest (RF), XGBoost, and LightGBM. These studies help to understand how the authors use similar features, performance metrics, challenges, and gaps.

2.2 Predictive analytics' function in IT hardware service management.

The broad field of managing IT hardware services to satisfy an organization's hardware infrastructure, which supports business operations effectively and reliably, is known as IT Hardware Service Management (ITHSM). Incident management, or the process of resolving issues as they arise, is considered a crucial component of ITHSM. In traditional ITHSM, problems are effectively identified and ensured to be resolved, but accurate prediction of them is often not achieved. This differs from predictive analytics, which employs historical data to forecast future results. This suggests that, in the context of IT support, things like product received resolution times and solver names can be forecasted, which are close to the target of our project system.

2.3 Prediction of Service Duration and Delivery Time.

Using past data, many studies have tried to predict how long it will take from the product received date to delivering or solving it. Now an attempt is made to describe the previous studies and how the product or service delivery time is predicted by the authors. Few studies are directly related to my project paper, but most are similar tasks.

2.3.1 Case study on delivery time determination in small batch production companies

This research investigates the prediction of delivery times in small-batch production. After collecting row data, the authors cleaned it, eliminated duplicates and missing values, and then applied ML models such as XGBoost. RMSE values of 9.5 business days were obtained, but it was demonstrated that performance is improved by domain-specific features[1].

2.3.2 A Deep Learning Based IT Service Desk Ticket Classifier Using CNN.

S.P. Paramesh and K.S. Shreedhara propose a CNN-based automated ticket classification model to improve IT service desk systems. Manual ticket assignment often leads to delays, errors, and misrouting, while traditional ML models struggle with sparsity and overfitting. Their approach applies NLP preprocessing and word embeddings to ticket descriptions, with CNN layers extracting key features for accurate categorization. Tested on real IT service desk data, the model achieved 91.2% accuracy, outperforming SVM, LR, Naive Bayes, and KNN. The system reduces manual effort, improves response time, enhances customer satisfaction, and supports organizational growth [2].

2.3.3 ML in IT Service Management.

In this paper, D. Zueva, A. Kalistratov, and A. Zuev propose an ML-based model for IT Service Management (ITSM) to predict incident resolution time. By analyzing historical incident data, the model estimates the likely resolution duration, helping identify potential delays. This predictive approach improves customer experience, reduces service desk effort, and lowers service costs. Tested on a real service desk

dataset, the model demonstrates practical utility for enhancing ITSM processes and efficiently managing infrastructure incidents[3].

2.3.4 Prediction of service time for home delivery services using ML.

The authors of this study used historical order data that included information about the location, items, clients, technicians, and service type. Predictions were made on the duration of delivery plus service. Features were selected, missing entries were handled, and the performance of various models was analyzed. It was discovered that RMSE was improved by having a large number of pertinent features[4].

2.3.5 Food Delivery Time Prediction in Indian Cities Using ML Models.

The primary goal of this study by A. Garg, M. Ayaan, S. Parekh, and V. Udandarao is to improve the accuracy of food delivery time estimates. This is crucial for customer satisfaction, operational effectiveness, and financial gain. Existing studies mostly use old, static data and ignore real-time factors like traffic, weather, local events, and location data, especially important in busy Indian cities. The study examines several ML algorithms, including LR, DT, Bagging, Random Forest, XGBoost, and LightGBM, and incorporates these real-time parameters into prediction models to address this. After careful data cleaning and feature selection, LightGBM performed the best with an R^2 score of 0.76 and a Mean Squared Error (MSE) of 20.59. This shows that using dynamic data with advanced models gives more accurate results [5].

2.3.6 A Framework for Predicting Service Delivery Efforts Using IT Infrastructure-to-Incident Correlation.

Joel W. Branch, Y. Diao, and L. Shwartz propose a framework to predict IT service delivery workforce requirements based on changes in infrastructure and system loads. Using support vector regression, the model forecasts ticket volumes and correlates them with server characteristics to optimize staffing. Evaluated on real service delivery data, the study shows accurate workload predictions are feasible, helping enterprises plan staffing efficiently and manage costs under varying IT conditions [6].

2.3.7 Performance Evaluation of ML Techniques on Resolution Time Prediction in Helpdesk Support System.

The goal of this study is to better allocate customer service resources by forecasting incident resolution times. The first step in the process is data preprocessing, which includes addressing irregularities, outliers, and missing values. In this paper, T.-E. Tai, Su-C. Haw, W.-E. Kong and K.-W. Ng used several ML algorithms such as LR, DT, RF, SVM Regression, and XGBoost. OLS analysis using a 95% confidence interval, a 70:30 train-test split, and full dataset training are all used in the experiments. MAE, MSE, and MAPE are the evaluation metrics that are employed. The best-performing regression models, according to the results, are DT and RF [7].

2.3.8 A ML-based help desk system for IT service management

F. Al-Hawari and H. Barham represent an in-house help desk system for GJU that supports ticket reporting, service requests, prioritization, classification, assignment, status updates, collaboration through comments, and report generation for management decisions. The system also manages services, roles, and email notifications. An ML model was developed for automatic ticket classification using preprocessing, stemming, TF-IDF, and an SVM algorithm. When ticket comments were added, accuracy increased from 53.8% to 81.4%. The system acts as a Single Point of Contact (SPOC) between IT staff and users, enabling process optimization and KPI measurement to evaluate IT staff performance and efficiency [8].

2.3.9 On the use of ML to predict the time and resources consumed by applications.

This paper explores predicting resource usage (execution time, memory, disk) in datacenters, clouds, and grids with diverse hardware. It tests several ML methods using both system and application-specific attributes. A new method, PQR2, extends the Predicting Query Runtime (PQR) algorithm by choosing the best regression model for each case. On bioinformatics applications BLAST and RAxML, PQR2 achieved the best accuracy, while SVM and k-NN were also effective for non-linear scenarios. Including detailed attributes improved overall prediction performance[9].

2.3.10 Automated IT Service Desk Systems Using ML Techniques.

S. P. Paramesh and K. S. Shreedhara propose an automated service desk ticket classifier to improve IT helpdesk systems. Manual misrouting of tickets causes delays,

wasted resources, and lower customer satisfaction. The proposed system uses historical ticket data to automatically classify new tickets based on their descriptions. Four classification algorithms, including Naive Bayes, LR, KNN, and SVM, were tested, with SVM performing best across all samples. The system handles unstructured, noisy data through cleaning techniques. Benefits include faster ticket resolution, better resource utilization, improved end-user experience, increased customer satisfaction, and overall business growth[10].

2.4 Prediction of Technician's Name or SolvedBy and Similar Classification Tasks

In this part, the previous studies will be described, and it will also be discussed how the problem solver is predicted by the author through classification. Limited research directly examines "who resolved a problem or support ticket," but most studies are similar tasks.

2.4.1 A systematic literature review of ML methods applied to predictive maintenance.

T. P. Carvalho, F. A. A. M. N. Soares, R.Vita, R. da P. Francisco, J. P. Basto, and S. G. S. Alcala present a systematic literature review on ML methods for Predictive Maintenance (PdM). With the rise of sensing technologies, industries generate vast amounts of production data that, when analyzed, can detect equipment faults early, reduce downtime, and improve efficiency. The paper reviews ML techniques, their performance, challenges, and opportunities, offering a strong foundation for future PdM research and industrial applications [11].

2.4.2 TaDaa: Real-Time Ticket Assignment Deep Learning Auto Advisor for Customer Support, Help Desk, and Issue Ticketing Systems.

This paper presents TaDaa (Ticket Assignment Deep Learning Auto Advisor), a system that applies Transformer models and ML to optimize ticket management in customer support and help desk systems. TaDaa performs three main tasks: assigning tickets to the correct group, recommending the best resolver, and retrieving relevant past tickets. Using a dataset with 3,000 groups and 10,000 resolvers, it achieved 95.2% accuracy for group assignment and 79.0% accuracy for resolver prediction. TaDaa addresses the challenge of tickets solved by multiple resolvers, enhancing accuracy,

reducing resolution time, and offering a flexible, generalizable solution [12].

2.4.3 A Survey on Hardware Failure Prediction of Servers Using ML and Deep Learning.

In their paper, N. Georgoulopoulos and his co-authors survey different methods for predicting hardware failures in servers, focusing on Hard Disk Drives (HDD), RAM, and CPUs. It is argued that traditional monitoring tools are not proactive enough. The paper explains how hardware errors can be effectively predicted by using Machine Learning (ML) and Deep Learning (DL) techniques before they occur. The authors find that most studies concentrate on HDD and RAM failures, while CPU failure prediction is less researched. The conclusion is that more research is needed, especially with the use of DL models for CPU faults [13].

2.4.4 ML-Based Approach for Hardware Faults Prediction.

Kasem Khalil, Omar Eldash, Ashok Kumar, and Magdy Bayoumi propose an ML-based method for early hardware fault prediction. Their approach combines FFT for frequency signatures, PCA for dimensionality reduction, and CNN for fault classification. The technique predicts aging, short-circuit, and open-circuit failures with 99% accuracy when tested on comparator and amplifier circuits employing 45 nm technology and factors including voltage, current, temperature, noise, and delay. Implemented in TensorFlow and validated on an FPGA device, it consumes 1.08 W, proving highly accurate and efficient [14].

2.4.5 Efficient support ticket resolution using Knowledge Graphs.

This article reviews 160,000 customer cases, showing that 90% of support time is spent on 10% of complex tickets requiring swarms of engineers or developer support. It proposes a Learning-to-Rank (LTR) task enhanced with Knowledge Graph embeddings from incident data, expertise ratings, knowledge base content, and swarming history. A novel system using Graph Neural Networks (GNNs) and VLLM embeddings ranks engineers based on experience and domain knowledge. Results show improved recommendations and efficiency compared to traditional methods like TF-IDF [15].

2.4.6 Random Forest-Based Machine Failure Prediction: A Performance Comparison.

This study evaluates twelve ML models for predictive maintenance using two manufacturing datasets (large and imbalanced). Performance was assessed using accuracy, precision, recall, F1-score, and ROC AUC. The accuracy is 99.5% on Dataset A and 88.7% on Dataset B. RF performed better than the others and showed resilience to noisy and unbalanced data. Transformers performed well on large data but struggled with smaller sets. LDA showed potential for simpler tasks, while naive Bayes, SVM, and logistic regression underperformed. Hybrid approaches and broader real-world validation are recommended [16].

2.4.7 A systematic literature review of ML applied to predictive maintenance.

This work presents a broad overview of using ML for predictive maintenance (PdM) of industrial equipment. With the rise of Industry 4.0 and advanced sensing technologies, PdM has become increasingly feasible, helping reduce costs, avoid failures, and improve efficiency. The review highlights that PdM studies often target specific equipment, making comparisons difficult. Common ML methods include SVM, Random Forest, ANN, deep learning, and k-means. Future research should focus on improving sensing, comparing multiple ML algorithms, applying ensemble methods, and developing new datasets to enhance robustness, accuracy, and practical adoption of PdM strategies [17].

2.5 Research Gap

Limited Research on Solver Prediction: Most studies focus on predicting service delivery time or resolution duration, but few directly predict who will solve a ticket or problem. Accurate assignment of technicians remains underexplored, especially in large-scale IT service environments.

Dual target prediction: Predicting both who resolves the issue ("solvedby") and delivery time / delivered days together is less common. Many studies do either classification or regression, but not both together.

Limited features: The majority of research has used few and local attributes. For example, user information, location within a city, product model, problem type, delays caused by the supply of parts, etc. These local features are often not present

(or not used) in many public datasets.

Domain specificity (IT hardware service): Many jobs are generic help desk, customer service, or supply chain/manufacturing. Very little research has focused on hardware servicing, model/product-specific delays, parts supply, etc.

Limited Use of Hybrid or Ensemble ML Approaches: Single ML models such as Random Forest, XGBoost, or CNN have dominated current research. Hybrid or ensemble methods, which can improve robustness and accuracy for unbalanced or complex datasets, have not been widely applied.

2.6 Summary

This chapter presented different studies on IT support services using ML. Researchers have applied many models, such as RF, DT, LR, KNN, SVM, CNN, Neural Networks, and hybrid approaches. This chapter reviews how ML and predictive analytics improve IT hardware service management (ITHSM). Researchers have applied many ML models, like Random Forest, XGBoost, LightGBM, SVM, and CNN, to predict service outcomes ("solved by" technicians), incident resolution times, and delivery durations. Automated ticket classification, deep learning, and transformer-based systems improve accuracy, reduce manual effort, and enhance customer satisfaction. Research also covers hardware fault prediction in servers, mainly HDD and RAM, with CPU faults being less explored. Despite advances, gaps remain in real-time data integration, solver prediction, CPU fault detection, and hybrid model validation, highlighting opportunities for future work in efficient IT hardware servicing. Overall, the literature shows that ML has strong potential for IT support in faster and more reliable decision-making. This motivates the work, where five ML classifiers are applied to achieve highly accurate IT hardware support.

CHAPTER III

System Model and Dataset Analysis

3.1 Overview of the Chapter

In this chapter, the proposed system and its capabilities are going to be discussed. The following section explains some of the feature selection techniques that are designed to improve model accuracy and simplicity. It describes the dataset characteristics in terms of RgSerialNO, UI, ProductModel, ReceiveDate, Solution, Problem, and how the data are pre-processed and partitioned. This chapter establishes the foundation for implementing a robust prediction system with ML techniques.

3.2 Proposed System Model

The workflow diagram is shown in Figure 3.7, for this ML project, starting with data collection, followed by the extraction of significant variables, and finally feature selection or preprocessing to prepare the dataset. The processed data is subsequently divided into training and testing subsets. The model's efficacy is evaluated by measures after the use of the algorithm. The process ends successfully if the model satisfies the predetermined requirements; if not, the workflow goes back to earlier phases, like feature selection or data preparation, to improve the technique and improve outcomes. This iterative process guarantees the creation of a reliable and accurate predictive system.

3.2.1 Dataset

The dataset includes IT hardware servicing records such as RgSerialNO, UI, ProductModel, ReceiveDate, Solution, Problem, etc. The dataset serves as a basis for the system. Since the quality of the data directly affects how well ML models perform in
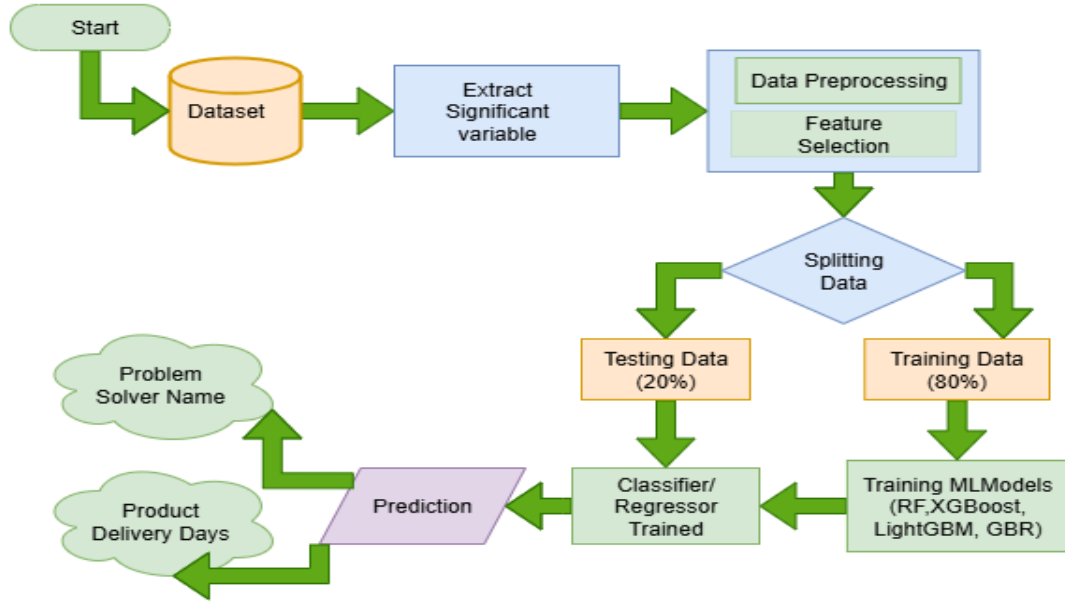
Figure 3.1: System design, data flow, and prediction steps.

correctly predicting SolvedBy and Delivery Days, it is crucial to gather a significant amount of reliable and precise data.

3.2.2 Extract Significant Variables

At this stage, the system determines the most significant variables in the data affecting SolvedBy and Delivery Days. For instance, ReceiveDate, Solution, Problem, and Speciality are important parameters. Here, the system eliminates the features that are not essential for the precision and effectiveness of the solver name and delivery days prediction, as not all data values are equally important for prediction.

3.2.3 Feature Selection

Feature selection is performed to select the most optimal features from the dataset, and discard the irrelevant or redundant ones. This makes it easier for the system, as it will get rid of some noise and not confuse the system. Predictions are faster and better when only the relevant features are selected. For ML models, it is also helpful to prevent overfitting and enhance the generalization for the coming unseen data.

3.2.4   Data Pre-processing

Data pre-processing is the most important part of a predictive model or system. Negative values, redundant data, and errors are commonly found in this dataset. Pre-processing addresses these issues by cleaning and pre-processing data. It consists of normalizing, handling missing values, and everything looks as the data is expected to be. In this part, all the categorical values, such as UI, Location, ProductModel, SolvedBy, etc., were encoded using LabelEncoder. And also converted the receive and delivery date into numeric values. Simple but effective to maintain a tidy and coherent dataset! If not pre-processed, models may predict incorrectly, so this is the task of an HD prediction system with ML.

3.2.5   Splitting Data

The dataset is categorized into two segments: testing and training. The ML model learns using training data, and its performance is evaluated using testing data. Typically, the division is carried out in such a way that, for example, 80% or 70% of the ML data set is utilised for training, while the remaining 20% or 30% is used for testing.

3.2.6   Training Data

The training data comprises the dataset utilized to develop the system. The algorithms used are RF, LGBM, and XGBoost for classification and RF and GB for regression. The model learns the non-linear patterns and dependencies that exist between the input features and the output categories. In accordance with the filing, the training data allows the system to comprehend the relationship between the solver name or delivery days and a variety of IT servicing parameters. Here, 80% of the data was used for training.

3.2.7   Testing Data

This portion of the data was used to evaluate how well it was trained. The model's ability to generalize to new data, which was absent during training, is tested. The test makes sure the model is learning to read, not just memorize. A positive testing result validates the strong prediction capability of the system. The remaining 20% of data is used for testing.

3.2.8 Classifier Trained

Once the system has been trained using the training data, it becomes a trained classifier. Now, this classifier can predict the solver name and delivery days. It utilizes what it learned during training to evaluate new data from the servicing dataset. A well-trained classifier is particularly critical in that it guarantees reliable predictions for hardware servicing decision-making.

3.2.9 Results

The last stage of the system outputs results. Using the hardware servicing data, the system will predict the problem solver's name and delivery time.

3.3 Choice of Attributes

Choosing the right features is vital to the ML workflow, as unnecessary attributes can negatively impact the classifier's performance. Furthermore, the time taken to execute the model has been decreased. The performance of each model created using ML techniques is evaluated for every iteration based on 11 characteristics, and performance metrics are documented.

3.4 Overview of the Dataset

In this section, the relationship of original attributes with encoded features used in the dataset is described.

RgSerialNO: It means the serial number of ITHS dataset.

UI: User Information.

Original Value	Encoded Value
Male	1
Female	0

Location: This denotes the particular location, branch, or region where the service was required. It assists in monitoring issues according to their regional or departmental source.

Original Value	Encoded Value
ACCL	0
ACML	1
AFL	2
ALP	3
Arkay	4
Azmeri	5
Badda Parking	6
CHO	7
Cortz	8
Gulshan-29	9
KM	10
NKK	11
Nafa	12
Safa	13

ProductModel: This is an identification for each product.

Original Value	Encoded Value	Original Value	Encoded Value
Acer CPU	0	Epson Printer	11
Acer Laptop	1	HP CPU	12
Acer Monitor	2	HP Laptop	13
Asus CPU	3	HP Printer	14
Asus Laptop	4	Lenovo CPU	15
Asus Monitor	5	Lenovo Laptop	16
Brother Printer	6	Lenovo Monitor	17
Canon Printer	7	Printer	18
Dell CPU	8	Projector	19
Dell Laptop	9	Scanner	20
Dell Monitor	10	UPS	21

Problem: A problem refers to a specific type of malfunction or fault that was reported.

Original Value	Encoded Value
Auto-Restart	0
Battery	1
Broken Issue	2
Color Problem	3
Display	4
HDD	5
Keyboard/Touchpad	6
Lamp/Temp	7
Motherboard	8
New Laptop/CPU	9
Not Work	10
OS Problem	11
Print Problem	12
SSD	13
Slow/Hang Problem	14

Solution: This code denotes any specific operation or repair executed to solve the reported issue.

Original Value	Encoded Value
Ready	0
Replace	1
Servicing Issue	2
Warranty Issue	3

Requisition: When a product needs a part, we take a requisition from the user.

Original Value	Encoded Value
No	0
Yes	1

SolvedBy: This column identifies the technicians or engineers who were responsible for executing the repair and effectively resolving the service tokens.

Original Value	Encoded Value
Moshiur	0
Sarju	1
Shajal	2
Shuvo	3

Speciality: This ensures that the appropriate specialist is assigned to the task.

Original Value	Encoded Value
CPU	0
Laptop	1
Monitor	2
Printer/UPS	3

3.5 Data Description

As mentioned earlier, the dataset collected in this study was from our office IT hardware service management. The data needed to be pre-processed because it also contained null or blank values. The dataset has 11 parameters. These features include RegSerialNO, UI, Location, ProductModel, ReceivedDate, Problem, Requisition, Solution, Speciality, SolvedBy, DeliveredDate, and two target variables indicating the problem solver name and delivery days. After data preprocessing, LevelEncoder was utilized to convert categorical values into numeric values. This well-organized dataset allows one to train and test binary classification models easily.

RgSerialNO	UI	Location	ProductModel	ReceiveDate	Problem	Requisition	Solution	SolvedBy	Speciality	DeliveryDate
1228	Male	CHO	HP CPU	01-07-2025	New Laptop/CPU	Yes	Ready	Sarju	CPU	01-07-2025
1229	Male	CHO	HP Laptop	01-07-2025	New Laptop/CPU	Yes	Ready	Shajal	Monitor	01-07-2025
1230	Male	CHO	Acer CPU	01-07-2025	OS Problem	No	Ready	Shajal	Monitor	Out
1231	Male	Arkay	HP Printer	01-07-2025	Not Work	No	Ready	Moshiur	Printer/UPS	10-07-2025
1232	Male	Arkay	HP CPU	01-07-2025	HDD	Yes	Ready	Sarju	CPU	13-07-2025

Table 3.1: Data Description

3.6 Visualisation Analysis of the Dataset

In this section, the visualization chart of IT hardware servicing systems will be explained. These figures illustrate the correlation of work among the four technicians (like Moshiur, Sarju, Shajal, and Shuvo) by analyzing the IT support dataset. The target feature is represented as SolvedBy; it shows that most of the problems are resolved by Moshiur, Shajal, and Sarju. In this dataset visualization chart, the most important correlation is SolvedBy between Speciality and ProductModel. All of these charts are shown below:
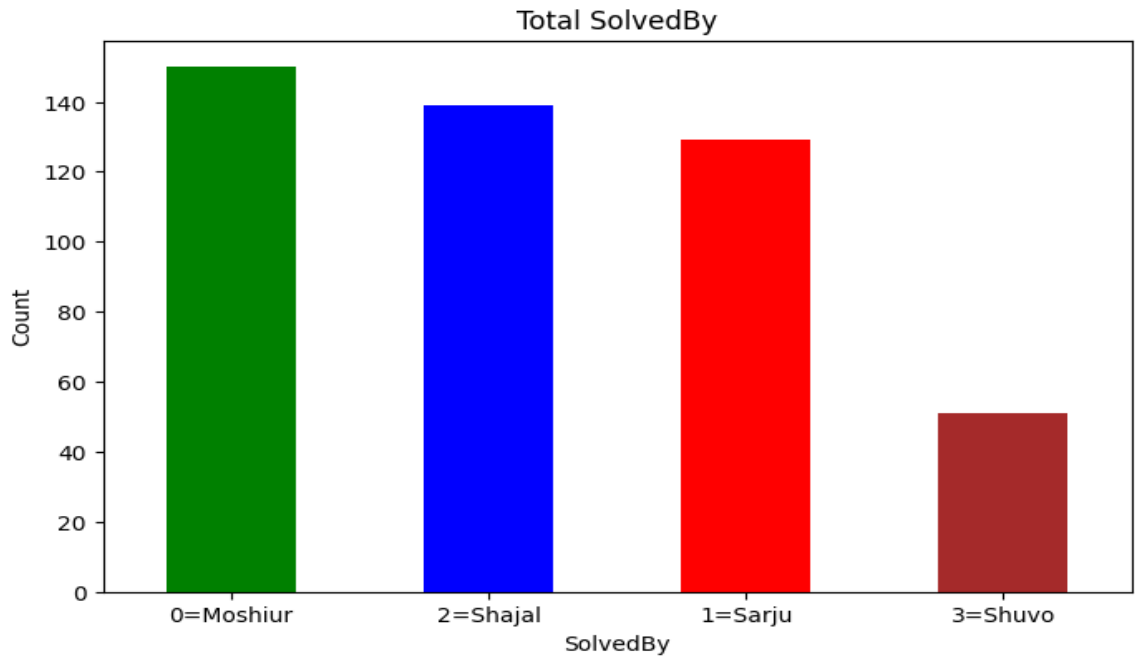
3.6.1 Frequency level of SolveBY:

Figure 3.2: Frequency and workload of problem solvers.

Figure 3.2, shows the total number of problems solved by four specialists. Moshiur has resolved the highest number of difficulties. Shajal and Sarju are solving near, but Shuvo fixed the fewest problems compared to others.

3.6.2 Correlation of Location Vs SolvedBy:

Figure 3.3 indicates the frequency chart between Location and SolvedBy. 'Location 7' is a significant point for technical problems in this visualization. Shajal resolved the largest number of difficulties at this location, closely followed by Moshiur and Sarju.
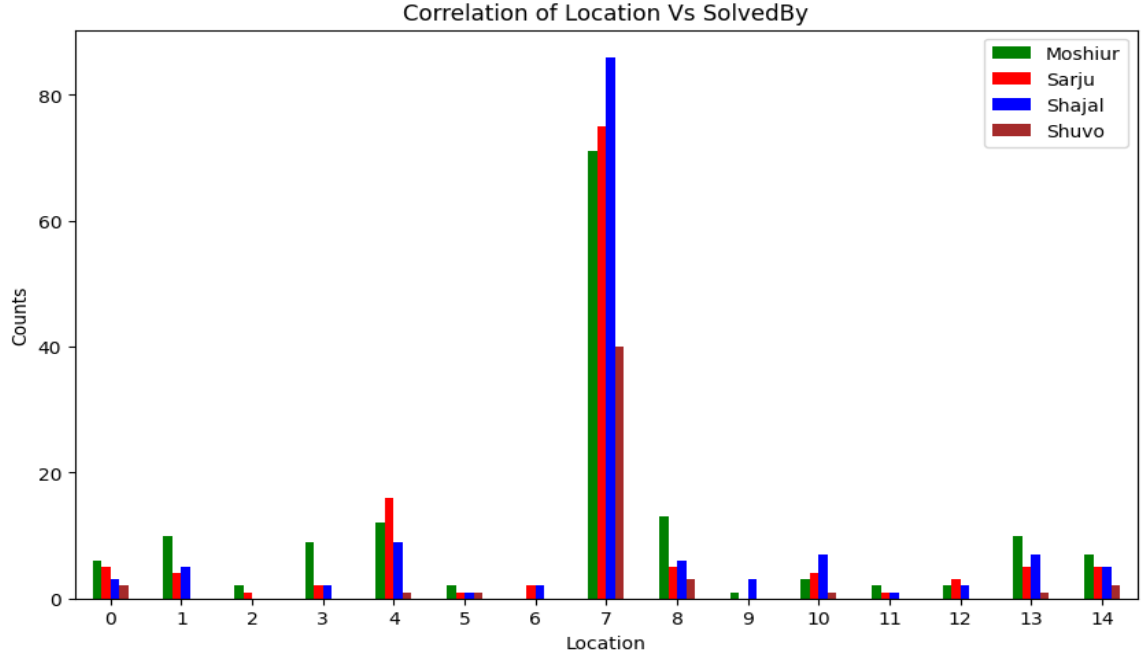
Figure 3.3: Mapping technicians to service locations.

3.6.3 Correlation of ProductModel Vs SolvedBy:

Figure 3.4 illustrates the correlation between ProductModel and SolvedBy. The ProductModel column has 21 attributes. Sarju is largely responsible for 'Product-Model 8'; Moshiur serves as the principal contact for models 7, 11, 14, and 21, while Shajal manages multiple cases for 'ProductModel 13'.
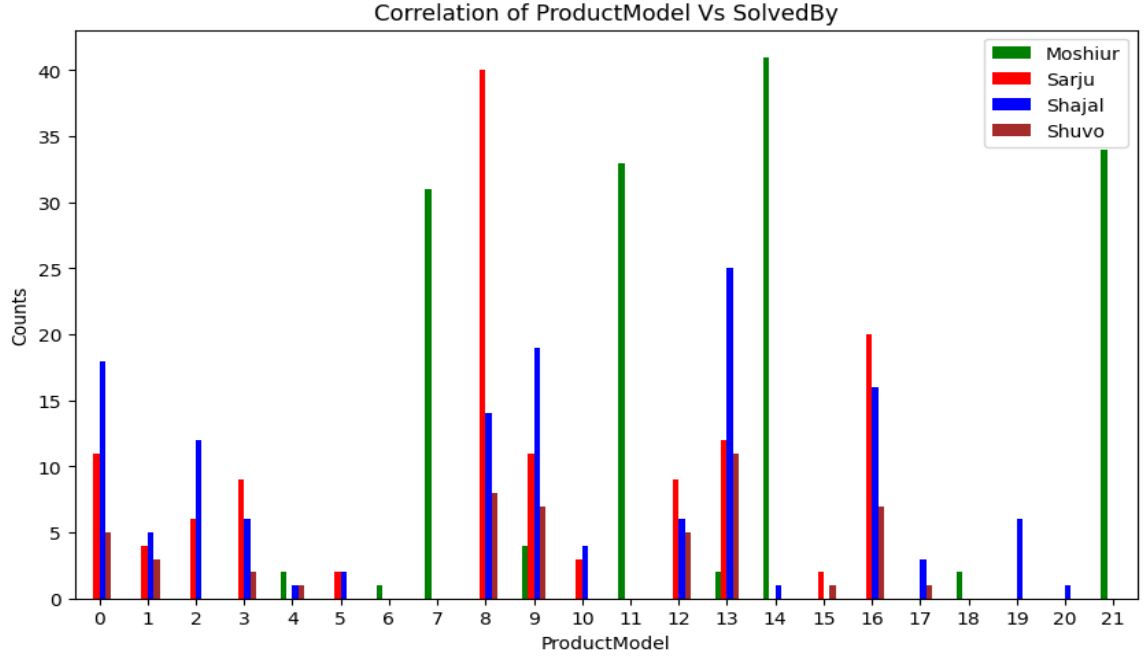
Figure 3.4: Product models and assigned solvers.

3.6.4 Correlation of problem Vs SolvedBy:

Figure 3.5 shows the relation between Problem and SolvedBy. The Problem column has 14 features. This graph clearly shows that technicians are experts at resolving specific types of issues. Moshiur has solved a significant number of "Problem 12" cases, making him a specialist in the case. Whereas Shajal handles the issues of number 10.
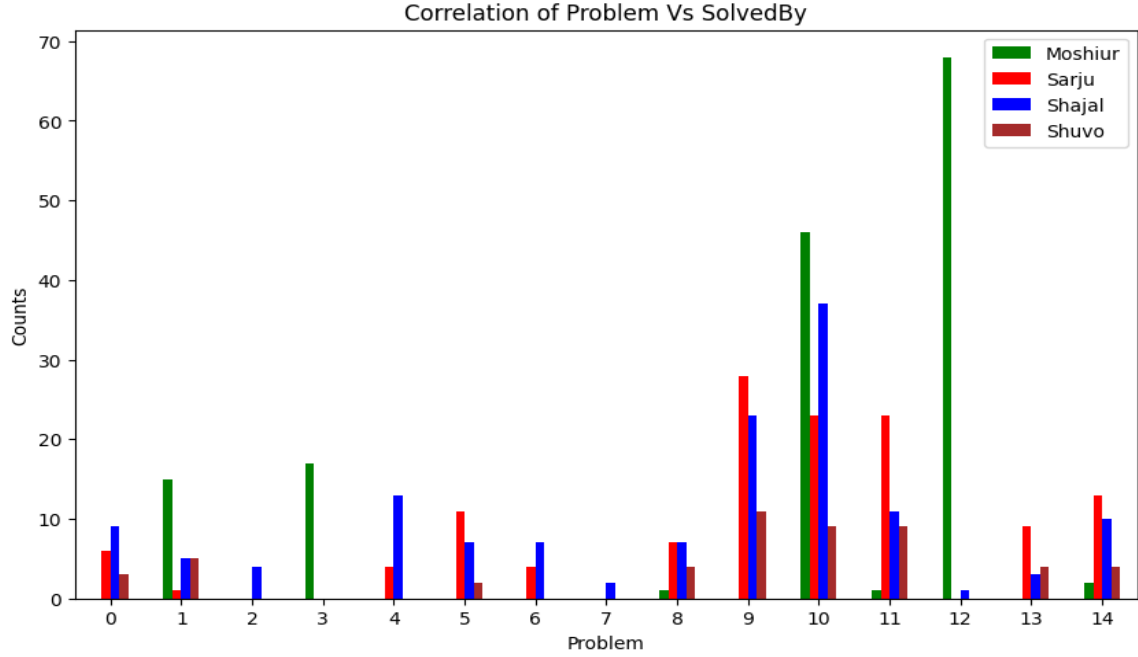
Figure 3.5: Hardware issues are handled by technicians.

3.6.5 Correlation of Speciality Vs SolvedBy:

Figure 3.6 displays the relation between Speciality and SolvedBy. The Specialty column has 4 characteristics. This graphic illustrates a flexible division of work based on specialization. Every person possesses an individual role: Sarju controls 'Speciality 0', Shuvo handles 'Speciality 1', Shajal is responsible for 'Speciality 2', and Moshiur manages 'Speciality 3'. There is no intersection between their designated responsibilities.
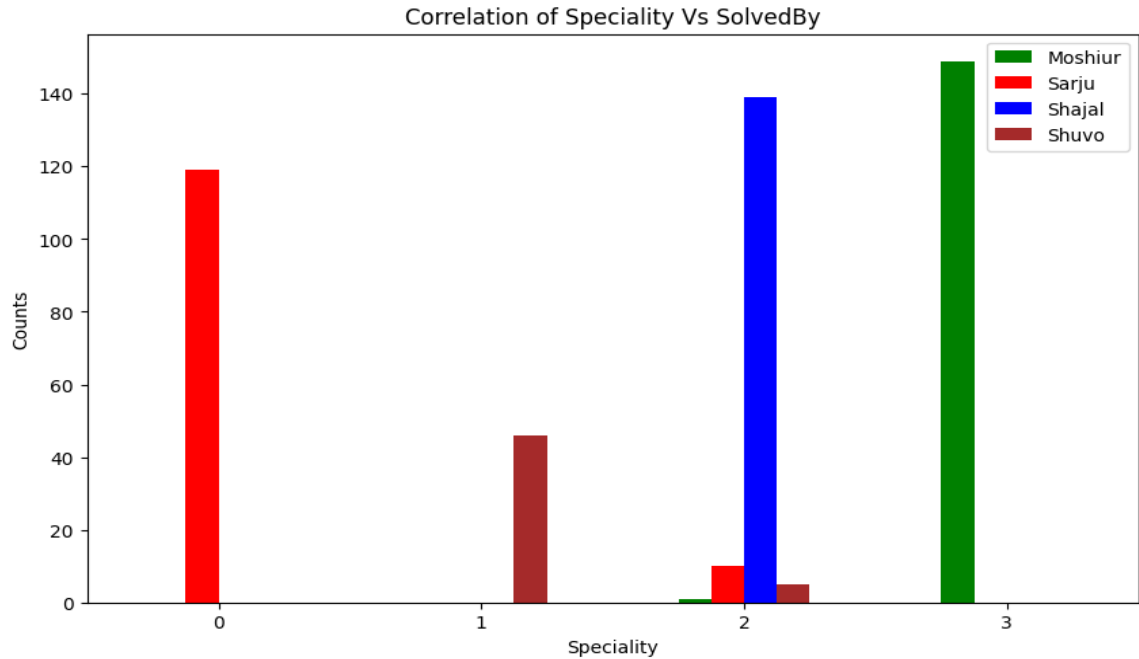
Figure 3.6: Shows the specific technician's performance.

3.7 Correlation Matrix Analysis

Figure 3.7 shows the correlation matrix. It displays the interconnections among several attributes in the dataset. Positive relationships are represented by red squares (1.00), which means the variables move in the same direction. Blue squares indicate a negative relation (-1.00). The white squares indicate no relationship. This matrix displays a strong relation between 'Received Month' and 'Delivery Month'.

Figure 3.7: Feature relationships in the dataset.

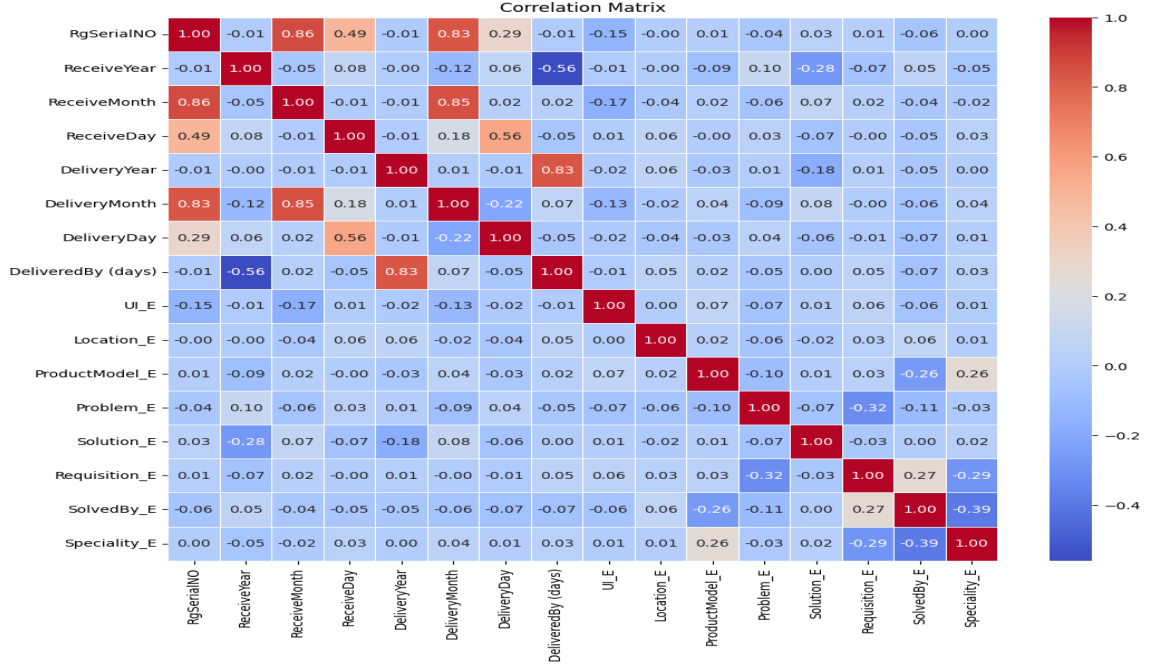## 3.8 Summary

This chapter describes the target prediction model proposed and the dataset applied with visualization analysis.

CHAPTER IV

System Tools and Algorithms

4.1 Overview of the Chapter

This chapter introduces the proposed system tools and ML algorithms used in this study. The focus is on improving model accuracy and reducing complexity. In this section, the system configuration, including hardware and software requirements, as well as Python libraries, will be discussed. It also discusses the ML algorithms used for classification and regression tasks. These tools and algorithms form the foundation for effective prediction of 'SolvedBy' and 'DeliveredDays'.

4.2 Configuration of the System

Computer configuration is most important for any ML project. In this project, a desktop computer was used. The configurations of this computer are given below:

Processor: Intel(R) Core (TM) i5-3470 CPU (3.20 GHz).

Operating System: Windows 11, 64-bit.

Installed RAM: 8GB.

Storage: 256 GB SSD and 1TB HDD

4.3 Software Tools Requirement for Implementation

The proposed illuminating algorithm has been implemented in Google Colab using Python programming language and its libraries. The following libraries are utilized for data manipulation, visualization, and encoding categorical values in my proposed project system: Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, LabelEncoder, joblib, and gradio.

4.3.1 Python 3.8+

Python is a popular programming language known for its clean and easy-to-read syntax. It is commonly used for web development, data analysis, machine learning, automation, and other applications. Its easy syntax and rich libraries such as Scikit-learn, NumPy, and Pandas make it well-suited for applications like working with data, performing complex mathematical operations, software processes, and also better system implementation.

4.3.2 Numpy Library

NumPy is a fundamental Python library, used to make programming with multi-dimensional arrays more expressive and easier. It's commonly used for managing enormous data sets, also known as "Big Data." NumPy provides a collection of programming functions for working with arrays, allowing us to operate on such arrays. NumPy was used for the manipulation of array data in this study, providing convenient and fast calculations about mathematical functions, as well as efficient handling of advanced computing operations.

4.3.3 Matplotlib Library

Matplotlib is written in Python, and it is an open-source plotting library that provides data visualization tools. Matplotlib has many built-in plotting functions similar to MATLAB's, using a similar method called pyplot. For example, every pyplot capacity can make a figure, a plotting area in a figure, and lines on a plotting zone; subscribes to markers enhancing our plots. Our results are graphically represented by means of output with the Matplotlib library.

4.3.4 Pandas

Pandas is a widely used and powerful Python library for data manipulation that provides user-friendly data structures (such as DataFrame and Series) for efficiently manipulating structured data. Key-value is important for advanced data science and analytics processes as it packages functions to import, clean, transform, and summarize data.

4.3.5 Scikit-learn Library

Scikit-learn is a popular free ML library that has functions for different clas-

sification, regression, and clustering algorithms, such as SVM, KMeans, and RF. It provides modules to extract features, perform feature selection, cross-validation, clustering, and dimensionality reduction. For standard scaling, it uses StandardScaler(), and for replacing missing entries, it uses SimpleImputer(). It also includes DTW and PCA. Train-test-split is a flexible function to split data into test or training datasets, which are used in model construction and feature selection/extraction in ML applications.

4.3.6 Seaborn

It's a Matplotlib-based Python library and is designed to be a statistical data visualization tool that provides high-level plotting functions for an intuitive way to get insight into our data. Its special features, focused on the visualization of distribution, relationship, or correlation, and categorical data with less code, make it even easier to visualize and comprehend the nature of data.

4.3.7 Label Encoder

Label encoder is a library in Python for converting categorical data to numerical values, which are used in ML models.

4.3.8 Joblib

Joblib is a Python package to save Python objects, such as data and ML models, for parallel computing. It is faster than pickle and more effective when working with big NumPy arrays. Joblib is a popular library used in ML to save trained models, load them later, and speed up computation tasks using multiple CPU cores.

4.3.9 Gradio

Gradio is an open-source Python library that easily builds interactive web/user interfaces (WIs/UIs) for ML models or any Python functions. Various types of features, such as text, images, audio, and other data, can be inputted, allowing model predictions to be provided instantly.

4.4 Machine Learning (ML) Models

ML is a subset of artificial intelligence that employs statistical and mathematical algorithms to derive insights from data and generate predictions or judgments

autonomously, without explicit programming. Its performance enhances with the processing of additional data. Deep learning, a subset of machine learning, uses neural networks to surpass numerous traditional techniques. ML is widely used in areas like language processing, vision, speech recognition, and healthcare. Its use in solving business problems is known as predictive analytics.

4.5 ML Algorithms for Classification Task (Predicting 'SolvedBy')

In this phase, three supervised ML classifiers, such as RF, XGBoost, and LGBM, will be outlined for analysis. The working principle of these algorithms will now be described.
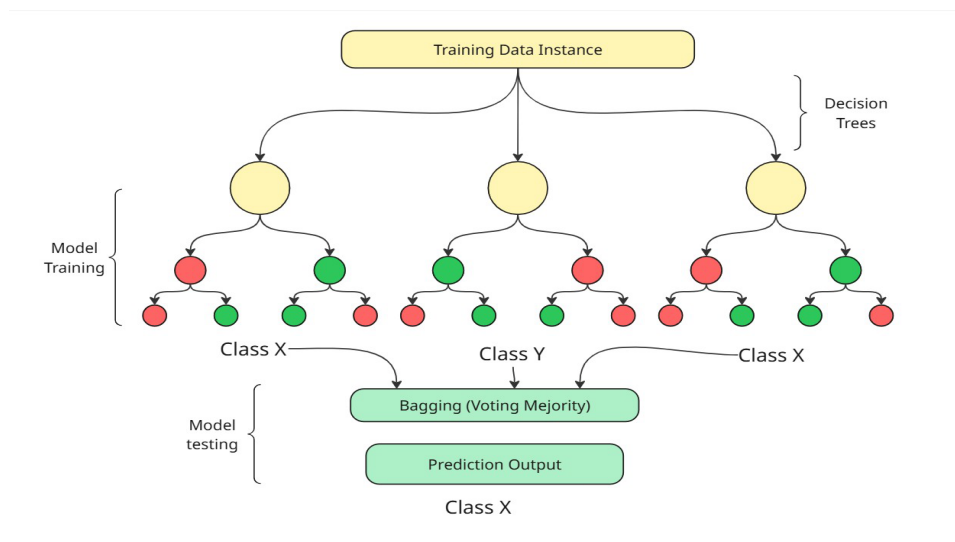
4.5.1 RF Classification

Figure 4.1: Illustrates the working procedure of the Random Forest classifier.

RF classifiers mean random forest classifiers. It is an ensemble ML algorithm that combines many decision trees to classify data. For each tree in the forest, a prediction is made, and the final class is determined through majority voting across all trees. It does this by training numerous decision trees on various randomly partitioned subsets of the data and features, thus lessening overfitting and enhancing accuracy. In situations involving classification, it is often the favored option.

4.5.2   XGBoost Classification

XGBoost is a robust ML algorithm founded on gradient boosting principles. It constructs an ensemble of decision trees, with each subsequent tree rectifying the flaws of its predecessors. The utilization of numerous shallow trees mitigates overfitting. XGBoost is recognized for its rapidity, adaptability, and superior predictive accuracy.
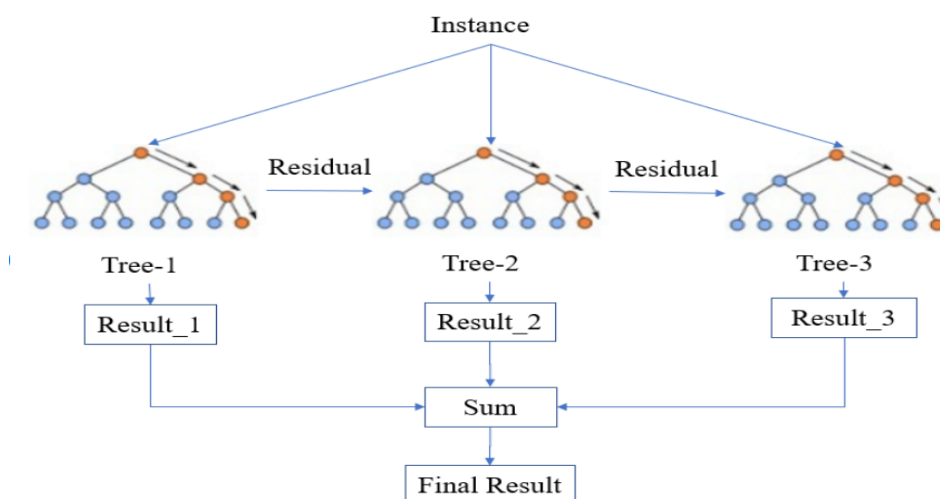

Figure 4.2: Showing how data is processed and predictions are generated efficiently.

XGBoost operates through multiple stages to construct a robust predictive model. It commences with a fundamental model, such as forecasting the mean result. Subsequently, it computes the discrepancies (residuals) between the predictions and the actual values. New decision trees are incrementally incorporated to forecast these errors and enhance precision. Gradient descent is employed to minimize the total error. A learning rate regulates the influence of each new tree to avert overfitting. Regularization is employed to manage model complexity. This process continues until the model attains a predetermined number of trees or the reduction in error ceases. The ultimate model integrates all tree projections.

4.5.3 LightGBM

LightGBM Classifier (LGBM) is an ML method originated by Microsoft and is of the GBDT type. It is also fast, efficient, and scalable, which benefits large data. LGBM can add many small decision trees step by step to enhance accuracy and remain memory-friendly. It has been commonly used for binary classification (e.g., disease detection, spam filtering, etc.). In LightGBM, decision trees are built one after another, with each new tree aiming to fix the mistakes made by the previous ones. It starts with a simple tree and keeps improving the model by focusing on the errors. Unlike traditional methods that grow trees step by step, LightGBM grows them leaf-wise, choosing splits that offer the highest information gain. This helps it capture complex patterns more effectively. It also uses a histogram-based approach, grouping continuous values into bins to save memory and speed up training. With many small trees working together, LightGBM delivers fast, efficient, and accurate predictions, especially on large datasets.
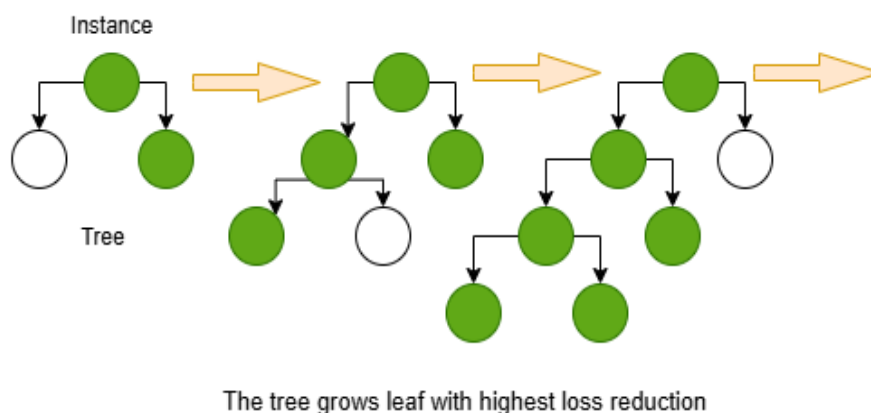

Figure 4.3: Illustrating how data predictions are efficiently produced.

4.6 ML Algorithms for Regressor Task (Predicting 'DeliveredDays')

In this part, two supervised ML regressor algorithms, such as RF and GB, will be discussed. The working principle of these algorithms will now be described.

4.7 RF Regressor

Random Forest Regression is an ML algorithm used to predict continuous values, which builds many DTs using random subsets of data (Bootstrap sampling) and features, ensuring diversity among trees. The prediction of each tree is made, and the overall result is an average of all tree predictions. This process is referred to as aggregation. This approach reduces the high variance and overfitting common in single decision trees, leading to more accurate, stable, and generalizable predictions. Random Forest is especially effective for complex datasets, improving reliability and overall performance in regression tasks. Figure 4.5 shows the workflow diagram.
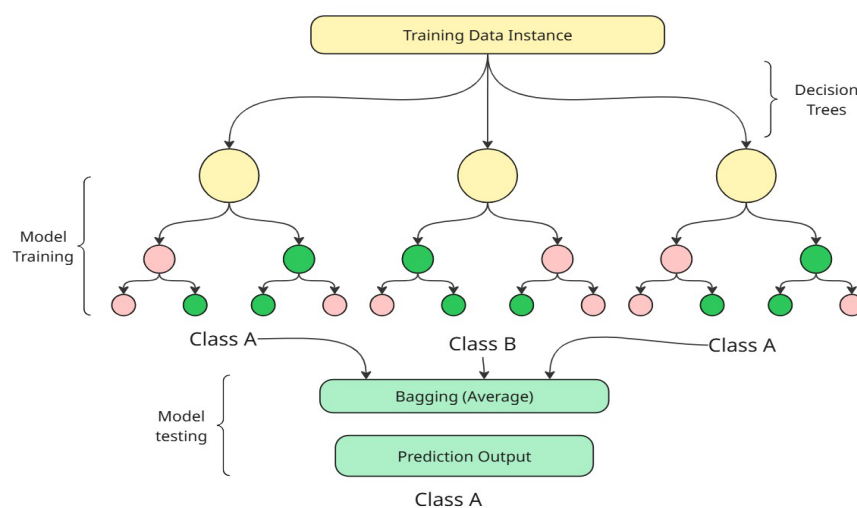

Figure 4.4: Showing how multiple trees combine for predictions.

Random Forest Regressor is a powerful, accurate, and easy-to-use model. It handles small or large datasets, reduces overfitting, works with messy data, requires little tuning, shows feature importance, and provides stable predictions for real-world regression tasks.

4.8 GB Regressor

Gradient Boosting Regressor predicts continuous values by building decision trees sequentially, each correcting the previous error. It weights better-performing trees more, learns complex patterns effectively, and delivers accurate predictions, making it widely used despite being slower than some other methods.

The Gradient Boosting Regressor predicts values by building multiple decision trees sequentially. It starts with a simple tree, then calculates errors (residuals)

between predictions and actual values. Each new tree is trained to correct these errors, gradually improving accuracy. This step-by-step process, called boosting, combines all trees for final predictions. Gradient descent guides each tree to reduce errors effectively. To prevent overfitting, the model uses small trees, regularization, and a learning rate that controls each tree's influence. The process continues until the model reaches the set number of trees or achieves minimal error, resulting in an accurate and robust prediction model. Figure 4.6 shows the workflow diagram.
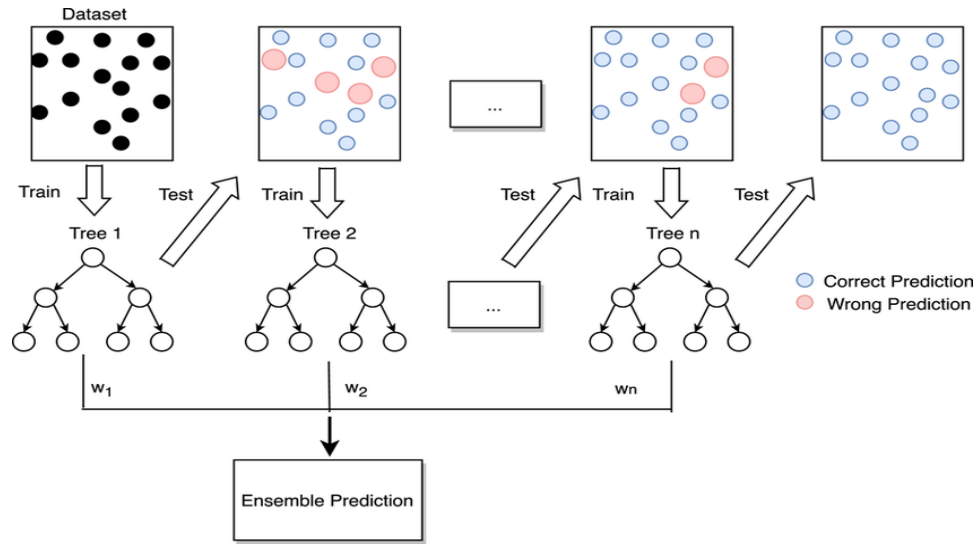

Figure 4.5: Illustrating sequential tree learning to improve prediction accuracy.

Gradient Boosting Regressor makes accurate predictions by combining many weak models sequentially. It handles complex, small, or large datasets, avoids overfitting with learning controls, requires little scaling, and identifies important features, making it ideal for real-world prediction tasks.

4.9 Summary

This chapter describes the system tools and ML algorithms used in this study to improve model accuracy and efficiency. It covers the hardware and software setup, including Python 3.8+ and libraries like NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn, LabelEncoder, joblib, and gradio libraries for data processing, visualization, and model implementation. The chapter explains supervised ML algorithms for classification ('SolvedBy') and regression ('DeliveredDays'), including Random Forest, XGBoost, LightGBM, and Gradient Boosting. It details their principles, workflows, and advantages, such as handling complex datasets, reducing overfitting, efficient training, feature importance, and providing accurate, stable predictions for real-world applications.

CHAPTER V

Result Analysis

5.1 Overview of the Chapter

This section centers on assessing the ML models, such as RF, XGBoost, and LightGBM classifiers, applied to predict the problem solver's name. Also, RF and GBR regressors were applied to predict product delivery days. Findings indicate that all of these classifier ML models achieve the highest accuracy, even attaining 96% during testing. In the regressor test, RF showed strong capability in forecasting resolution time (MSE: 13.09, RMSE: 3.62, and R-Square: 0.57). The chapter also provides visual comparisons of the models' performance and interface screenshots that illustrate how the system forecasts the target (Solver Name and Delivery Days).

5.2 Performance Metrix

If the model correctly identifies the target as a positive case, this is known as a True Positive (TP). If the model accurately recognizes the target as a negative case, this is known as a True Negative (TN). If a problem is wrongly labeled as a target due to a negative test result, this mistake is termed a False Negative (FN). Likewise, if a test result incorrectly identifies the target as a positive case, this error is called a False Positive (FP).

Accuracy: Accuracy is a performance metric that shows how many predictions are found correctly overall for the model. It is the ratio of all correct predictions to the total predictions 5.1. The formula is given below:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

Precision: Precision is the ratio of accurately predicted positive samples 5.2. The formula used for this calculation is,

$$Precision = \frac{TP}{TP + FP} \qquad (5.2)$$

Recall: Recall is a metric that quantifies the number of actual positive cases accurately identified by a model 5.3). The calculation formula is,

$$Recall = \frac{TP}{TP + FN} \qquad (5.3)$$

F1-score: The F1-score is a metric that uses the harmonic mean of precision and recall to combine them into a single value 5.4. The formula is,

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \qquad (5.4)$$

MSE: The average of the squares of the variations between the actual and predicted values is known as the MSE 5.5. The calculation formula is as follows:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \qquad (5.5)$$

RMSE: Root Mean Squared Error (RMSE) is a metric utilized to evaluate the accuracy of a regression model in predicting a continuous target variable 5.6. The average squared differences between actual and predicted values are represented by their square root.

$$\text{R}MSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \qquad (5.6)$$

R-squared (R^2): The coefficient of determination, or R-squared (R^2), is a regression metric that evaluates how effectively a model accounts for the variance in the target (dependent) variable 5.7.

$$\text{R}^2 = 1 - \frac{\sum_i (y_i - (\bar{y})_i)^2}{\sum_i (y_i - \bar{y})^2} \qquad (5.7)$$

5.3 Performance of the Predictive Framework

Among these three classifiers, all of them achieved the highest accuracy, attaining 96% on the testing dataset, and both regressors also achieved very strong time predictions.
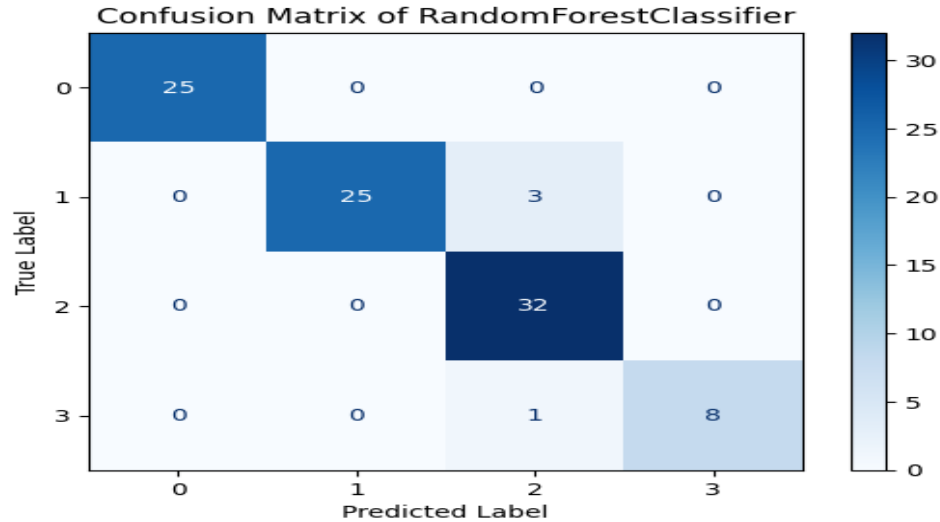
5.3.1 Confusion Matrix of RF Classifier

Figure 5.1: Showing correct and incorrect classifications for each class.

In Figure 5.1, the RF confusion matrix is shown in the first picture. The predicted labels are shown in the columns, and the actual (true) labels are shown in the rows. This matrix accurately classifies 25 cases for class 0, and there is no misclassification. For class 1, the model correctly predicted 25 cases, but misclassified 1 case as class 2. The models accurately predicted all 32 cases for class 2, with zero errors, and correctly predicted 8 cases for class 3; however, 1 case was incorrectly classified as class 2.

5.3.2 Classification report of RF

The RF model used in this classification report can predict one of four classes: 0, 1, 2, or 3. With an overall accuracy of 96%, meaning it accurately predicts 96 out of 100 cases. The classification report for the RF model shows perfect performance on the dataset. The model achieved an accuracy, recall, and F1-score of 1.00 for class 0, based on 25 samples. Based on 28 samples, the model's accuracy is 1.00, recall is 0.89, and F1-score is 0.94 for class 1. The model's accuracy, recall, and F1-score for

	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	25
1	1.00	0.89	94	28
2	0.89	1.00	0.94	32
1	1.00	0.89	0.94	9
Accuracy			0.96	94
Macro avg.	0.97	0.95	0.96	94
Weighted avg.	0.96	0.96	0.96	94

Table 5.1: Classification Report of RF

class 2 are (1.00, 0.89, and 0.94), respectively, based on 28 samples. The model has an accuracy of 1.00, a recall of 0.89, and an F1-score of 0.94 for class 3, derived from 28 samples. Accuracy, recall, and F1-score all have weighted averages and macro averages of 0.96
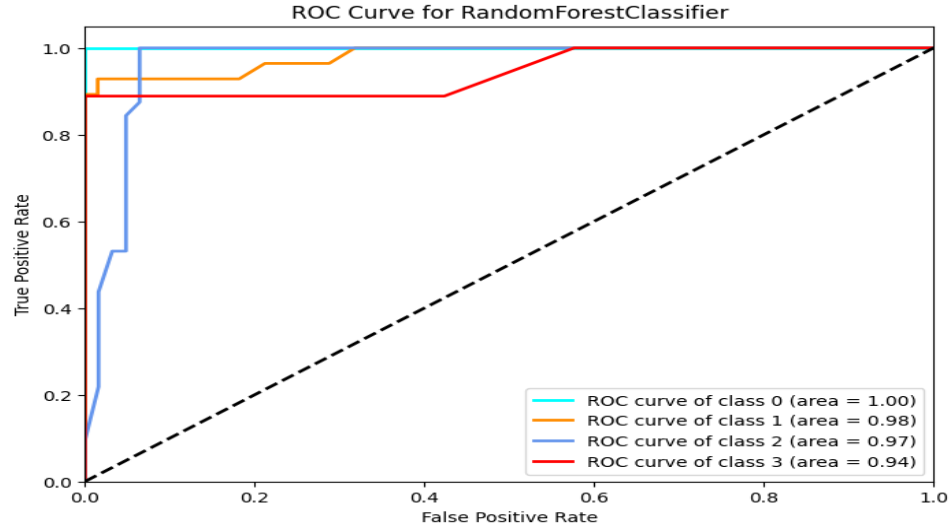
5.3.3 ROC Curve Analysis of RF

Figure 5.2: Illustrating the model's true positive versus false positive rates.

The ROC (Receiver Operating Characteristic) curve illustrates the performance of an RF classifier in categorizing four different classes (0, 1, 2, and 3). Here, each class represents the problem solver's name.

The ROC curve illustrates the True Positive Rate (TPR) on the y-axis in relation to the False Positive Rate (FPR) on the x-axis across various thresholds. The nearer the curve is to the top-left corner, the more proficient the model is in differentiating that class.

Each line represents a different class:

Class 0 (cyan): AUC = 1.00—perfect prediction.

Class 1 (orange): AUC = 0.98—outstanding.

Class 2 (blue): AUC = 0.97—excellent.

Class 3 (red): AUC = 0.94—very good.

The AUC (Area Under Curve) value spans from 0 to 1. An AUC nearing 1 indicates the model excels at distinguishing that class from the others. An AUC of 0.5 indicates random chance in predictions.
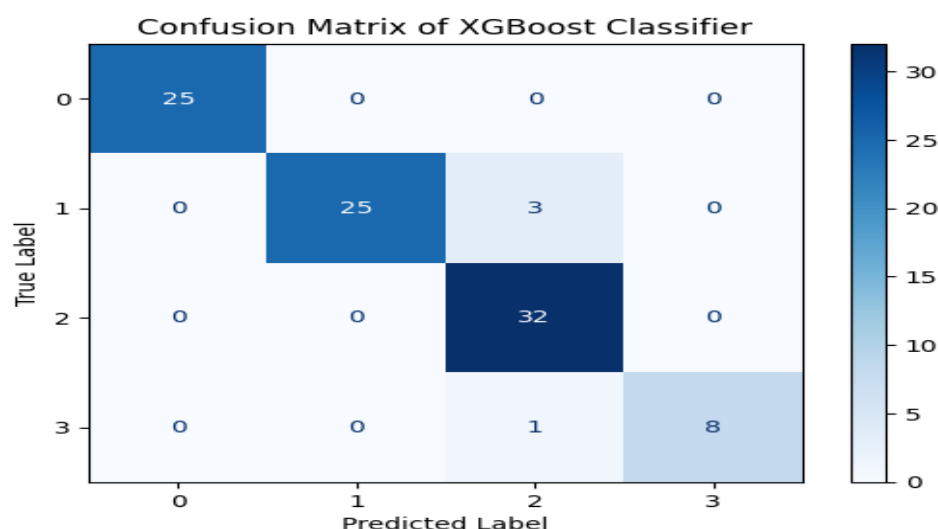
5.3.4 Confusion Matrix of XGBoost Classification

Figure 5.3: Showing correct and incorrect predictions for each class clearly.

The XGBoost confusion matrix in the original image is shown in Figure 5.3. The actual (real) labels are shown in the rows, and the anticipated labels are shown in the columns. The XGBoost confusion matrix classification procedure is equivalent to RF, since the confusion matrix XGBoost's output is similar to the RF classifiers.

5.3.5 Classification report of XGBoost

Four classes can be predicted by the XGBoost model employed in this classification report: 0, 1, 2, or 3. It predicts 96 out of 100 cases correctly, or an overall accuracy of 96%. Here, the explanation of the XGBoost classification report is the same as that of RF.

	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	25
1	1.00	0.89	94	28
2	0.89	1.00	0.94	32
1	1.00	0.89	0.94	9
Accuracy			0.96	94
Macro avg.	0.97	0.95	0.96	94
Weighted avg.	0.96	0.96	0.96	94

Table 5.2: Classification Report of XGBoost

5.3.6   ROC Curve Analysis of XGBoost:
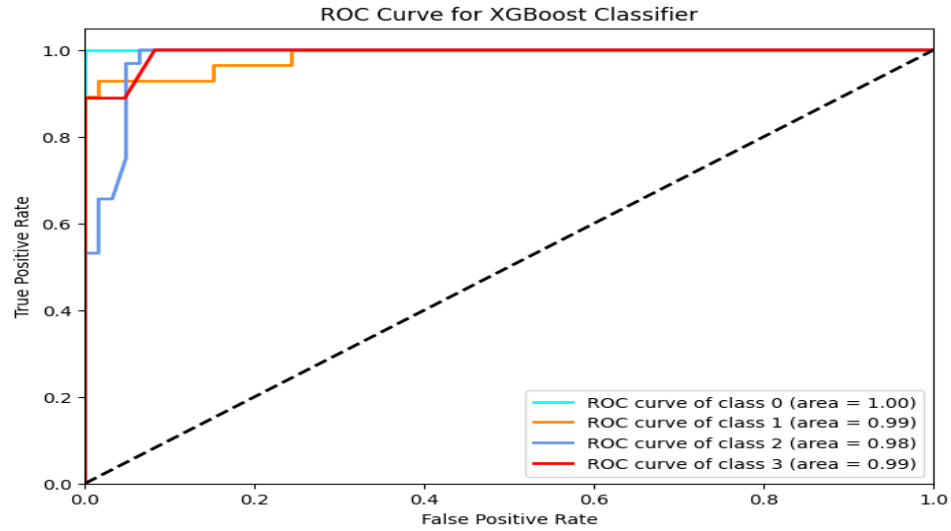
Figure 5.4: Showing the trade-off between true and false positive rates.

The effectiveness of an XGBoost classifier in categorizing four separate classes, 0, 1, 2, and 3, is displayed by this ROC (Receiver Operating Characteristic) curve. The explanation of this ROC curve is similar to the ROC curve of RF.

Each line represents a single class:

Class 0 (cyan): AUC = 1.00—perfect prediction.

Class 1 (orange): AUC = 0.99—almost perfect.

Class 2 (blue): AUC = 0.98—outstanding.

Class 3 (red): AUC = 0.99—almost perfect.

5.3.7   Confusion Matrix of LightGBM Classification

In Figure 5.5, the RF confusion matrix is shown in the first picture. The predicted labels are shown in the columns, and the actual (true) labels are shown in the rows.
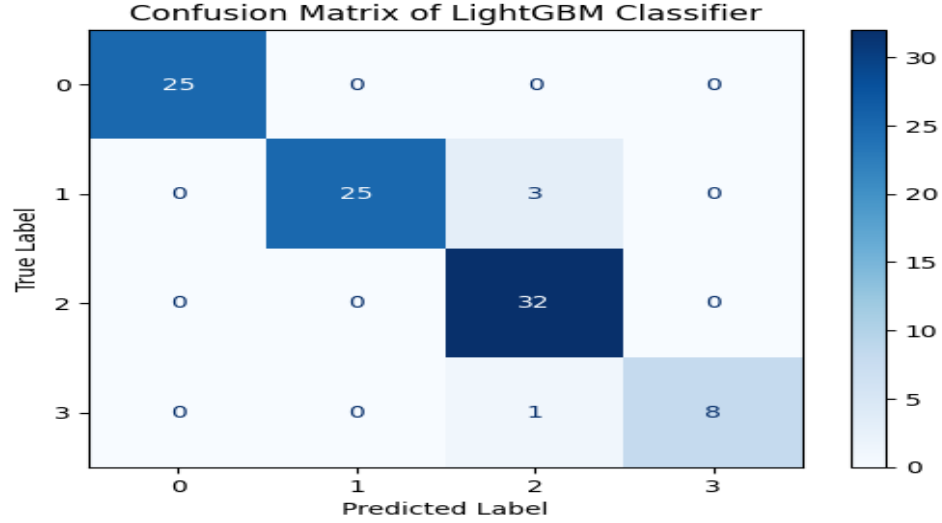
41

Figure 5.5: Showing the number of correct and incorrect predictions.

The models exhibit good accuracy, since the majority of predictions fall along the diagonal, indicating that true labels correspond with predicted labels. Only a limited number of errors occurred when class 1 and class 3 were misidentified as class 2.

5.3.8 Classification report of LightGBM

Table ??shows the LightGBM classification report. In this part, four classes are indicated in different solver names, and the illustration of LightGBM looks like RF classification.

	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	25
1	1.00	0.89	94	28
2	0.89	1.00	0.94	32
1	1.00	0.89	0.94	9
Accuracy			0.96	94
Macro avg.	0.97	0.95	0.96	94
Weighted avg.	0.96	0.96	0.96	94

Table 5.3: Classification Report of LightGBM

5.3.9 ROC Curve Analysis of LightGBM:

This Receiver Operating Characteristic (ROC) curve illustrates the performance of a LightGBM classifier in the classification of four unique classes: 0, 1, 2, and 3. The explanation of this ROC curve is comparable to that of RF.
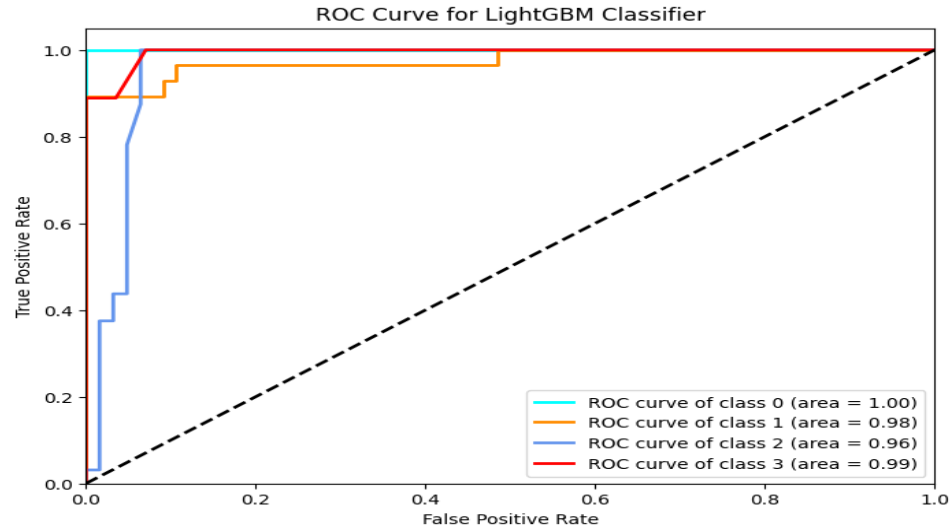
Figure 5.6: Showing the model's true positive rate against the false positive rate.

Here, each class represents a single solver name.

Class 0 (cyan): AUC = 1.00—perfect prediction.

Class 1 (orange): AUC = 0.98—outstanding.

Class 2 (blue): AUC = 0.96—excellent.

Class 3 (red): AUC = 0.99—almost perfect.

5.3.10 Performance of GBR and RF Regressor (Time Prediction)

The GBR and RF regressors are ML methods utilized for predicting continuous values. In this study, those algorithms were used for predicting product delivery time in days. The RFR model achieved strong capability in forecasting resolution time compared to the GBR model. Table 5.4 shows the performance of both regressions.

The performance of the RF regressor,

Mean Squared Error (MSE): 13.09

Root Mean Squared Error (RMSE): 3.62

R-squared (R^2) : 0.57

The performance of the GB regressor,

Mean Squared Error (MSE): 17.28

Root Mean Squared Error (RMSE): 4.16

R-squared (R^2) : 0.43

5.4 Interface of the project

Figure 5.7 shows an accuracy chart of various ML models, such as RF, XGBoost, and LightGBM. All three model accuracy scores are 96%. Visually, their ratings varied, but all models showed great predictive performance, corresponding with the ideal confusion matrices in the previous pictures.
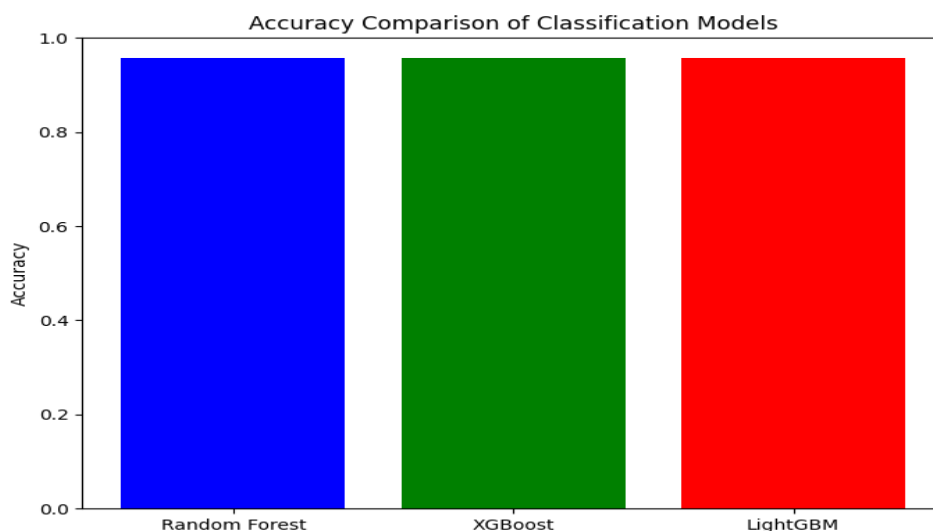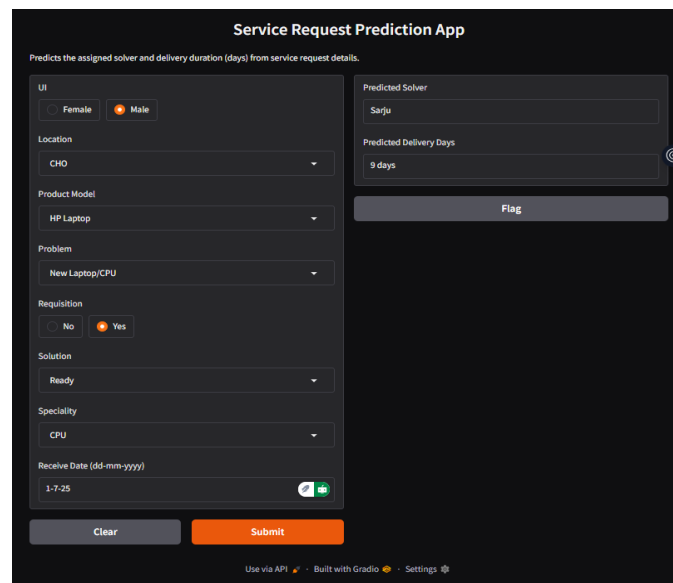

Figure 5.7: Accuracy chart of various Models

5.5 User Interface of the project

The project's user interface provides a user-friendly platform for IT hardware support systems in both small and large corporate offices. Almost certainly built using Python tools, the interface accepts various input data such as UI, Locations, ProductModel, Speciality, and more. After the information is submitted, the system utilizes one of the trained ML models (RF, XGBoost, LightGBM) to categorize the problem solver name, and also uses (RFR, GBR) for product delivery time (in days). This interface transforms complex ML predictions into straightforward, understandable outputs that support the study targets [SolvedBy (solver name) and Delivery Days]. In the project study, LightGBM and RFR algorithms are being used for the user interface. The user interface is shown in Figure 5.8


Figure 5.8: User interaction and output display.

5.6　Summary

This chapter presents the analysis of results for the predictive system of IT hardware services, utilizing various ML methods. The results of both targets (SolvedBy, Delivery Days) are excellent.

CHAPTER VI

Conclusion and Future Work

6.1   Conclusion

This study aims to introduce an intelligent, dual-model framework to address key inefficiencies in IT hardware support. By combining a high-accuracy LightGBM classifier for problem solver name with a robust GBR for resolution time forecasting, the study provides a comprehensive solution that converts a traditional, manual system to digital models. The results of this study for IT hardware maintenance are based on ML models, including a LightGBM classifier and a GB regressor. The design performed well with over 96% accuracy on the IT problem solver name and an average error of 3.62 days for the time forecast of resolution. However, the study has some drawbacks, as the models utilize a dataset from a singular organization with limited features, and accuracy is dependent on the dataset quality. Altogether, my study demonstrates that predictive analytics is not only a technology solution but also a strategic vehicle for developing intelligent, future-oriented IT service management. Finally, the system successfully predicted both targets, and the user interface showed those target results for IT servicing management to our corporate office.

6.2 Future works

In the future, an attempt will be made to upgrade the overall system by incorporating real-time data and datasets from numerous corporate offices to enhance the acceptance and normalization of the IT support system in Bangladesh and other countries. Our future work includes the creation of an automated ticket assignment for the system, which would make the platform more accessible, especially in large corporate offices. To ensure that the Intelligent Operations Framework (IOF) is efficient, the system must be constantly updated with new service data. As IT tickets continue to accumulate, the ML models need to periodically retrain themselves for new problems, hardware changes, and technician performance. This keeps the predictions up-to-date. The verification of data quality checks integrity was performed, as an erroneous value indicates issues with the system. The system can also be improved by adding new functionalities and evaluating more advanced models, such as deep learning. An attempt will be made to update the user interface for IT servicing management of any small or large corporate offices.

References

[1] L. T. C. H. J. D. M. S. Alexander Rokoss, Marius Syberg, "Case study on delivery time determination using a machine learning approach in small batch production companies," *Journal of intelligent manufacturing*, vol. 35, p. 3937–3958, 2024.

[2] K. S. S.P. Paramesh, "A deep learning based it service desk ticket classifier using cnn," *Ictact journal on soft computing*, vol. 145, pp. 675–679, 2018.

[3] A. Z. Dmitry Zuev, Alexey Kalistratov, "Machine learning in it service management," *The Annals of thoracic surgery*, vol. 38, no. 23, p. 1805–1814, 2017.

[4] T. H. Jan Wolter, "Prediction of service time for home delivery services using machine learning," *Soft computing*, vol. 28, no. 06, p. 5045–5056, 2024.

[5] S. P. Ananya Garg, Mohmmad Ayaan and V. Udandarao, "Food delivery time prediction in indian cities using machine learning models," *Indraprastha institute of information technology*, 2025.

[6] J. W. Branch, Y. Diao, and L. Shwartz, "A framework for predicting service delivery efforts using it infrastructure-to-incident correlation," pp. 1–8, 2014.

[7] W.-E. K. Tong-Ern Tai, Su-Cheng Haw and K.-W. Ng, "Performance evaluation of machine learning techniques on resolution time prediction in helpdesk support system," *International journal on robotics, automation and sciences*, vol. 6, no. 2, 2024.

[8] F. Al-Hawari and H. Barham, "A machine learning based help desk system for it service management," *Journal of King Saud University*

- *Computer and Information Sciences*, vol. 33, no. 6, pp. 702–718, 2021.

[9] A. Matsunaga and J. A. Fortes, "On the use of machine learning to predict the time and resources consumed by applications," pp. 495–504, 2010.

[10] S. Paramesh and K. Shreedhara, "Automated it service desk systems using machine learning techniques," pp. 331–346, 2018.

[11] T. P. Carvalho, F. A. Soares, F. R. d. P. Vita, Roberto, J. P. Basto, and S. G. Alcalá, "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers & Industrial Engineering*, vol. 137, p. 106024, 2019.

[12] S. J. L. Feng, Leon and Bill, "Tadaa: Real-time ticket assignment deep learning auto advisor for customer support, help desk, and issue ticketing systems," *arXiv preprint arXiv:2207.11187*, 2022.

[13] H. A. K. K. T.-I. M. K. K. M. Georgoulopoulos, Nikolaos and A. I, "A survey on hardware failure prediction of servers using machine learning and deep learning," pp. 1–5, 2021.

[14] K. Khalil, O. Eldash, A. Kumar, and M. Bayoumi, "Machine learning-based approach for hardware faults prediction," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 11, pp. 3880–3892, 2020.

[15] S. Varghese and J. Tian, "Efficient support ticket resolution using knowledge graphs," *arXiv preprint arXiv:2501.00461*, 2024.

[16] Y. Yang and H. Wang, "Random forest-based machine failure prediction: A performance comparison," *Applied Sciences*, vol. 15, no. 16, 2025.

[17] "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers and Industrial Engineering*, vol. 137, p. 106024, 2019.

Appendix