C O V E N T R Y

U N I V E R S I T Y

Faculty of Engineering, Environment and Computing

School of Computing, Electronics and Mathematics

MSc Data Science EECT109

7150CEM – Data Science Project

Predicting Pro-long Hospital Stay of Covid-19 Patients Using Machine Learning

Author: Virat Kanji Bamaniya

SID: 12249347

Supervisor: Long Chen

Submitted in partial fulfilment of the requirements for the Degree of Master of Science in MSc Data Science

Academic Year: 2022/23

Declaration of Originality

I declare that this project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed: Virat Kanji Bamaniya Date: 09/10/2022

Please complete all fields.

First Name:	Virat Kanji
Last Name:	Bamaniya
Student ID number	12249347
Ethics Application Number	P142450
1st Supervisor Name	Long Chen
2nd Supervisor Name	Mark Johnston

This form must be completed, scanned and included with your project submission to Turnitin. Failure to append these declarations may result in your project being rejected for marking.

Abstract

We forecast the result of current patients based on the data of previous patients using machine learning algorithms. Depending on which models are utilised, the forecasts' accuracy may change. Large amounts of data are helpful since machine learning algorithms rely on the data that is supplied. Computer forecasts are growing better every day to assist us in our daily lives. Humans can only process a certain amount of information before making a choice, thus we may be mistaken since visual perception and the enormous quantity of data that exists might differ greatly. Large data sets can be computed quickly by computers. As one of the most straightforward techniques, logistic regression is utilised as the baseline for machine learning models in this project. In order to compare each model and provide a clear and precise solution to our problem, four more algorithms were applied. The Extreme Gradient Boosting model is the approach that has the most potential for accuracy. Since its debut, it has developed into one of the most efficient machine learning algorithms and consistently outperforms the majority of other algorithms. Gradient Boosting, Random Forest, and K-nearest Neighbours are further methods for forecasting length of stay. The results of the studies reveal that the XGBoost model can predict the length of stay of COVID-19 patients with the highest accuracy; the accuracy reached was 60.91%. To make use of the entire dataset, the prediction was performed with various k-values in mind. A study of the literature revealed that no published work predicted the number of days that a patient would require a hospital bed. It is intended that this effort would assist hospitals and other healthcare facilities to use their resources more effectively and help saving more lives.

Table of Contents

Acknowledgements

I want to start by sincerely thanking my supervisor, Long Chen, for all of his wise counsel, unwavering encouragement, and patience. I also like to thank Mark Johnston , my 2nd supervisor and Dr. Rochelle Sassman, the module leader, for all of their help and support.

I want to express my deep gratitude to Sheldon, Krishna, and Ankit, three of my classmates, for your constant feedback assistance in helping me improve in a competitive setting and for building confidence.

Finally, I want to thank my friends and family for their unfailing support and understanding during this year.

1 Introduction

Humans can get respiratory illnesses from the coronavirus virus. The term "corona" refers to the virus's spikes, which resemble crowns as shown in Figure 1. Other respiratory diseases include the common cold, SARS (severe acute respiratory syndrome), and MERS (middle east respiratory syndrome). The new strain of the coronavirus known as SARS-CoV-2 was initially discovered in Wuhan, China, in December 2019. Following that, the virus spread to every country in the world.

Animals including camels, cats, and particularly bats were frequently reported to have coronaviruses. The viruses persist in the bodies of the animals but do not in any way infect them. Due of the virus's fast rate of mutation, it is possible for it to alter as it spreads from one animal to another. Eventually, the virus spread from animals to people. The first afflicted individuals contracted the disease after consuming an animal that had been exposed to the SARS-CoV-19 virus.

Considering how quickly the infection may travel via the air, the mouth, nose, and eyes are the main entry points for the virus. Additionally, you can contract an infection from an infected person by touching their sick hands. The mucous membrane in the back of your throat and nasal passages are the first areas of the body the virus impacts. It spreads by destroying other cells in your throat, after which it moves on to the lungs. The virus has the ability to spread to other places of your body after harming the lung tissues.

Even if you have the SARS-CoV-2 virus, it might take a few days before you start experiencing symptoms. You can spread the infection to other people by coming into touch with them, even if no symptoms are present. You will stop being infectious to other individuals after 10 days of the symptoms appearing. If you visited any area of the world where COVID-19 was reported, you may unknowingly be carrying the virus (Cleveland Clinic, 2020).

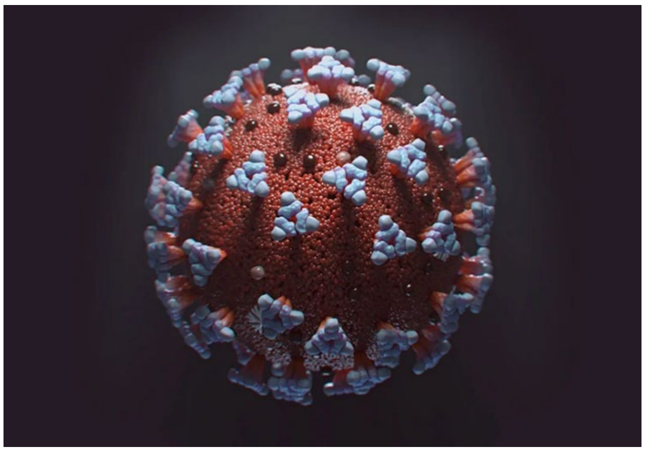


Figure 1: SARS-CoV-2 Virus showing the crown like structure on its cell membrane

You may avoid contracting COVID-19 in a variety of methods. The best strategy is to avoid crowded areas and keep a distance of 6 feet from other people whenever feasible. Masks and other coverings for your mouth and nose work wonders. It's highly recommended to wash your hands with hand soap several times during the day and to use hand sanitizer after coming into touch with someone else. Vaccination should be a top focus. Although no vaccination is 100% effective, the likelihood of contracting the virus is significantly decreased. Early detection of covid-19 symptoms can aid in at-home management of the condition. Extracorporeal membrane oxygenation (ECMO), which replaces the functions of the heart and lungs, and mechanical ventilation for oxygen are two distinct ways that different phases of COVID are treated.

It has been difficult for everyone in the world to deal with the COVID-19 epidemic. In the world, many occupations are performed in offices or at specialised locations rather than in homes. Even while many IT professions may be done from home, other tasks, like maintaining servers, still need to be done in person. People have modified their life as a result of the epidemic in several ways, including working from home, switching jobs, and changing occupations. However, there are others who are unemployed.

The medical sector has been hit the worst. Doctors, nurses, and other healthcare professionals still have jobs that require them to go to hospitals. They can't do their jobs remotely. During the epidemic, healthcare professionals had to take extra measures. Despite this, health professionals occasionally fell ill while at work. It was necessary to properly manage resources in every area of our society. In the epidemic, resources played a crucial role. At times like the epidemic, it becomes especially important to monitor resources and manage them effectively.

Many people were forced to wait for work to resume or figure out how to work from home because the majority of businesses, offices, and other workplaces had closed. The healthcare industry will always include working at a job site. Effective management was required of all healthcare resources. The hospitals become overcrowded as the virus spread. Many patients who required urgent care did not receive it.

Despite being on the road to recovery, some patients took up space at a hospital. People with COVID may be able to follow the recommended rehabilitation path while staying at home later on in their recovery. Additionally, home treatments might aid with recuperation. Resources in hospitals are constrained due to the restricted number of hospitals and available beds. As we get broader knowledge about COVID during this epidemic, we must acquire patient data and allocate resources wisely.

Here, the length of hospitalisation and the seriousness of the patient's disease are key factors. The number of hospitals in a city and the number of hospital beds are two examples of resources at the hospital that should be monitored. Since the severity of the sickness dictates the sort of hospital room needed, the types of beds are equally important. We can make effective judgments that benefit patients and hospitals by gathering data from many institutions and allocating resources where they are required. We can forecast how long a patient will stay in the hospital by applying data science. The information we have was gathered, thus it is entirely factual and was not at anyway conjectured or expected.

We may use data from previous patients to predict the outcome of present patients using machine learning algorithms. The usage of numerous distinct models may affect how accurate the forecasts are. Large amounts of data are helpful in the process since machine learning models depend on the data given. Computer forecasts are growing better every day to assist us in our daily lives. Numerous elements influence the relationship between a patient and a hospital. In one city, hospitals exist. Other hospitals in various cities may differ slightly in terms of their facilities, available treatments, or areas of specialisation.

Another consideration is the type of patient admitted. Is the patient experiencing an emergency and in need of a hospital resource right away? Even while some patients' needs may not be as urgent, they nevertheless require the usage of certain hospital amenities. Ages of patients are a significant factor that correlates with specific illnesses. For both extremely young and very old individuals, certain illnesses are deadly. While younger individuals can get by with minimal care before switching to bed care, older folks require immediate attention.

1.1 Background to the Project

The goal of this initiative is to assist in the right and effective distribution of hospital resource demands. The resources must be distributed fairly across all locations because the epidemic has afflicted everyone on earth. Wasting any kind of resource is undesirable. The healthcare department is the major one that has been continuously operating. I want to support those who work in the medical field. Any assistance, no matter how basic or difficult, will be of great assistance to those who perform midnight shifts in hospitals. In order to manage several hospitals or communicate with other hospitals, hospital managers also require a lot of assistance with managing the day-to-day resources of hospitals.

1.2 Project Objectives

Predicting how many days the patient will spend in the hospital is the major goal. It is quite challenging to obtain extremely accurate findings, such as a certain number of days. With really powerful computers, it is possible, but not with a laptop or PC used at home. An estimate is made of the patient's hospital stay in terms of days. The machine learning model loses accuracy when making many predictions. The accuracy would be much worse if you were forecasting, say, 10 outcomes than if you were forecasting just 2. The management of hospital time and resources still benefits greatly from the forecast of the range of days.

1.3 Overview of This Report

The paper begins by conducting research on other journals and publications that have conducted related studies. Then, any similar study papers discovered are thoroughly examined to see what more research we might do to effect change. The approach of the system employed in the provided report is presented after the literature study. The machine learning models utilised in the paper exhibit a mathematical and scientific methodology. The methodology section explains the rationale behind the algorithm's selection. We see the dataset and its metrics after the methods part. It provides us with knowledge about the columns in the dataset, how it was compiled, and the dataset itself. On the dataset, matrices like the confusion matrix and F1-score are utilised to determine its efficacy.

An essential component of a machine learning-based research project is data pre-processing and preparation. The outputs of the algorithms will likewise be contaminated, as if the data had not been thoroughly cleansed. To clean the dataset and prepare it for usage, many data purification techniques are performed. The dataset is then analysed using several machine learning algorithms. The next step is to calculate the accuracy of each model to determine which of the five models is the best. For the classification of three classes, the accuracy of the models provided is good, hovering around 60%.

2 Literature Review

The global healthcare systems were significantly impacted by the COVID-19 outbreak. focusing in particular on hospital bed availability. The pandemic heightened interest in the scientific community in developing useful countermeasures to this issue, such as forecasting tools for hospital stay length. Predicting which hospital resources COVID-19 patients will use has been the subject of several research, some of which have made use of machine learning. Regarding duration of stay, COVID-19, and hospital resources, there are a lot of associated studies and journals. However, I'm just bringing up material that is directly relevant to the project I'm working on. The discussion that follows examines relevant works and the methodologies they employed.

In order to forecast three major ICU resources, including ICU Admission, ICU Mortality, and Length of ICU Stay for COVID-19 patients, (Dan et al., 2020) created a machine learning model. The COVID-19-diagnosed 733 patients who were admitted as inpatients to the Huangpi Hospital of Traditional Chinese Medicine (Wuhan, China) between January and March 2020 made up the dataset. The study built binary classifiers for predicting the need for ICU and death/survival in ICU using a multivariate prediction method employing the top ten factors with the greatest AUC values. For the binary classification, they utilized support vector machine (SVM). To maintain an even size distribution throughout the various groups, the ensemble learning approach was used as well. The least absolute shrinkage and selection operator (LASSO) regression model was used to estimate, using many variables, how long survivors would need to stay in the ICU. This study demonstrates how machine learning can predict which COVID-19 patients require ICU admission, which patients will sadly still pass away even after being admitted to the ICU, and among survivors, the length of ICU stay. All forecasts were made 1–15 days before the patients were actually brought to the intensive care unit. The strong predictive power offers a quantitative benchmark for more effective resource planning and allocation in the ICU.

With this strategy, (Olivato et al., 2022) goal is to analyse and streamline the length of stay forecast. More precisely, the predictions are made on whether a patient will have a short stay (no more than seven days) or a long stay (more than 7 days). By collaborating with the Spedali Civili di Brescia Hospital in northern Italy, data from a total of over 6,000 hospitalised patients from March 2020 to December 2021 were submitted. There are two primary parts to both the SLOS and outcome models that are implemented. Iterative

Imputer with Bayesian Ridge Regression is the first option for handling missing values. With this method, the missing values of the output feature are predicted using a Bayesian Ridge Regression algorithm that has been trained on the new dataset. The actual machine learning algorithm is the second element. They took into consideration ensembles of Decision Trees with bagging (Random Forests, ExtraTrees), boosting (XGBoost, LightGBM), and feed-forward neural network approaches as learning algorithms. The F1 score and ROC-AUC with their best model both reached a value of 0.76 in the experimental findings, demonstrating strong performance in identifying patients with long and short stays. They specifically demonstrate how the WOC (Worst Outcome Censoring) technique may be used to improve performance.

The work of (Etu et al., 2022) is a retrospective and prognostic analysis of a prediction model's effectiveness. The objective of this study is to create a model for predicting the ED (Emergency Department) LOS of COVID-19 patients (i.e., LOS < 4 hours or >= 4 hours). To correct the unbalanced dataset, where observations from the minority class were randomly replicated, the synthetic minority oversampling method (SMOTE) was used. For their assessment, they used the learning techniques logistic regression, gradient boosting, decision trees, and random forests. In this study, the baseline machine learning approach is logistic regression (LR). From March 16, 2020, to December 29, 2020, patient-level data were retroactively pulled from the Henry Ford Hospital electronic health records (EHR) data repository. 57,665 people visited the ED overall throughout the research period. Based on a mix of patient demographics, comorbidities, and operational ED data, the study revealed important characteristics that are linked to extended stays in COVID-19 patients. To predict the ED LOS of COVID-19 patients, they creatively trained four prediction models on these variables. The models may be updated and retrained to be used in other EDs to predict COVID-19 patient LOS, even though they were first trained using locally obtained data and clinical information from Henry Ford Hospital.

The goal of this publication is to create machine learning models that can forecast and estimate the number of days COVID-19 patients in Saudi Arabia would stay in the ICU. This will help the health sectors better manage their resources and evaluate if they are prepared to accept new patients and deliver the treatment they require. (Alabbad et al., 2022) also want to pinpoint the characteristics that are most crucial in determining whether a patient needs an intensive care unit and how long they are likely to remain there. The dataset, which includes clinical and demographic information on 895 patients who visited the hospital and tested positively for COVID-19, was collected from King Fahad University Hospital in Dammam City, in the eastern region of Saudi Arabia. The classifiers used for this publication are the Ensemble model, Extreme Gradient Boosting XGBoost, and Random Forest RF. Random Forest RF and XGBoost are the most popular machine learning (ML) methods for predicting mortality, severity, and duration of stay in ICU. The model performance was the average of utilising four alternative k values since the KNN imputation technique may use multiple k values: k = 3, k = 5, k = 7, k = 10, k = 15 and provide varied results. In order to coordinate and manage the process of taking on new patients in the ICU and hospitals, healthcare professionals can use the generated model's high accuracy of 94.16% using Random Forest to anticipate the time when the ICU beds will be available. The results revealed that the most important factors influencing the requirement for ICU admission and the length of stay in the facility are age, CRP, and days requiring nasal oxygen assistance.

(Hong et al., 2020) used traditional descriptive statistics. The probability of ProLOS (Prolonged length of stay) was predicted using a multivariate regression model, with factors chosen using a stepwise methodology. Outside of Wuhan city, their study characterised the clinical features of COVID-19 and discovered that the sickness was less severe than in the main epidemic location. To predict a lengthy hospital stay, a multivariate model with good discrimination was created. In the Zhejiang province, a tertiary care hospital hosted the trial from January 20 until February 20, 2020. For the quantitative study, 75 verified COVID-19 cases were included. Only 70% of individuals, according to the study, said that their early symptoms included fever. This questions the validity of the present diagnostic standards, which exclude COVID-19 from suspicion unless there is a fever and evidence of a lower respiratory disease.

Standard techniques, like the Kaplan-Meier estimator, call upon previous assumptions that are unworkable in light of the available data. (López-Cheda et al., 2021) attempted to predict the time-to-event and event probabilities of patients who were hospitalised, without parametric priors and compensating for individual factors, using real-time surveillance data from the early weeks of the COVID-19 pandemic in Galicia (Spain). The dataset includes 10 454 verified COVID-19 cases that were reported from March 6 to May 7 of 2020 in Galicia (Northwest Spain). Their analysis only included the 2453 patients who were admitted to the hospital or ICU during that time since not all of them required hospitalisation. The number of days from the patient's initial admittance into the HW or ICU until they experienced the result of interest was the LoS that was taken into consideration for patients who had multiple HW and/or ICU hospitalizations; this was the first reported LoS. The regional public health agency, Dirección Xeral de Sade Pblica, provided the data. They utilised a non-parametric mixed cure model and evaluated how well it predicted hospital ward (HW)/ICU LoS compared to regularly employed approaches for predicting survival. They demonstrated that the suggested model outperformed conventional methods by offering more precise ICU and HW LoS estimations. Finally, they used a Monte Carlo technique to replicate COVID-19 hospital demand while applying their model estimates. They presented proof that, in order to effectively estimate HW and ICU occupancy as well as discharge or mortality outcomes, it is essential to account for sex, which is frequently ignored in prediction models. The model predicts that 11 ICU beds are sufficient to prevent overload in the ICU if age and sex are ignored when making predictions (unconditional model). The available ICU beds should be adjusted to 12 instead, since that ICU capacity will be surpassed for 18 days, in order to avoid the ICU from being overcrowded.

In order to better allocate resources and offer better patient care during the ongoing epidemic, (Henzi et al., 2021) aims to provide early projections of each patient's length of stay in the critical care unit. As of the snapshot date, June 15, 2020, there were 557 severely sick COVID-19 patients hospitalised to an ICU in Switzerland, of whom 481 had already been discharged from the ICU or had passed away. This represents 86.36% of the patients for whom the LoS is accessible. The COVID-19 dataset refers to all 473 patients for whom covariates and LoS observations are available. They created a novel semiparametric distributional index model based on variables that were available 24 hours after the patient's admission to the critical care unit. The Swiss Society of Intensive Care Medicine's Minimal Dataset was used to train the model on a sizable cohort of patients with acute respiratory distress syndrome. Then, they make unique predictions for each patient's length of stay in the RISC-19-ICU registry. Building on isotonic distributional regression, distributional index models (DIMs) are semi-parametric models for distributional regression (IDR). The whole conditional distribution of the LoS given variables may be estimated using a distributional regression model. The LoS of individual patients with COVID-19 may now be predicted probabilistically in a calibrated and informative manner according to a novel probabilistic model. Long-term sufferers could be identified early. The difference between the anticipated LoS distributions based on the training data is greater than the difference between the LoS distributions of female and male COVID-19 patients. Compared to female patients, the predictions for male patients are more consistent with the empirical distribution of the observed LoS of the COVID-19 patients. Female patients are more likely to have "longer than predicted" LoS than male patients because male COVID-19 patients act more similarly to patients in the training data than female COVID-19 patients do.

3 Methodology

The method for creating the prediction models to determine how long COVID-19 patients will spend in the hospital is presented in this part. It gives a summary of the methodology, a quick rundown of the machine learning techniques used, and information on how the dataset was prepared. The machine learning model was constructed as the initial phase. The next step was to get it ready for processing. After that, tests were run to determine which model performed the best. The constructed models were then subjected to assessment. Utilizing the constructed model for prediction purposes is the last stage.

3.1 Summary of the suggested strategy

The purpose of this study is to forecast how long COVID-19 patients will stay in hospitals. First, we pre-processed the data to fill in any gaps or determine whether any of the data were uneven. Second, we examined a variety of machine learning models and chose the best one for my study. Following that, the provided data can be used to train the machine learning models. The best methods for predicting a patient's duration of stay are then evaluated. To test the models' accuracy, experiments with different K-fold values were conducted and experiments with Hyperparameter Tuning and RandomizedSearchCV are also conducted. The issue of selecting a set of ideal hyperparameters for a learning algorithm is known as hyperparameter optimization or tuning.

3.2 Machine Learning

Machine learning (ML) uses statistical models and algorithms to help computers identify patterns in vast volumes of data. These patterns are then used by a model to predict or describe fresh data. In its most basic form, machine learning teaches a machine to learn without having its capabilities explicitly programmed. The simplest kind of machine learning, which is a subset of artificial intelligence, employs algorithms to separate data, learn from it, and then make judgments or predictions about things in the actual world. In other words, machine learning creates models automatically from data that is supplied into a platform by using algorithms. The expertise of an expert is typically captured in programmed rules for rule-based systems, but when data is changing, it can be challenging to update and maintain these rules. The benefit of machine learning is that it can make data-driven probabilistic predictions and learn from growing amounts of data supplied into the algorithms. This capacity to swiftly and successfully use and apply extremely complicated algorithms to today's big data applications is a relatively recent development. The basic architecture of an active machine learning algorithm is depicted in the Figure 2 (*What Is Machine Learning?*, n.d.).

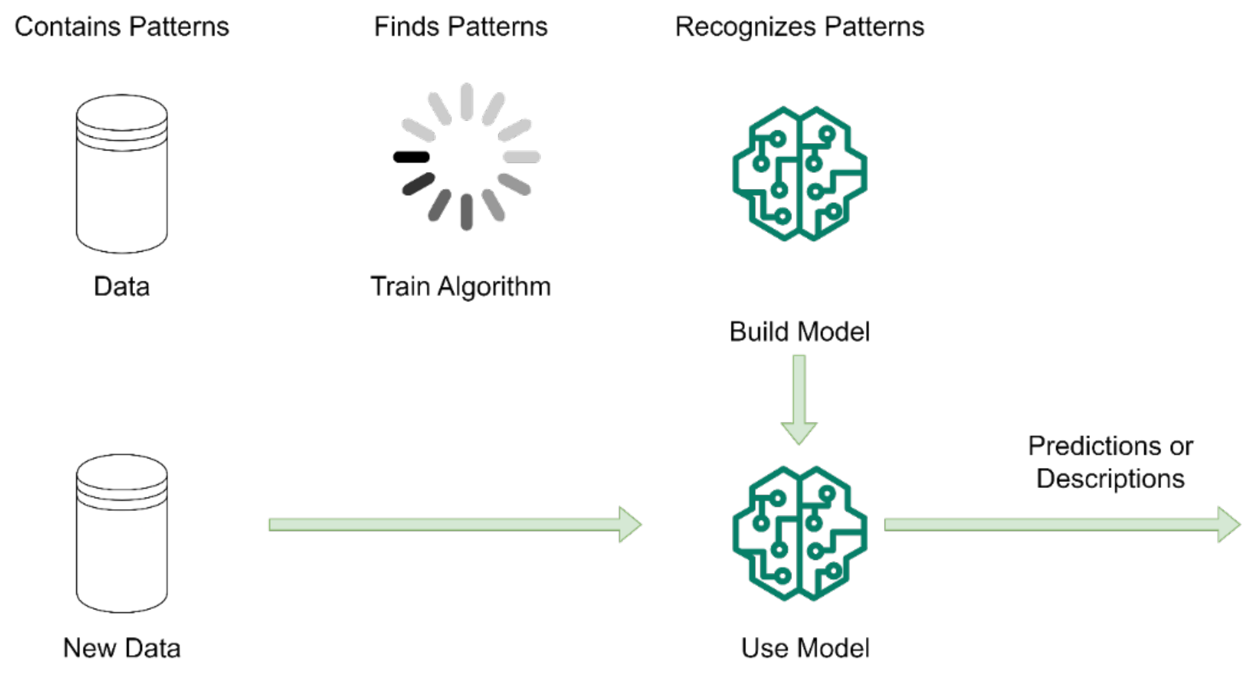


Figure 2: In simple terms training a model can be shown as the above image

3.3 Machine Learning Algorithms

Since there are several diverse machine learning methods, we may analyze them and examine the model accuracy. One model serves as our baseline model. Then, using various parameters, we compare the models and assess their accuracy. The models that are taken into account are chosen based on their capacity for prediction and popularity. The following models are taken into consideration: Logistic Regression, Random Forest, K-Nearest Neighbors, Gradient Boosting, and Extreme Gradient Boosting.

3.3.1 Logistic Regression

A machine learning model called logistic regression is utilised in a variety of real-world applications in a wide range of sectors, including health and the biological sciences, credit scoring and the financial system, sports and gaming matchup predictions, and many more. The logit function is the foundation for logistic regression. In essence, logit ties the distribution of the dependent variable to the independent variable. The log of odds, or "logit," is a mathematical formula that describes the ratio of success to failure. The natural log of the chances in favour of the positive event is returned by the logit function, which accepts the probability of the positive event as a variable (So, 2022).

$$logit(p(x)) = ln(\frac{p(x)}{1-p(x)})$$

Equation 1: The logit function

The inverse of the logit function is The Sigmoid Function. In essence, it calculates the likelihood that the positive event will occur given the inputs. It is continuous, nonlinear, and limited between Y=0 and Y=1. Overall, the logit and sigmoid functions play a significant role in logistic regression.

$$\phi(z) = \frac{1}{1+e^{-z}}$$

Equation 2: The Sigmoid Function

Where

$$z = logit(p(y=1|x))$$

Equation 3: Value of z

Interpretation of terms

In the above equations, the terms used are as follows (Wikipedia Contributors, 2019):

- The logit function is logit(p(x)). The logit(p(x)) equation shows that the logit, often known as the log-odds or natural logarithm of the odds, is equal to the expression for linear regression.
- In stands for the natural logarithm.
- p(x) is the probability that the dependent variable equals a case, given some linear combination of the predictors. The probability p(x) falls between 0 and 1.
- base e is a symbol for the exponential function.

3.3.2 Random Forest

The result of a random forest algorithm, which employs an ensemble learning technique made up of several decision trees, is the general consensus on the best solution to the issue. The wisdom of the crowds is applied to data science through ensemble learning approaches, which integrate many machine learning (ML) algorithms to create a superior model. They are predicated on the idea that a group of individuals with varying levels of expertise may come to a solution to a problem more effectively than a single person with more expertise. Decision trees, a common metaphor for addressing problems, are a component of random forest. A sequence of true/false questions concerning components of a data collection are used by decision trees to arrive at an answer as show in Figure 3.

A method known as "bagging" is used by random forest to construct entire decision trees concurrently using random bootstrap samples of the data set and features. Because of randomness, bias is less likely to occur because individual trees have minimal correlations with one another. The issue of overfitting, which happens when a model integrates too much "noise" in the training data and makes bad judgments as a result, is further mitigated by the existence of numerous trees (*What Is a Random Forest?*, n.d.).

Thanks to bagging and replacement sampling, it manages missing values and keeps excellent accuracy even when there are significant quantities of missing data. The model is a useful tool for dimensionality reduction since it can handle very big data sets with hundreds of input variables. Although they perform better than decision trees, gradient-boosted tree ensembles like XGBoost have higher accuracy than random forests. It works well for both classification and regression issues.

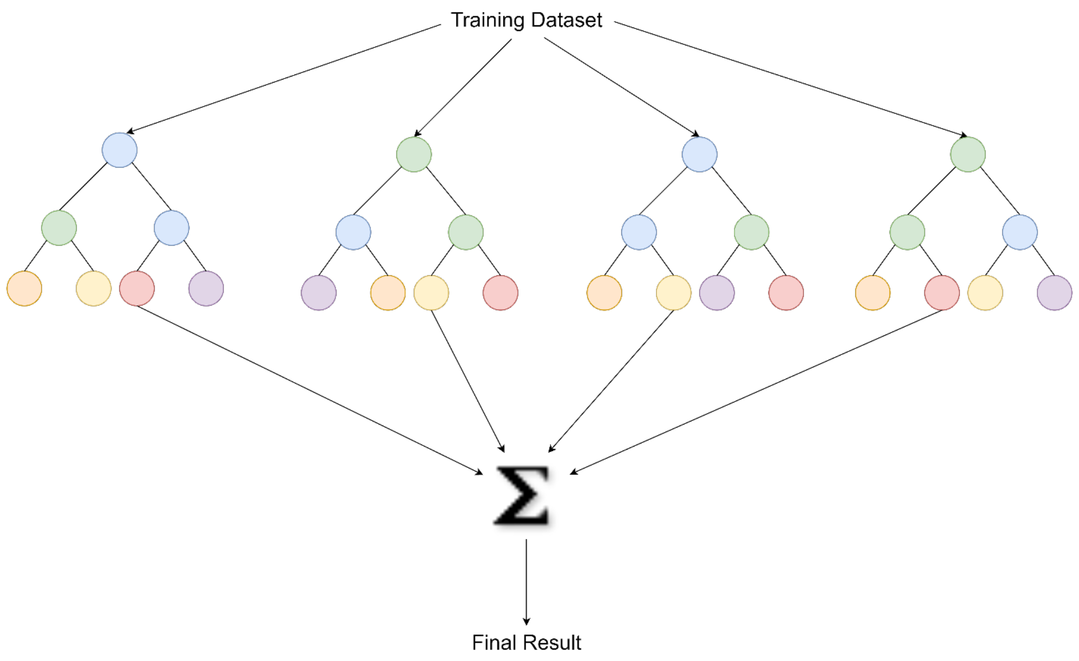


Figure 3: Random Forest Tree is just like a tree with calculating the average of all trees at the end

3.3.3 K-Nearest Neighbours

The k-nearest neighbours algorithm, sometimes referred to as KNN or k-NN, is a supervised learning classifier that employs proximity to produce classifications or predictions about the grouping of a single data point. Although it may be applied to classification or regression issues, it is commonly employed as a classification method since it relies on the idea that comparable points can be discovered close to one another (JavaTpoint, n.d.).

Noting that the KNN technique just saves a training dataset rather than going through a training phase, it should be noted that it belongs to a family of "lazy learning" models. Additionally, this implies that all calculation takes place at the time a classification or prediction is being formed. It is also called an instance-based or memory-based learning approach since it significantly depends on memory to retain all of its training data.

Despite its declining popularity, it is still one of the fundamental algorithms in data science because of how straightforward and precise it is. But as a dataset expands, KNN becomes less effective, lowering the performance of the whole model. Simple recommendation systems, pattern recognition, data mining, financial market forecasting, intrusion detection, and other applications are among its frequent uses.

3.3.4 Gradient Boosting

Particularly with huge and complicated data, gradient boosting is a method gaining interest for its prediction speed and accuracy. Individual machine learning models or an ensemble of models can be fitted to data. A group of straightforward individual models joined together to form an ensemble form a more potent new model. An ensemble may be made using machine learning boosting. Starting with the data, an initial model is fitted (such as a tree or a linear regression). Then, a second model is created with the goal of successfully forecasting the scenarios in which the first model fails. These two models should perform better together than they would separately. Then you numerously carry out this boosting technique (Hoare, 2017).

One kind of machine learning boosting is gradient boosting. It is predicated on the hunch that when prior models are coupled with the best feasible upcoming model, the overall prediction error is minimised. Setting the desired results for this subsequent model in order to reduce mistake is the important concept. Because target outcomes are determined for each case based on the gradient of the error with respect to the prediction, this method is known as gradient boosting. In the space of potential predictions for each training instance, each new model moves in the direction that minimises prediction error.

3.3.5 Extreme Gradient Boosting

Extreme Gradient Boosting, often known as XGBoost, is a decision tree-based machine learning technique that employs boosting to enhance performance. Since its debut, it has emerged as one of the most successful machine learning algorithms, consistently outperforming the majority of other algorithms, including normal decision trees, the random forest model, and logistic regression. The gradient boosting solution XGBoost pushes the limits of processing power for boosted tree algorithms. It is scalable and incredibly accurate. Its main purpose when it was developed was to improve the efficiency and performance of machine learning models. Trees are constructed using XGBoost in parallel as opposed to GBDT's sequential method. It employs a level-wise approach, scanning over gradient values and assessing the quality of splits at each potential split in the training set using these partial sums (Clarke, 2021).

For data scientists, it is significant that XGBoost and XGBoost machine learning models offer the best efficiency of prediction performance and computation time when compared to other methods. This has been confirmed by several benchmarking studies, which also explains why data scientists find it appealing. Initially, XGBoost implementations in both Python and R were created. Today, XGBoost includes package implementations for Java, Scala, Julia, Perl, and other languages as a result of its widespread use. The XGBoost library is now accessible to more developers thanks to these implementations. XGBoost was recognised as one of InfoWorld's prestigious Technology of the Year award winners in 2019 (Nvidia, n.d.).

While the XGBoost model frequently surpasses the accuracy of a single decision tree, it does so at the expense of decision trees' inherent interpretability. Following the decision-making route of a decision tree, for instance, is simple and obvious, but it is considerably more difficult to follow the paths of hundreds or thousands of trees. Some model compression strategies allow turning an XGBoost into a single "born-again" decision tree that roughly approximates the same decision function in order to achieve performance and interpretability. A second order Taylor approximation is utilised in the loss function to create the connection to the Newton Raphson technique, which is how XGBoost operates in function space as opposed to gradient boosting, which operates in function space as gradient descent. XGBoost algorithm is below (Wikipedia Contributors, 2019b).

Input: training set $\{(x_i, y_i)\}_{i=1}^{N}$, a differentiable loss function $L(y, F(x))$, a number of weak learners M and a learning rate α.

Algorithm:

1. Initialize model with a constant value:

$$\hat{f}_{(0)}(x) = \arg\min_{\theta} \sum_{i=1}^{N} L(y_i, \theta).$$

2. For m = 1 to M:
   1. Compute the 'gradients' and 'hessians':

$$\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}_{(m-1)}(x)}.$$

$$\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x)=\hat{f}_{(m-1)}(x)}.$$

   2. Fit a base learner (or weak learner, e.g. tree) using the training set $\left\{x_i, -\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)}\right\}_{i=1}^{N}$ by solving the optimization problem below:

$$\hat{\phi}_m = \arg\min_{\phi \in \mathbf{\Phi}} \sum_{i=1}^{N} \frac{1}{2}\hat{h}_m(x_i)\left[-\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)} - \phi(x_i)\right]^2.$$

$$\hat{f}_m(x) = \alpha\hat{\phi}_m(x).$$

   3. Update the model:

$$\hat{f}_{(m)}(x) = \hat{f}_{(m-1)}(x) + \hat{f}_m(x).$$

3. Output $\hat{f}(x) = \hat{f}_{(M)}(x) = \sum_{m=0}^{M} \hat{f}_m(x).$

Equation 4: A generic unregularized XGBoost algorithm

4 Dataset and Metrics

4.1 Data Collection Method

Choose the approach that is most appropriate for your study based on the information you wish to gather. A quantitative approach is used mostly in experimental research. Qualitative research techniques include ethnographies, focus groups, and interviews. Both quantitative and qualitative approaches may be used for collecting secondary data through surveys, observations, archival research, and more. It is easier to immediately respond to your research questions when you carefully evaluate the approach you will employ to collect data (Bhandari, 2020).

My approach of gathering data is referred to as secondary data collecting. Since Covid-19 is a pandemic, data is kept for a variety of purposes. The more data I have access to, the better I can forecast and base my model on because my endeavour is about prediction. Access to primary data is possible, but it won't be more efficient. Primary data is unique information that researchers gather with a specific goal in mind. On the other hand, secondary data is gathered for a different reason than the one for which it is used.

4.1.1 Secondary data collection

Any dataset gathered by a party other than the one using it is referred to as secondary data (also known as second-party data). Secondary sources of information are quite helpful. Large, superior databases that aid in resolving corporate issues may be created by researchers and data analysts thanks to them. Analysts can improve the quality and accuracy of their findings by including secondary data into their databases. Secondary data is primarily obtained from other organisations. But within an organization, collected data that is later repurposed is also referred to as secondary data. Secondary data offers a number of advantages and disadvantages (Hillier, 2022).

The affordability of employing secondary data analysis is one of its most obvious benefits. The researcher does not have to spend any money, time, or effort on the data gathering phases of their study because those phases have already been completed by someone else. The enormous quantity and range of data that are now made available to the public is another advantage of evaluating secondary data as opposed to gathering and analysing original data. The ability to improve and enlarge current datasets is another advantage of secondary data. Strategies for collecting primary data might also be influenced by secondary data. They can provide researchers or analysts early insights regarding the kinds of data they might wish to gather themselves in the future (Foley, 2018).

A secondary data set could not have the precisely specified information that would enable a researcher to answer his or her query if the researcher embarks on a study with a very specific inquiry in mind. It is impossible to know the lengths to which the researchers who collected the data went to ensure validity or quality, or if they encountered problems like low response rates or respondents misinterpreting what a question was really asking, without actually developing surveys and distributing them to the appropriate populations. The researcher performing the study has little control over the content of their secondary data collection because they did not gather the data they would be using.

4.2 Dataset

Due to the COVID-19 pandemic, healthcare needs in hospitals and intensive care units (ICUs) throughout the world have increased at a rate that has put an unprecedented pressure on health systems. Determining the consequent demands for hospital resources (beds, personnel, and equipment) has become a top issue for many nations as the epidemic worsens. Estimates of how long patients with COVID-19 will require various levels of hospital care are necessary to project future demand. Patient duration of stay is one crucial statistic to monitor and forecast if one wishes to increase the effectiveness of healthcare management in a hospital, even if there are many use cases for data science in healthcare management.

At the time of admission, this metric assists hospitals in identifying patients who pose a high LOS risk (patients who will remain longer). Once identified, patients at high risk for LOS can have their treatment plans improved to reduce LOS and lessen the possibility of infection in staff or visitors. Additionally, previous awareness of LOS might help with planning practicalities like room and bed allotment. The challenge is to oversee the professional and ideal operation of hospitals.

Dataset files are in CSV format. The data in the CSV file might unexpectedly alter when it is opened in Excel as Excel don’t properly open CSV files. More than 450k patients make up the dataset. Over 310k rows of data are in the training CSV file, while over 130k rows of patient data are in the test CSV file. The testing dataset does not have the column data that is to be predicted i.e., the length of a patient's hospital stay. Further information about the columns in the given dataset can be seen below.

Table 1: The columns of the dataset used in the training of machine learning models

Column name	**Datatype**	**Description**
case_id	int64	This is the patient's identification number for their hospital case.
Hospital_code	int64	Using this, you can identify the Hospital's unique code.
Hospital_type_code	object	Distinct code for the hospital type
City_Code_Hospital	int64	The city where the hospital is located is designated by this code.
Hospital_region_code	object	The region in which the hospital is situated is indicated by this code.
Available Extra Rooms in Hospital	int64	This indicates the amount of additional rooms the hospital has available.
Department	object	Hospital division in charge of the patient's case
Ward_Type	object	Code indicating the Ward type
Ward_Facility_Code	object	Facility code for the Ward
Bed Grade	float64	This depicts the current condition of the ward's beds.
patientid	int64	Patient's individual identification number
City_Code_Patient	float64	City code of the where the patient is located
Type of Admission	object	The hospital's recorded admission category for the patient

Severity of Illness	object	Severity of the patient's illness as reported when they were admitted
Visitors with Patient	int64	Count of people who visited the patient in the hospital
Age	object	Age of the patient within a certain range
Admission_Deposit	int64	Patient-provided deposit made at the time of hospital admission
Stay	object	The patient's hospital stay was measured in days range

4.3 Metrics

Starting with the terminology of a certain field is sometimes challenging. To utilise the tools, we must be able to recall a variety of terminology related to machine learning. Precision, Recall, and F1-Score are some fundamental concepts. These have to deal with gaining a more detailed understanding of a classifier's performance as opposed to focusing only on overall accuracy (Kanstrén, 2020).

True/False Positives and Negatives

Instances can be classified as positive or negative using a classifier.

Positive instances are those that are identified as belonging to the class the classifier is attempting to identify. For instance, a classifier searching for images of houses might categorise such images as positive.

Negative instances are those that are designated as not belonging to the class we are attempting to identify. For instance, a classifier searching for images of houses should mark images of temples (and no houses) as negative.

The ideas of True Positive, True Negative, False Positive, and False Negative serve as the foundation for precision, recall, and F1-Score.

4.3.1 F1 Score

Precision: How many of my positive forecasts actually came true? If you want to reduce false positives, precision is a wonderful area to concentrate on.

$$\text{Prediction} = \frac{\boxed{\text{TP}}}{\boxed{\text{TP}} + \boxed{\text{FP}}}$$

Equation 5: Number of True Positives (TP) divided by the Total Number of True Positives (TP) and False Positives (FP)

Recall: How many of the actual positive examples were ones that I accurately predicted to be positive? In fields like medicine, where you really want to reduce the possibility of missing positive cases, recall is crucial. In most of these situations, it is more expensive to overlook a positive case than to label something incorrectly as positive.

$$\text{Recall} = \frac{\boxed{\text{TP}}}{\boxed{\text{TP}} + \boxed{\text{FN}}}$$

Equation 6: Number of True Positives (TP) divided by the Total Number of True Positives (TP) and False Negatives (FN)

We should consider at both precision and recall in order to evaluate model performance thoroughly. The F1 score is a useful statistic that takes into account both of them.

$$\text{F1 Score} = \frac{\boxed{\text{TP}}}{\boxed{\text{TP}} + \frac{1}{2}(\boxed{\text{FP}} + \boxed{\text{FN}})}$$

Equation 7: F1 Score in terms of True Positive (TP), False Positive (FP), and False Negative (FN)

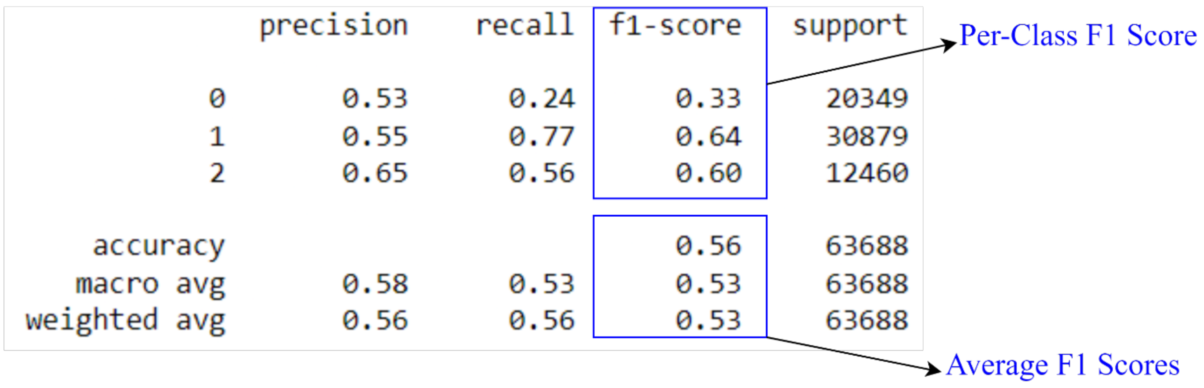

```
              precision    recall  f1-score   support

           0       0.53      0.24      0.33     20349
           1       0.55      0.77      0.64     30879
           2       0.65      0.56      0.60     12460

    accuracy                           0.56     63688
   macro avg       0.58      0.53      0.53     63688
weighted avg       0.56      0.56      0.53     63688
```

Figure 4: This is the Classification Report for the Logistic Regression

Macro Average F1 Score: The macro average is the simplest average since it just takes into account the per-class f1 scores.

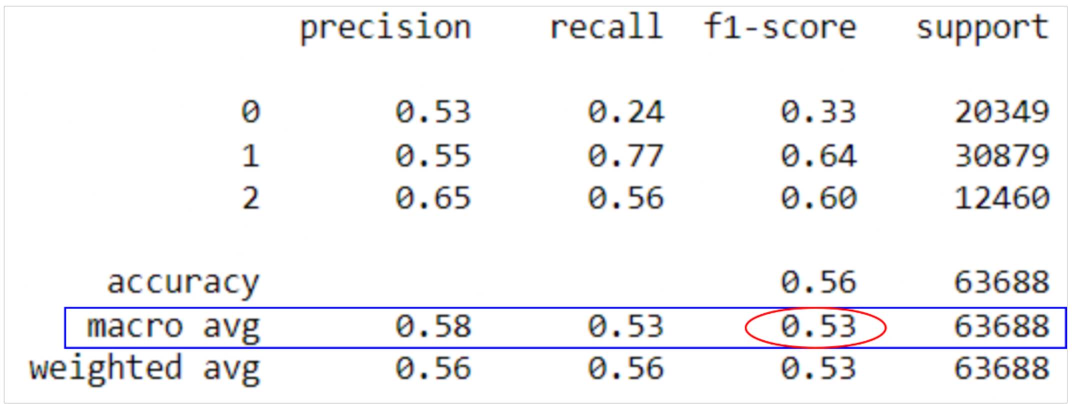
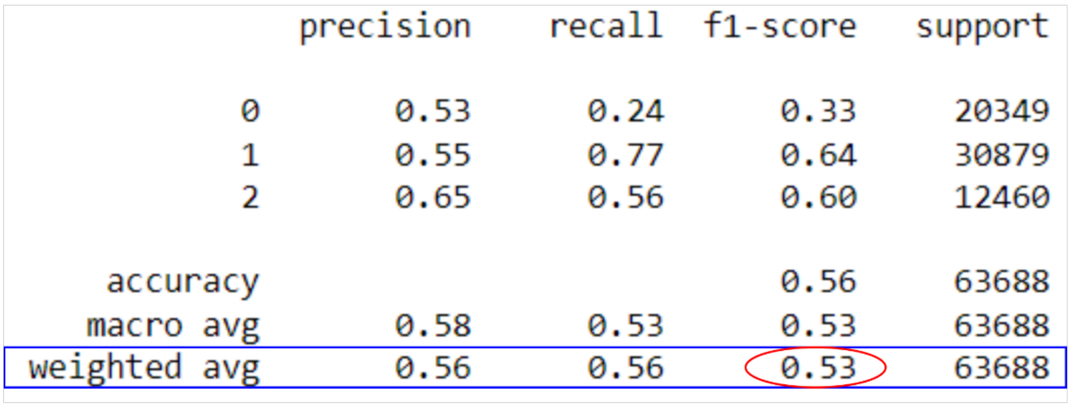

```
              precision    recall  f1-score   support

           0       0.53      0.24      0.33     20349
           1       0.55      0.77      0.64     30879
           2       0.65      0.56      0.60     12460

    accuracy                           0.56     63688
   macro avg       0.58      0.53      0.53     63688
weighted avg       0.56      0.56      0.53     63688
```

Figure 5: The red circle shows us the Macro Average F1 Score

Weighted Average F1 Score: The weighted-averaged F1 score is determined by averaging all the F1 scores for each class while taking into account the actual occurrences of that class.

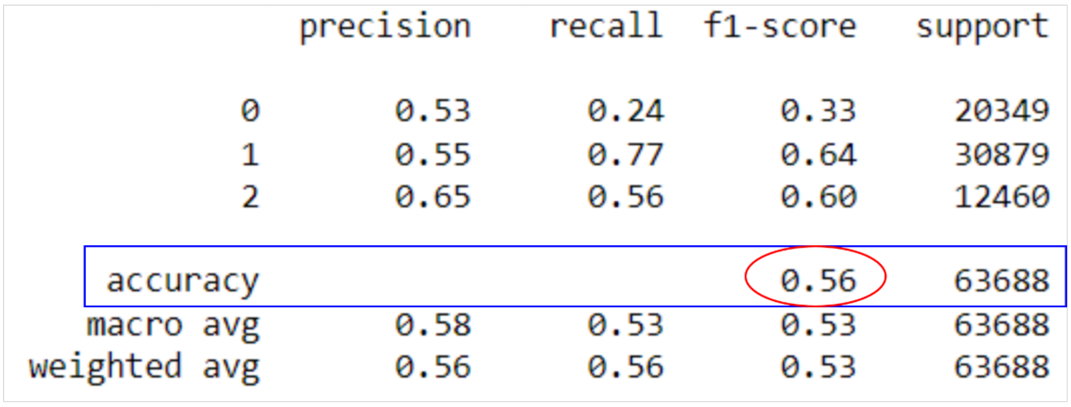


```
              precision    recall  f1-score   support

           0       0.53      0.24      0.33     20349
           1       0.55      0.77      0.64     30879
           2       0.65      0.56      0.60     12460

    accuracy                           0.56     63688
   macro avg       0.58      0.53      0.53     63688
weighted avg       0.56      0.56      0.53     63688
```

Figure 6: In the above image the red circle shows the Weighted Average F1 Score

Micro Average F1 Score:

By adding together the True Positives (TP), False Negatives (FN), and False Positives (FP), micro averaging creates a global average F1 score. In order to get our micro F1 score, we first add the corresponding TP, FP, and FN values across all classes. You might be asking why there is no row saying "micro average" and why our micro F1 score of 0.56 is represented as "accuracy". This is due to the fact that micro-averaging simply calculates the percentage of properly categorised observations among all observations. We can compute total accuracy using this definition (Leung, 2022).



```
              precision    recall  f1-score   support

           0       0.53      0.24      0.33     20349
           1       0.55      0.77      0.64     30879
           2       0.65      0.56      0.60     12460

    accuracy                           0.56     63688
   macro avg       0.58      0.53      0.53     63688
weighted avg       0.56      0.56      0.53     63688
```

Figure 7: The red circle shows out Micro Average F1 Score which is also our Accuracy

Since micro-F1, micro-precision, and micro-recall all have the same value, the classification report only has to provide a single accuracy rating.

4.3.2 Confusion Matrix

A confusion matrix is a table that lists how many guesses a classifier made correctly and incorrectly. It is employed to evaluate a classification model's effectiveness. It may be used to calculate performance measures like accuracy, precision, recall, and F1-score in order to assess the effectiveness of a classification model. Because they provide a more accurate picture of a model's performance than classification accuracy provides, confusion matrices are frequently employed. High TP and TN rates and low FP and FN rates are indicators of a strong model. It's usually preferable to utilise the confusion matrix as your machine learning model's assessment criterion when working with an unbalanced dataset (Suresh, 2020).

There are no positive or negative classes in multi-class classification, in contrast to binary classification. Since there are no positive or negative classes, it could be first challenging to locate TP, TN, FP, and FN, but it's actually rather simple (Mohajon, 2020).

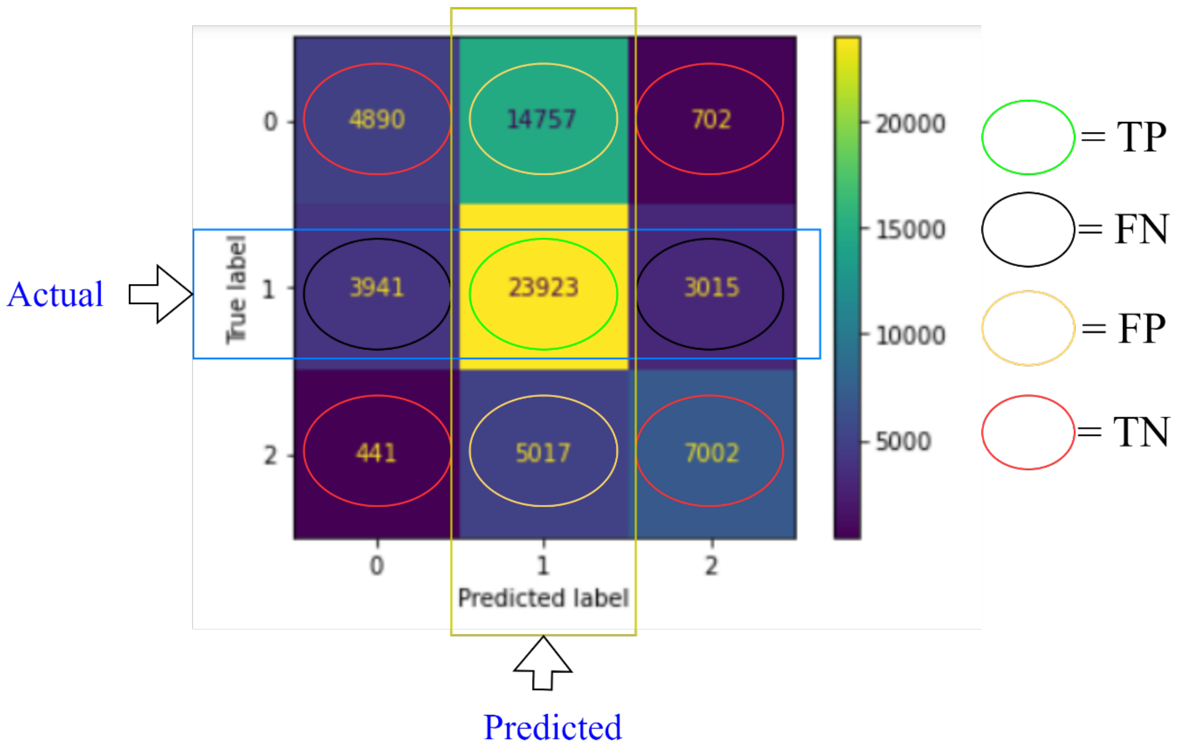


Figure 8: This shows the confusion matrix containing 3 classes indicating the TP, FN, FP, TN of class label 1

4.3.3 Correlation

A statistical metric called correlation describes how closely two variables are related to one another. Positive and negative correlations are the two primary categories. When two variables move in the same direction, there is a positive correlation; when one grows, the other increases as well. When two variables change in the opposite directions—when one rises, the other falls—there is a negative correlation. To test theories regarding the causes and effects of relationships between variables, correlation can be utilised. In the actual world, correlation is frequently utilised to forecast trends.

The correlation coefficient can have any value between -1 and 1. It is argued that there is a positive correlation between two variables if the value is 1. In other words, as one variable rises, the other variable rises as well. It is argued that there is a negative correlation between the two variables if the value is -1. In other words, as one variable rises, the other variable falls. There is no association between the two variables if the value is 0. This indicates that the variables' relationships to one another alter randomly (Kumar, 2022).

A form of graphic called a correlation heatmap shows the strength of correlations between numerical variables. To determine which variables are connected to one another and how strongly they are associated, correlation graphs are utilised. Typically, a correlation plot includes a number of numerical variables, each of which is represented by a column. The relationships between each pair of variables are shown by the rows. The correlation heatmap produced to visualise the linear relationship between several variables is shown below.

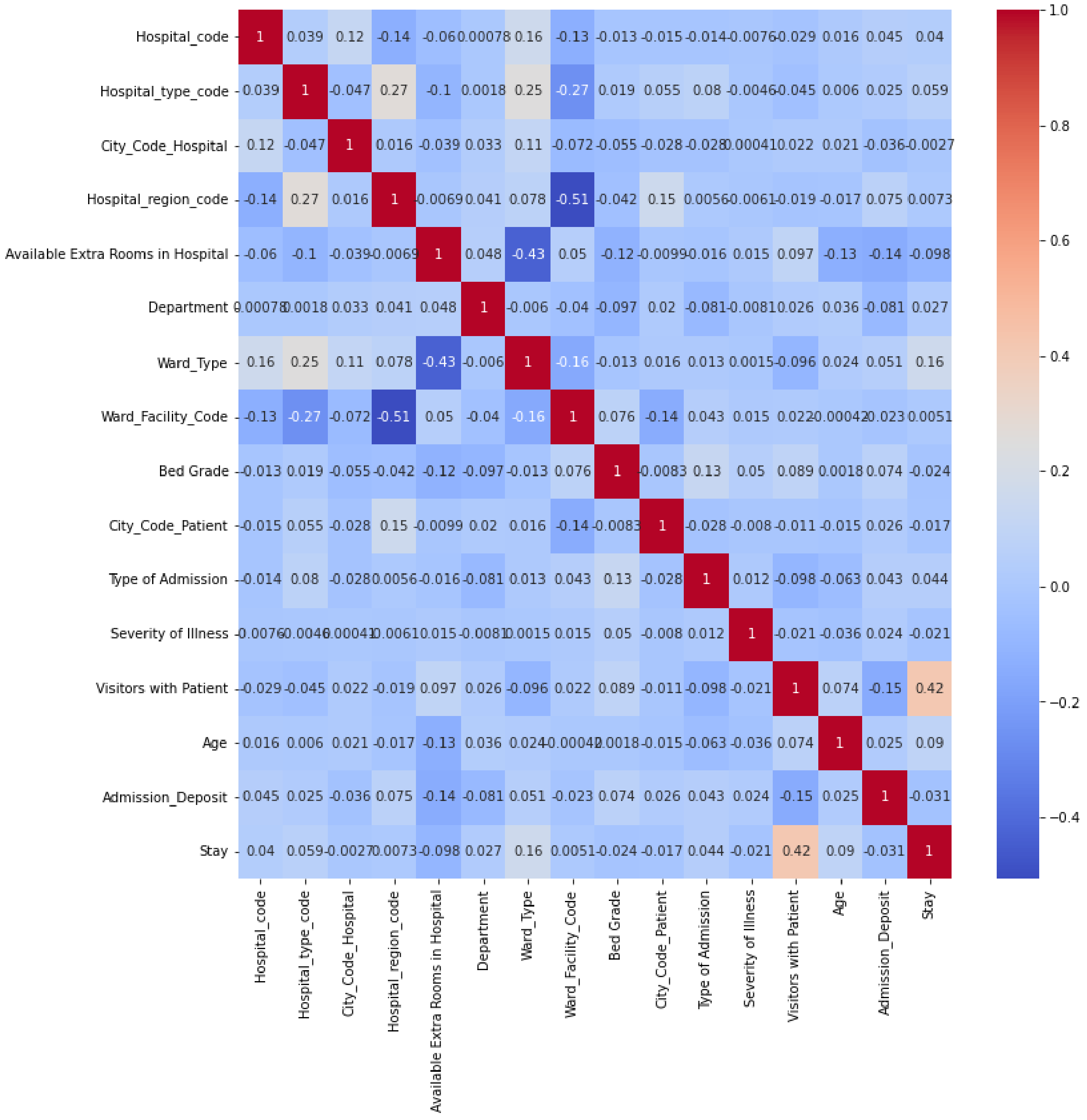


Figure 9: Correlation Heatmap of my dataset which shows relationship between different columns

4.3.4 Skewness

The degree to which a random variable's probability distribution deviates from the normal distribution is measured by its skewness. The probability distribution without any skewness is known as the normal distribution. Positively skewed probability distributions have their tails on the right, whereas negatively skewed distributions have their tails on the left (Sharma, 2020).

To calculate skewness, many formulae are available. Pearson's median skewness is one of the simplest. It benefits from the fact that in a skewed distribution, the mean and median are not equal. How many standard deviations separate the mean and median is shown by Pearson's median skewness. A Pearson's median skewness of exactly zero occurs in real observations infrequently. You can assume that your data has zero skew if its value is near to 0 (Turney, 2022).

$$\text{Pearson's median skewness} = 3 \times \frac{(\text{Mean} - \text{Median})}{\text{Standard deviation}}$$

Equation 8: Formula to calculate Pearson's median skewness

If your statistical process calls for a normal distribution yet your data are skewed, you typically have three options.

Avoid action. T tests, ANOVAs, and linear regressions are only a few examples of statistical tests that aren't very sensitive to skewed data. Perhaps it would be better to overlook the skew, especially if it is slight or moderate.

Change your model. Consider using a model that doesn't make the assumption of a normal distribution. Your data could respond well to non-parametric tests or generalised linear models.

Transform the variable. A skewed variable can also be altered to become less skewed. To "transform" anything is to give all of the observations of a variable the same function.

The below Figure 10 shows positive shew in our admission deposit column.

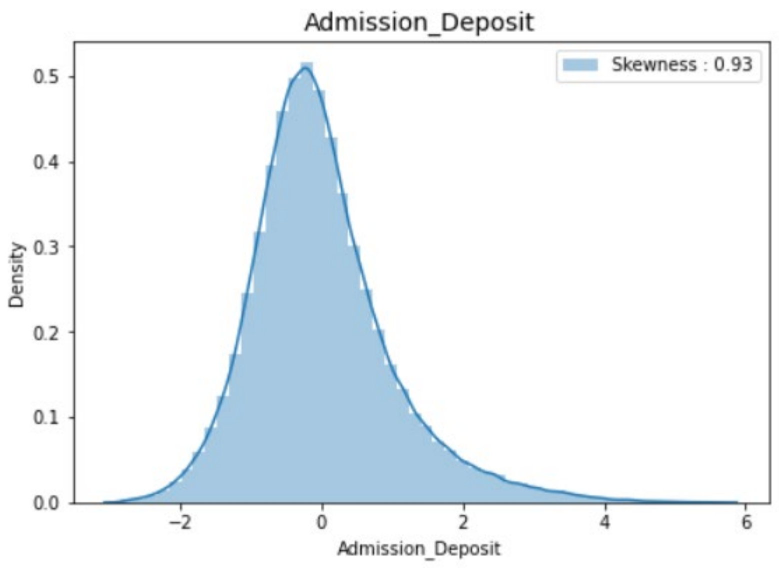


Figure 10: Positive skew shown in our admission_deposit column

5 Data Pre-processing

It is not possible to immediately run machine learning and analytics algorithms on raw data. Your data must first undergo pre-processing in order for machines to correctly "read" or comprehend it. Text, photos, video, and other types of unprocessed, real-world data are all incredibly disorganised. It frequently lacks regular, homogeneous data and frequently contains mistakes and inconsistencies in addition to being incomplete. Machines enjoy processing information that is well-organized. Therefore, a computer can easily calculate structured data, such as whole numbers and percentages. Unstructured data must first be cleaned and prepared before analysis, though, including text and pictures (Geisler Mesevage, 2021).

Potentially more crucial than the strongest algorithms are clean, pre-processed data, to the point where machine learning models trained on unclean data may even hinder the analysis. As a result, algorithms could provide "junk" results for you. Your data may be out of range or contain an erroneous feature depending on your data collection methods and sources. There may be blank fields or missing values in your dataset. It could even contain text data with typos, unnecessary symbols, URLs, etc. Your data will be significantly more accurate after being properly pre-processed and cleaned if you do this. We frequently hear about the value of "data-driven decision making," yet faulty data can still result in poor conclusions.

5.1 Data Features

The "features" that make up datasets can be used to describe them. By staydays, hospitalcode, bedgrade, etc., for example. Features, often referred to as attributes, are represented as columns in datasets. When pre-processing your data, it's critical to comprehend what "features" are because you'll need to select which ones to concentrate on based on your company objectives. Category and Numerical characteristics, which are the two main types of features used to characterise data.

Colors of a home, animal species, month of the year, True/False, positive, negative, neutral, etc. are examples of categorical values. Whole numbers, fractions, and percentages can all be used to represent numerical quantities. House prices, word counts in a paper, trip times, and other numerical aspects are examples of numerical features.

5.2 Data Outliers

Some outliers indicate genuine values resulting from population-wide natural variation. Other outliers could be caused by inaccurate data entry, broken equipment, or other measurement mistakes. You must exercise caution when dealing with outliers in data cleansing since they aren't necessarily indicative of stale or inaccurate data. An outlier's most likely cause will determine what action you should take with it.

Since they only reflect normal changes in your sample, true outliers should always be kept in your dataset. In variables with skewed distributions, when numerous data points are dispersed far from the mean in one direction, true outliers are also present. When you have a skewed distribution or a lot of outliers, it's crucial to use the right statistical tests or measurements. A boxplot may be used to identify outliers, as seen in the Figure 11.

There are several probable causes of outliers that don't represent genuine values, including measurement mistakes, data input or processing problems, and unrepresentative sample. This kind of outlier is troublesome since it is unreliable and can skew the findings of your study. In real life, it can be challenging to distinguish between various kinds of outliers. Outliers can be found using mathematics and statistical techniques (Zach, 2022).

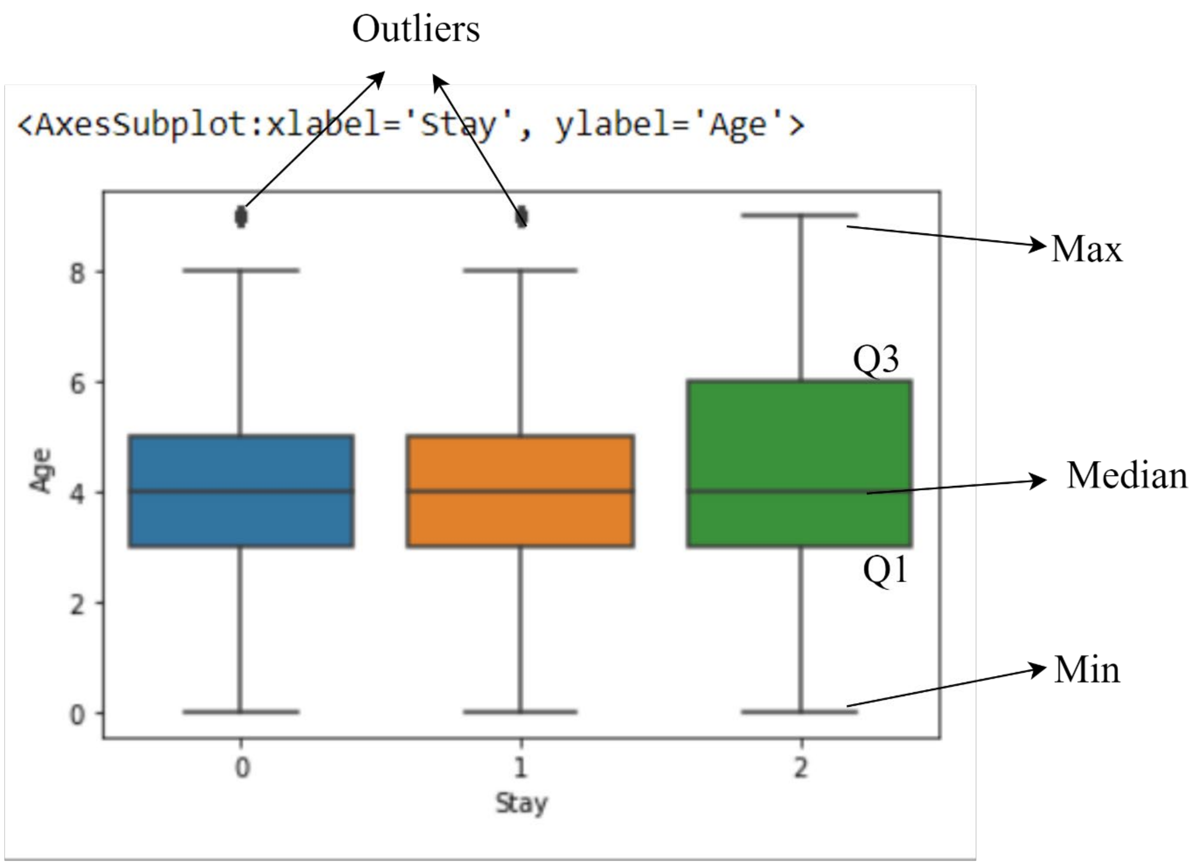


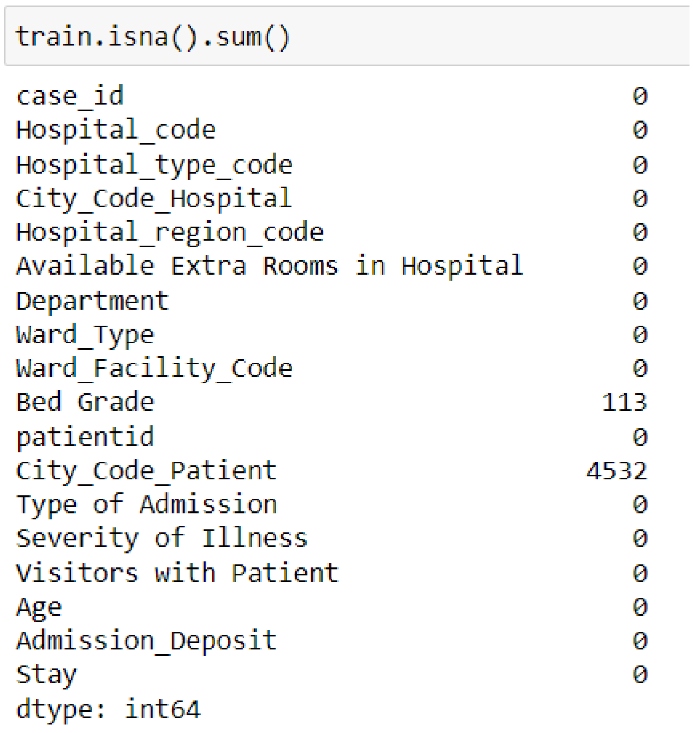
Figure 11: Outliers shown by a circle or dot which are outside the whiskers of the boxplot

5.3 *Missing Data*

When no information is given for one or more elements, a full unit, or both, this is known as missing data. Many datasets in DataFrame occasionally come with missing data, either because the data was never gathered or because it was present but was not captured. Pandas has two values for missing data: None and NaN. To indicate missing or null values, Pandas handle None and NaN similarly to each other. There are various helpful functions for identifying, eliminating, and replacing null values that might help this standard (*Working with Missing Data in Pandas*, 2019).

isna(): The dataframe.isna() method of Pandas is used to find missing values. If the values are NA, a boolean object is returned. NA values like None and numpy.NaN are translated to True values. False values are assigned to everything else.

The amount of missing values in the dataset is depicted in the picture Figure 12. After using the isna function, I utilise the sum function to tell us the total number of true values.



```
train.isna().sum()
```

```
case_id                               0
Hospital_code                         0
Hospital_type_code                    0
City_Code_Hospital                    0
Hospital_region_code                  0
Available Extra Rooms in Hospital     0
Department                            0
Ward_Type                             0
Ward_Facility_Code                    0
Bed Grade                           113
patientid                             0
City_Code_Patient                  4532
Type of Admission                     0
Severity of Illness                   0
Visitors with Patient                 0
Age                                   0
Admission_Deposit                     0
Stay                                  0
dtype: int64
```

Figure 12: Total number of missing values in the respective column

fillna(): The pandas fillna() function manages and allows the user to replace NaN numbers with other values, similar to how the pandas dropna() method maintains and removes Null values from a data frame.

5.4 *Scaling*

This implies that you're altering your data to make it compatible with a certain scale, such as 0-100 or 0-1. When employing techniques based on estimates of how far away data points are, such as support vector machines (SVM) or k-nearest neighbours, you should scale the data (KNN). These algorithms assign the same emphasis to changes of "1" in any numerical characteristic. By scaling your variables, you may make it easier to compare several variables side by side.

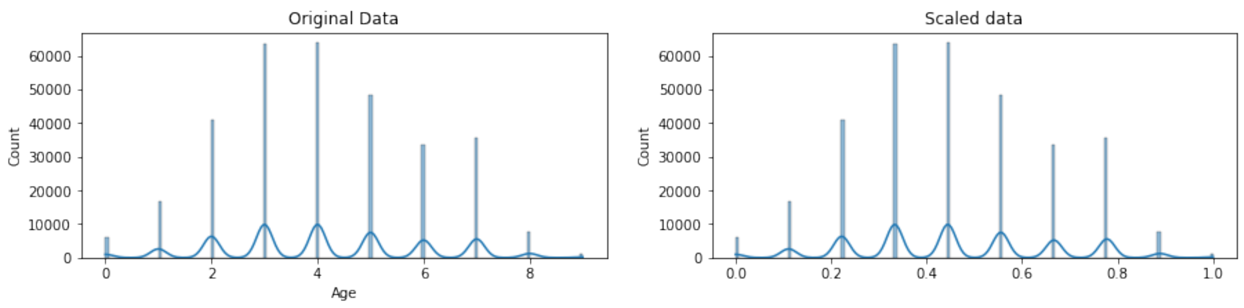


Figure 13: The shape of the data doesn't change, but that instead of ranging from 0 to 8ish, it now ranges from 0 to 1

5.5 Normalization

Scaling only modifies the data's range. A more drastic change is normalisation. Normalization is the process of transforming your observations into something that can be compared to a normal distribution. If you want to use a machine learning or statistics approach that presumes your data is normally distributed, you should generally normalise your data. Examples of this are Gaussian naive Bayes and linear discriminant analysis (LDA).

Normalization is also known as Min-Max scaling. Here's the formula for normalization (A. Bhandari, 2020):

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Equation 9: Here, Xmax and Xmin are the maximum and minimum values of the feature respectively

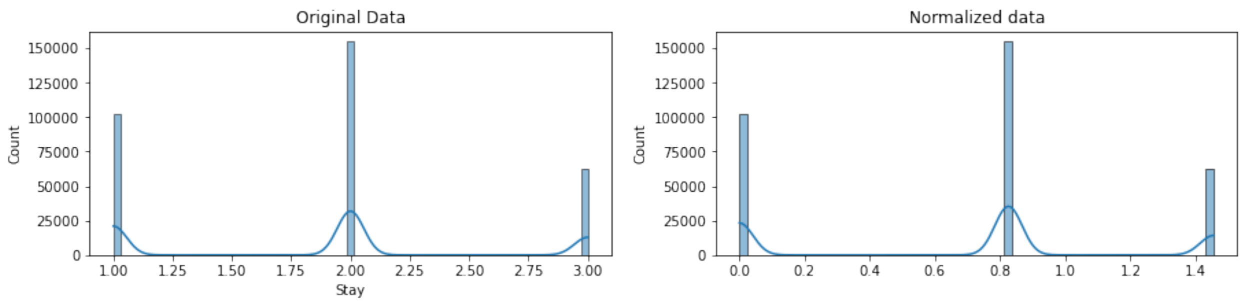


Figure 14: Notice that the shape of our data has changed pushing slightly to the right

6 Experimental Results

6.1 K Fold Cross Validation

K-fold The dataset is divided into a K number of folds during cross-validation, which assesses the model's performance when faced with fresh data. K is the number of groups into which the data sample is divided. For instance, we may refer to this as a 5-fold cross-validation if the k-value is 5 as show in Figure 15. At some stage throughout the technique, each fold serves as a testing set.

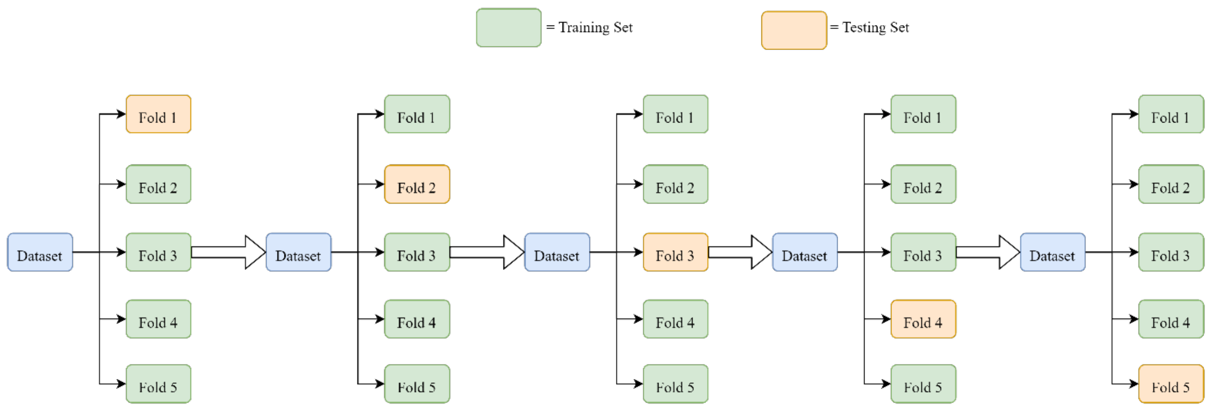


Figure 15: If the k-value is 5 we get this type of 5-fold cross-validation splits in the dataset

All components of the data can be utilised as testing data when K-fold cross validation is applied. This enables us to evaluate the performance of our model by using all of the data from our dataset for both training and testing. We would have 10 outcomes to utilise in our evaluation of the model's performance, for instance, if we set our k value at 10. Having 10 separate accuracy results where all of the data was used in the test phase is always going to be better and more trustworthy than utilising one accuracy result that was obtained by a train-test split, if accuracy were to be our primary metric (Arya, 2022).

A number of tests were carried out to evaluate the performance of the five generated prediction models indicated in 3.3 in order to meet the objectives of this research project. The K-fold numbers employed for the project's models were 2, 3, and 5. Accuracy, precision, recall, and f-score were used to calculate the performances of several models with various K-fold values. In the sections below, the created models and their performance are discussed.

6.2 Model 1: Logistic Regression

Logistic regression was used to build the first model. The results of this model on the validation set for classifying the number of days to discharge the patient using various K-fold values are shown in the table below. Using the LR classifier, the 2-fold outperformed the other K-folds in terms of accuracy performance with 54.51% accuracy as seen in table below. As we raised the k-value, we saw that the accuracy was declining.

Table 2: The results of predicting the Length of Stay of patients in hospital for Model 1

K-fold	*Accuracy*	*Precision*	*Recall*	*Micro F1*	*Macro F1*
2	0.5451	0.5726	0.5173	0.5451	0.4986
3	0.5366	0.5662	0.5184	0.5366	0.5004
5	0.5279	0.5562	0.5154	0.5279	0.4974

6.3 Model 2: Random Forest

Random forest was used to build the second model. The results of this model on the validation set for classifying the number of days to discharge the patient using various K-fold values are shown in the table below. Using the RF classifier, the 2-fold outperformed the other K-folds in terms of accuracy performance with 58.24% accuracy as seen in table below.

Table 3: The results of predicting the Length of Stay of patients in hospital for Model 2

K-fold	*Accuracy*	*Precision*	*Recall*	*Micro F1*	*Macro F1*
2	0.5824	0.5911	0.5820	0.5824	0.5821
3	0.5762	0.5861	0.5798	0.5762	0.5791
5	0.5698	0.5776	0.5763	0.5698	0.5741

6.4 Model 3: K-Nearest Neighbours

K-Nearest Neighbours was used to build the third model. The results of this model on the validation set for classifying the number of days to discharge the patient using various K-fold values are shown in the table below. Using the KNN classifier, the 2-fold outperformed the other K-folds in terms of accuracy performance with 54.29% accuracy as seen in table below.

Table 4: The results of predicting the Length of Stay of patients in hospital for Model 3

K-fold	*Accuracy*	*Precision*	*Recall*	*Micro F1*	*Macro F1*
2	0.5429	0.5631	0.5154	0.5429	0.5311
3	0.5411	0.5626	0.5170	0.5411	0.5323
5	0.5387	0.5585	0.5154	0.5387	0.5306

6.5 Model 4: Gradient Boosting

Gradient Boosting was used to build the fourth model. The results of this model on the validation set for classifying the number of days to discharge the patient using various K-fold values are shown in the table below. Using the LR classifier, the 2-fold outperformed the other K-folds in terms of accuracy performance with 58.51% accuracy as seen in table below. Out of the five models that were provided, this one performed the slowest.

Table 5: The results of predicting the Length of Stay of patients in hospital for Model 4

K-fold	*Accuracy*	*Precision*	*Recall*	*Micro F1*	*Macro F1*
2	0.5851	0.5972	0.5872	0.5851	0.5746
3	0.5779	0.5896	0.5875	0.5779	0.5743
5	0.5727	0.5805	0.5848	0.5727	0.5728

6.6 Model 5: Extreme Gradient Boosting

Extreme Gradient Boosting was used to build the fifth model and it had the greatest accurate. The results of this model on the validation set for classifying the number of days to discharge the patient using various K-fold values are shown in the table below. Using the LR classifier, the 2-fold outperformed the other K-folds in terms of accuracy performance with 60.91% accuracy as seen in table below.

Table 6: The results of predicting the Length of Stay of patients in hospital for Model 5

K-fold	*Accuracy*	*Precision*	*Recall*	*Micro F1*	*Macro F1*
2	0.6091	0.6217	0.6057	0.6091	0.6069
3	0.6023	0.6149	0.6036	0.6023	0.6039
5	0.5979	0.6081	0.6004	0.5979	0.6006

6.7 Hyperparameter Tuning

Hyperparameters in machine learning algorithms let you adjust the algorithm's behaviour to fit your particular dataset. Hyperparameters, in contrary to parameters, are chosen by the model's practitioner. It is frequently used to utilise random or grid search techniques for different hyperparameter values since it might be difficult to determine what values to use for the hyperparameters of a particular algorithm on a specific dataset. The tuning procedure takes longer the more hyperparameters of an algorithm you need to adjust. As a result, choosing a minimal selection of model hyperparameters is preferable before doing a search or tuning (Brownlee, 2019).

Not every model hyperparameter is crucial. To acquire a decent result quickly as a machine learning practitioner, you must be aware of the hyperparameters to concentrate on. We saw gains in accuracy in the machine learning algorithms after hyperparameter adjustment. Some hyperparameters only work with certain types of datasets. Therefore, not all hyperparameters are beneficial. When using GridSearchCV, if the dataset is enormous, as it was in our instance, it takes a long time to run through all of the supplied hyperparameters. Therefore, we employed RandomizedSearchCV, which follows a similar workflow to GridSearchCV. The RandomizedSearchCV uses arbitrary selections of hyperparameters to run, which is how it differs from the other two. As a result of the randomised format, it is also quicker than GridSearchCV. So, in our project, we've used RandomizedSearchCV.

Our best score using hyperparameterized logistic regression was 0.5610. The hyperparameters that RandomizedSearchCV selected were “solver”: “newton-cg”, “penalty”: “l2”, and “C”: 0.1. newton-cg is good for multiclass problems.

Using hyperparameterized Random Forest, we achieved best score of 0.5964. RandomizedSearchCV chose the hyperparameters “n_estimators”: 1000 and “max_features”: “log2”. The more number of trees the better, so n_estimators 1000 is preferred over 10 and 100.

The best score we could get with hyperparameterized K-Nearest Neighbors was 0.5707. The hyperparameters “weights”: “uniform”, “n_neighbors”: 13, and “metric”: “euclidean” were selected via RandomizedSearchCV. Number of neighbours as 13 was better for the algorithm. By uniform weights all points in each neighbourhood are weighted equally.

The greatest score we could achieve with the hyperparameterized Gradient Boosting Classifier was 0.6072. The hyperparameters "subsample": 1.0, "n_estimators": 1000, "max_depth": 9, and "learning_rate": 0.1 were selected using RandomizedSearchCV. It took a very long time to execute the gradient boosting classifier with hyperparameter adjustment. Of the five algorithms, it was the slowest.

With hyperparameterized XGBoost, we were able to get best score of 0.6235. Using RandomizedSearchCV, the hyperparameters “subsample”: 0.8, “objective”: “binary:logistic”, “min_child_weight”: 5, “max_depth”: 3, “gamma”: 0.5, and “colsample_bytree”: 0.8 were chosen. min_child_weight defines the minimum sum of weights of all observations required in a child. The maximum depth (max_depth) of a tree, is same as in GBM. Gamma specifies the minimum loss reduction required to make a split. It makes the algorithm conservative. subsample denotes the fraction of observations to be randomly samples for each tree. colsample_bytree denotes the fraction of columns to be randomly samples for each tree. objective defines the loss function to be minimized. Most of the hyperparameters helps with data overfitting problems.

7 Conclusions

As they provide medical services with excellent tools to anticipate, monitor, track, and control the pandemic, artificial intelligence technologies have been widely used in the fight against the COVID-19 pandemic. The goal of this effort was to offer a framework for managing medical resources and streamlining services for COVID-19 patients. The research described in this study intended to create a prediction model to forecast the number of days COVID-19 patients will spend in the hospital. The five classifiers used—LR, RF, KNN, GB, and XGBoost—are known to work effectively for the task and were used as prediction models. With XGBoost, the generated model's accuracy was 62.35%, making it a useful tool for hospitals and healthcare professionals. Five prediction models were developed for this goal utilising various ML classifiers with k-value tweaking.

7.1 Achievements

The findings indicate that each model handled the dataset differently. In order to choose the model with the best performance, results were compared as seen in Table 7. Comparing the results of all the models, we can see that model 5, which used Extreme Gradient Boosting, had the best accuracy of 60.91% without hyperparameter tuning in predicting the range of days till hospital discharge. As can be observed, XGBoost achieves the greatest results across all performance metrics. For accuracy, precision, recall, Micro F1-score, and Macro F1-score it received scores of 60.91%, 62.17%, 60.57%, 60.91% and 60.69%, respectively. The performance of the remaining classifiers was marginally lower.

Table 7: Comparing all results of all the five models without hyperparameter tuning

Models	*Accuracy*	*Precision*	*Recall*	*Micro F1*	*Macro F1*
Logistic Regression	0.5451	0.5726	0.5173	0.5451	0.4986
Random Forest	0.5824	0.5911	0.5820	0.5824	0.5821
K-Nearest Neighbours	0.5429	0.5631	0.5154	0.5429	0.5311
Gradient Boosting	0.5851	0.5972	0.5872	0.5851	0.5746
Extreme Gradient Boosting	0.6091	0.6217	0.6057	0.6091	0.6069

All five machine learning models underwent the hyperparameter tweaking procedure to increase their accuracy. The algorithms with the tweaking executed more slowly than expected. The results of combining RandomizedSearchCV with hyperparameter algorithm adjustment are displayed below.

Table 8: : Comparing best score of all the five models with hyperparameter tuning

Models	*Best Score*
Logistic Regression	0.5610
Random Forest	0.5964
K-Nearest Neighbours	0.5707
Gradient Boosting	0.6072
Extreme Gradient Boosting	0.6235

7.2 Future Work

The established model and the data gathered will be used in the future to estimate and forecast COVID-19 patient demand for various needs, allowing for improved resource management. A separate dataset gathered from other parts of the world can also be utilised with the constructed model.

The past and future of the models need to be much better understood. I'm particularly interested in finding out how to better properly estimate large datasets on a simple system. Because the amount of data in our daily lives will only increase, we must process it without the use of supercomputers.

8 Student Reflections

I am pleased with the results of this project report. A little assistance is thought to go a long way because the epidemic is affecting so many people worldwide. Given my limitations, I've done some decent things. Home computers lack the processing capacity of supercomputers used by major corporations. Even running the hyperparameters for my machine learning models was challenging for me. Running a single process with parameter adjustment takes a lengthy time. When GridSearchCV took even longer, I switched to RandomSearchCV because it was quicker.

The intellectual or theoretical portion of any given new learning used to make me eager. However, while I worked on the project, I discovered several procedures that I would not have learned about if I hadn't been studying machine learning in practise. I am aware that the majority of machine learning is applied; the theoretical side of machine learning only accounts for a small fraction. If it weren't for this research, I wouldn't have learned how much computing power machine learning requires.

Focusing exercises are a tactic that has significantly improved my ability to study. When I got stuck on a problem, I used to lose focus. I used to put off doing things whenever I had the chance, whether it was after finishing a task or while I was having trouble with a current task. I found the pomodoro technique to be quite helpful if I was having attention issues. I was able to put a lot more effort into my project and find solutions to the problems I was having by better focusing.

If I could go back in time and change something, I would spend more time investigating the machine learning methods I employed. The models have a lot more than I realised. There was far more knowledge about the models' past and future. I really want to learn how to forecast massive datasets more accurately on a basic machine. We need to process the data without the use of super computers because the amount of data in our everyday lives will continue to grow. Some variants, like the XGBoost, perform far better in terms of accuracy and speed than its predecessor, the Gradient Boosting.

9 Project Management

9.1 Project Schedule

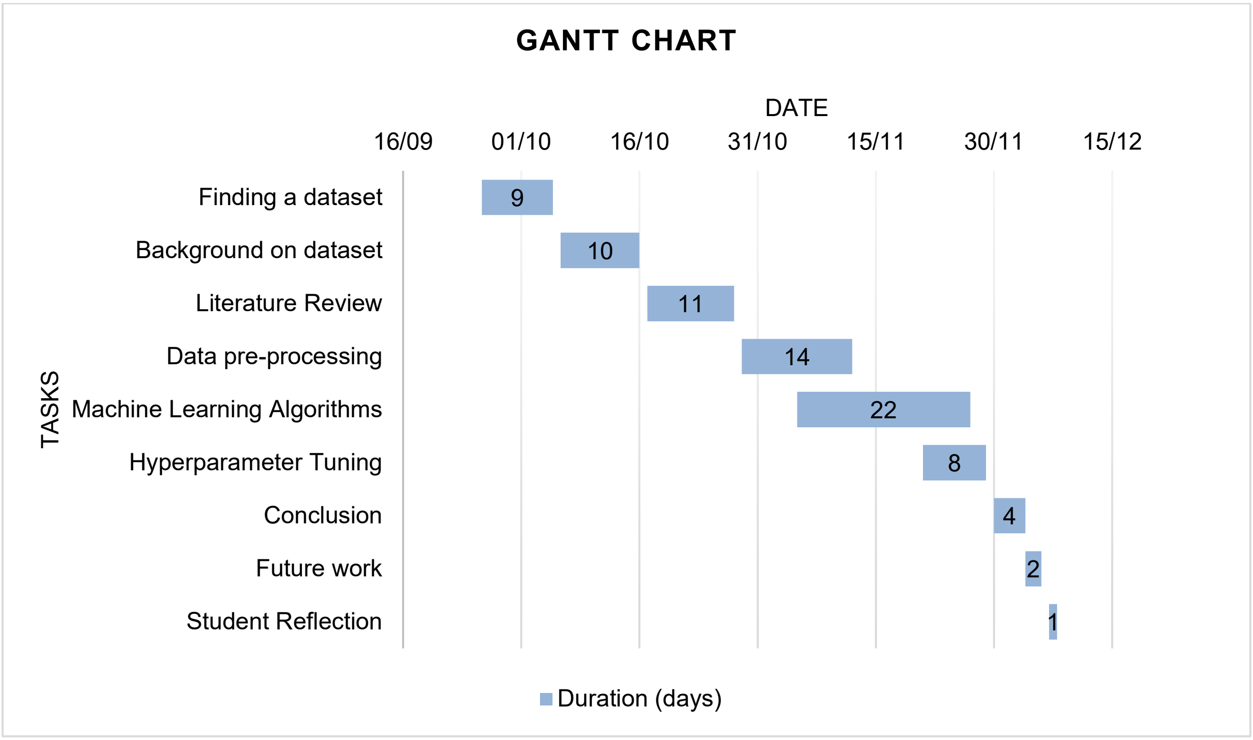


Figure 16: Gantt chart showing my progress

The aforementioned graph above displays the daily development of my project report. Although there were few adjustments here and there, the project did not go exactly as expected. Even though they weren't part of the plan, all the modifications were progressive. Choosing a dataset took more longer than I had planned because I had trouble finding one that I liked. I am pleased with the dataset I chose because it had a lot of attributes that were helpful in making the prediction.

9.2 Quality Management

To validate the accuracy of my predictions, I cross-validated the machine learning models using various k-values. With the use of hyperparameter tweaking of algorithms and RandomizedSearchCV to apply the parameters, I was able to further enhance my prediction. To see if there was anything I could do to improve the quality of the forecasts, I ran the models multiple times.

9.3 Social, Legal, Ethical and Professional Considerations

Among the dataset's considerations were Social, Legal, Ethical, and Professional Considerations. I took notice that my dataset did not include any personal data about any specific individuals. Only patient data with patient id and case id as the primary information was available to me. After I presented the dataset to the ethical approval committee, the research was deemed to be low risk.

Bibliography and References

Alabbad, D. A., Almuhaideb, A. M., Alsunaidi, S. J., Alqudaihi, K. S., Alamoudi, F. A., Alhobaishi, M. K., Alaqeel, N. A., & Alshahrani, M. S. (2022). Machine learning model for predicting the length of stay in the intensive care unit for Covid-19 patients in the eastern province of Saudi Arabia. *Informatics in Medicine Unlocked*, *30*(2352-9148), 100937. https://doi.org/10.1016/j.imu.2022.100937

Arya, N. (2022, July 11). *Why Use k-fold Cross Validation?* KDnuggets. https://www.kdnuggets.com/2022/07/kfold-cross-validation.html

Bhandari, A. (2020, April 3). *Feature Scaling | Standardization Vs Normalization*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/

Bhandari, P. (2020, June 5). *Data Collection | A Step-by-Step Guide with Methods and Examples*. Scribbr. https://www.scribbr.com/methodology/data-collection/

Brownlee, J. (2018, May 21). *A Gentle Introduction to k-fold Cross-Validation*. Machine Learning Mastery. https://machinelearningmastery.com/k-fold-cross-validation/

Brownlee, J. (2019, December 12). *Tune Hyperparameters for Classification Machine Learning Algorithms*. Machine Learning Mastery. https://machinelearningmastery.com/hyperparameters-for-classification-machine-learning-algorithms/

Clarke, M. (2021, May 29). *How to create a classification model using XGBoost in Python*. Practicaldatascience.co.uk. https://practicaldatascience.co.uk/machine-learning/how-to-create-a-classification-model-using-xgboost

Cleveland Clinic. (2020, November 12). *Coronavirus Disease (COVID-19): What Is It, Symptoms, Causes & Prevention*. Cleveland Clinic. https://my.clevelandclinic.org/health/diseases/21214-coronavirus-covid-19

Dan, T., Li, Y., Zhu, Z., Chen, X., Quan, W., Hu, Y., Tao, G., Zhu, L., Zhu, J., Jin, Y., Li, L., Liang, C., Wen, H., & Cai, H. (2020, December 1). *Machine Learning to Predict ICU Admission, ICU Mortality and Survivors' Length of Stay among COVID-19 Patients: Toward Optimal Allocation of ICU Resources*. IEEE Xplore. https://doi.org/10.1109/BIBM49941.2020.9313292

Draw.io. (2022). *Flowchart Maker & Online Diagram Software*. App.diagrams.net. https://app.diagrams.net/

Etu, E.-E., Monplaisir, L., Arslanturk, S., Masoud, S., Aguwa, C., Markevych, I., & Miller, J. (2022). Prediction of Length of Stay in the Emergency Department for COVID-

19 Patients: A Machine Learning Approach. *IEEE Access*, *10*(42243-42251), 42243–42251. https://doi.org/10.1109/ACCESS.2022.3168045

Excel Gantt chart tutorial +Free Template + Export to PPT. (n.d.). Office Timeline. Retrieved December 8, 2022, from https://www.officetimeline.com/gantt-chart/how-to-make/excel

Ferreira, Anselmo. (2019). How to Write a Machine Learning Paper for (not so) Dummies.

Foley, B. (2018, March 31). *Benefits of Using Secondary Data Analysis for Your Research*. Alchemer. https://www.alchemer.com/resources/blog/secondary-data-analysis/

Geisler Mesevage, T. (2021, May 24). *What Is Data Preprocessing & What Are The Steps Involved?* MonkeyLearn Blog. https://monkeylearn.com/blog/data-preprocessing/

Henzi, A., Kleger, G.-R., Hilty, M. P., Wendel Garcia, P. D., & Ziegel, J. F. (2021). Probabilistic analysis of COVID-19 patients' individual length of stay in Swiss intensive care units. *PLOS ONE*, *16*(2), e0247265. https://doi.org/10.1371/journal.pone.0247265

Hessner, S. (2018, July 19). *Why are precision, recall and F1 score equal when using micro averaging in a multi-class problem? – Simon's blog*. Simon's Blog. https://simonhessner.de/why-are-precision-recall-and-f1-score-equal-when-using-micro-averaging-in-a-multi-class-problem/

Hillier, W. (2022, May 24). *What is Secondary Data? [Examples, Sources & Advantages]*. Careerfoundry.com. https://careerfoundry.com/en/blog/data-analytics/what-is-secondary-data/

Hoare, J. (2017, June 5). *Gradient Boosting Explained - The Coolest Kid on The Machine Learning Block*. Displayr. https://www.displayr.com/gradient-boosting-the-coolest-kid-on-the-machine-learning-block/

Hong, Y., Wu, X., Qu, J., Gao, Y., Chen, H., & Zhang, Z. (2020). Clinical characteristics of Coronavirus Disease 2019 and development of a prediction model for prolonged hospital length of stay. *Annals of Translational Medicine*, *8*(7), 443–443. https://doi.org/10.21037/atm.2020.03.147

Huilgol, P. (2019, August 24). *Accuracy vs. F1-Score*. Medium. https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2

JavaTpoint. (n.d.). *K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint*. Www.javatpoint.com. Retrieved November 22, 2022, from https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Kanstrén, T. (2020, October 31). *A Look at Precision, Recall, and F1-Score*. Medium. https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec

Kohli, S. (2019, November 18). *Understanding a Classification Report For Your Machine Learning Model*. Medium. https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397

Kumar, A. (2022, April 16). *Correlation Concepts, Matrix & Heatmap using Seaborn*. Data Analytics. https://vitalflux.com/correlation-heatmap-with-seaborn-pandas/#:~:text=with%20each%20other.-

Leung, K. (2022, January 9). *Micro, Macro & Weighted Averages of F1 Score, Clearly Explained*. Medium. https://towardsdatascience.com/micro-macro-weighted-averages-of-f1-score-clearly-explained-b603420b292f

López-Cheda, A., Jácome, M.-A., Cao, R., & De Salazar, P. M. (2021). Estimating lengths-of-stay of hospitalised COVID-19 patients using a non-parametric model: a case study in Galicia (Spain). *Epidemiology and Infection*, *149*(E102). https://doi.org/10.1017/s0950268821000959

Mohajon, J. (2020, September 9). *Confusion Matrix for Your Multi-Class Machine Learning Model*. Medium. https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826

Morkovich, E. (2011, July 14). *Project Background*. My Management Guide. https://mymanagementguide.com/project-background/

Nvidia. (n.d.). *What is XGBoost?* NVIDIA Data Science Glossary. Retrieved November 24, 2022, from https://www.nvidia.com/en-us/glossary/data-science/xgboost/

Olivato, M., Rossetti, N., Gerevini, A. E., Chiari, M., Putelli, L., & Serina, I. (2022). Machine Learning Models for Predicting Short-Long Length of Stay of COVID-19 Patients. *Procedia Computer Science*, *207*(1877-0509), 1232–1241. https://doi.org/10.1016/j.procs.2022.09.179

Scribbr - Your path to academic success. (2019). Scribbr. https://www.scribbr.com/

Sharma, A. (2020, July 5). *What is Skewness in Statistics? | Statistics for Data Science*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/07/what-is-skewness-statistics/

So, R. (2022, September 25). *Logistic Regression — Classification, Mathematics, and Python*. Medium. https://medium.com/@rvs6736/logistic-regression-classification-mathematics-and-python-dd8c827ef568

Stack Overflow. (2022). *Stack Overflow - Where Developers Learn, Share, & Build Careers*. Stack Overflow. https://stackoverflow.com/

Student self assessment and reflection / Examples and templates / Reporting to parents & whānau / Home - Assessment. (2019). Tki.org.nz. https://assessment.tki.org.nz/Reporting-to-parents-whanau/Examples-and-templates/Student-self-assessment-and-reflection

Suresh, A. (2020, November 20). *What is a confusion matrix?* Medium. https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5

Tam, A. (2022, February 22). *Easier Experimenting in Python*. MachineLearningMastery.com. https://machinelearningmastery.com/easier-experimenting-in-python/

Turney, S. (2022, May 10). *Skewness | Definition, Examples & Formula*. Scribbr. https://www.scribbr.com/statistics/skewness/

What is a Random Forest? (n.d.). NVIDIA Data Science Glossary. Retrieved November 21, 2022, from https://www.nvidia.com/en-us/glossary/data-science/random-forest/

What Is Deep Learning? | How It Works, Techniques & Applications. (2019). Mathworks.com. https://uk.mathworks.com/discovery/deep-learning.html

What is Machine Learning? (n.d.). NVIDIA Data Science Glossary. Retrieved November 18, 2022, from https://www.nvidia.com/en-us/glossary/data-science/machine-learning/

Wikipedia Contributors. (2019a, April 12). *Logistic regression*. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Logistic_regression

Wikipedia Contributors. (2019b, September 9). *XGBoost*. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Xgboost

Working with Missing Data in Pandas. (2019, January 3). GeeksforGeeks. https://www.geeksforgeeks.org/working-with-missing-data-in-pandas/

Zach. (2022, September 1). *How to Read a Box Plot with Outliers (With Example)*. Statology. https://www.statology.org/how-to-read-box-plot-with-outliers/

Appendix A – Project Code

This github link will take you to a Jupyter Notebook file containing all of my code: https://github.com/viratbamaniya/MScDataScienceProject/blob/main/Project%20Code.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[ ]:



import pandas as pd
import numpy as np
import statistics
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import precision_score, recall_score, f1_score

from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler

from sklearn import metrics
from sklearn.model_selection import KFold
from sklearn.model_selection import RandomizedSearchCV

from mlxtend.preprocessing import minmax_scaling


# In[ ]:


train = pd.read_csv("C:/Users/virat/Desktop/7150/train_data.csv", index_col=0)
test = pd.read_csv("C:/Users/virat/Desktop/7150/test_data.csv", index_col=0)


# In[ ]:


# Display dataset
train


# In[ ]:


# Columns with their dataset
train.info()


# In[ ]:


# Describes the data
train.describe()


# In[ ]:
```

```
# Get the total number of missing values
train.isna().sum()


# In[ ]:


# If the .csv file is opened in excel11-20 is converted to Nov-20
train = train.replace('Nov-20', '11-20')
test = test.replace('Nov-20', '11-20')


# In[ ]:


# Fill the missing values using statistics module
train['Bed Grade'].fillna(statistics.mode(train['Bed Grade']),inplace=True)
train['City_Code_Patient'].fillna(statistics.mode(train['City_Code_Patient']),inplace=True)


# In[ ]:


# Columns are of no use so we are removing them
train.drop(['case_id', 'patientid'], axis=1, inplace=True)


# In[ ]:


# We can't predict 11 classes so we are merging them into 3 classes to predict
train['Stay'].replace('More than 100 Days', '>100', inplace=True)
train['Stay']= train['Stay'].replace(
    {'0-10':0, '11-20':0, '21-30':1, '31-40':1, '41-50':1, '51-60':2,'61-70':2,'71-80':2,'81-
90':2,'91-100':2,'>100':2})


# In[ ]:


# Dividing columns into categorical and numerical columns
cat_cols=[]
num_cols=[]

for col in train.columns:
    if train[col].dtypes=='object':
        cat_cols.append(col)
    else:
        num_cols.append(col)


# In[ ]:


# Plot of count of values into each variables
i=1
plt.figure(figsize=(15,20))
for col in cat_cols:
    plt.subplot(5,2,i)
    sns.countplot(train[col])
    i=i+1
plt.show()


# In[ ]:


# Plot of density of values into each variables
```

```
i=1
plt.figure(figsize=(15,20))
for col in num_cols:
    plt.subplot(4,2,i)
    sns.distplot(train[col])
    i=i+1
plt.show()


# In[ ]:


# Changing some columns from numerical type to categorical as they fit better
cat_cols.append('Bed Grade')
cat_cols.append('City_Code_Hospital')
cat_cols.append('City_Code_Patient')

num_cols.remove('Bed Grade')
num_cols.remove('City_Code_Hospital')
num_cols.remove('City_Code_Patient')


# In[ ]:


# Encoding categorical columns
le= LabelEncoder()
for col in cat_cols:
    train[col]= le.fit_transform(train[col])


# In[ ]:


# Standardizing numerical columns
```

```
ss= StandardScaler()
train[num_cols]= ss.fit_transform(train[num_cols].values)


# In[ ]:


# Correlation heatmap of dataset variables
plt.figure(figsize=(12,12))
sns.heatmap(train.corr(), annot=True, cmap='coolwarm')


# In[ ]:


# Plotting skewness of these 3 columns
num_data = train[['Available Extra Rooms in Hospital', 'Bed Grade',
'Admission_Deposit']]
fig, ax =plt.subplots(2,2, figsize=(14,10))
fig.tight_layout(pad=5.0)

for ax, n in zip(ax.flatten(), num_data.columns.tolist()):
   sns.distplot(ax=ax, a=num_data[n].dropna(), label="Skewness :
%.2f"%(num_data[n].skew()))
   ax.set_title(n, fontsize = 14)
   ax.legend(loc = 'best')


# In[ ]:


# Removing predicting columns from training file
y= train['Stay']
X= train.drop('Stay', axis=1)
```

```
# In[ ]:



# Box-plotting to see outliers in dataset
sns.boxplot(x = 'Stay', y = 'Age', data = train)



# In[ ]:



# Plotting scaling of dataset
scaled_data = minmax_scaling(train['Age'], columns=[0])

fig, ax = plt.subplots(1, 2, figsize=(15, 3))
sns.histplot(train['Age'], ax=ax[0], kde=True, legend=False)
ax[0].set_title("Original Data")
sns.histplot(scaled_data, ax=ax[1], kde=True, legend=False)
ax[1].set_title("Scaled data")
plt.show()



# In[ ]:



# Plotting normalization of dataset
normalized_data = stats.boxcox(train['Stay'])

fig, ax=plt.subplots(1, 2, figsize=(15, 3))
sns.histplot(train['Stay'], ax=ax[0], kde=True, legend=False)
ax[0].set_title("Original Data")
sns.histplot(normalized_data[0], ax=ax[1], kde=True, legend=False)
ax[1].set_title("Normalized data")
plt.show()
```

```
# In[ ]:



# Changing k-values to see the difference
kf = KFold(n_splits = 2)
# kf = KFold(n_splits = 3)
# kf = KFold(n_splits = 5)

for train_index, test_index in kf.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]



# In[ ]:



# Running Logistic Regression
lr = LogisticRegression(max_iter=1000)
lr.fit(X_train,y_train)
y_pred = lr.predict(X_test)

print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))

confusion = confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion)
cm_display.plot()
plt.show()

print(precision_score(y_test,y_pred, average='macro'))
print(recall_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='micro'))
```

```
# In[ ]:


# Running Random Forest
rf = RandomForestClassifier()
rf.fit(X_train,y_train)
y_pred = rf.predict(X_test)

print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))

confusion = confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion)
cm_display.plot()
plt.show()

print(precision_score(y_test,y_pred, average='macro'))
print(recall_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='micro'))


# In[ ]:


# Running K-Nearest Neighbours
knn = KNeighborsClassifier()
knn.fit(X_train,y_train)
y_pred = knn.predict(X_test)

print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))

confusion = confusion_matrix(y_test, y_pred)
```

```
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion)
cm_display.plot()
plt.show()

print(precision_score(y_test,y_pred, average='macro'))
print(recall_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='micro'))


# In[ ]:


# Running Gradient Boosting, this is the slowest
gb = GradientBoostingClassifier()
gb.fit(X_train,y_train)
y_pred = gb.predict(X_test)

print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))

confusion = confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion)
cm_display.plot()
plt.show()

print(precision_score(y_test,y_pred, average='macro'))
print(recall_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='micro'))


# In[ ]:
```

```
# Running Extreme Gradient Boosting
xgb = XGBClassifier()
xgb.fit(X_train,y_train)
y_pred = xgb.predict(X_test)

print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))

confusion = confusion_matrix(y_test, y_pred)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion)
cm_display.plot()
plt.show()

print(precision_score(y_test,y_pred, average='macro'))
print(recall_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='macro'))
print(f1_score(y_test,y_pred, average='micro'))


# In[ ]:


# Splitting of dataset normally
X_train, X_test, y_train,y_test= train_test_split(X,y,test_size= 0.2, stratify=y,
random_state=42)


# In[ ]:


# Running all the five algorithms
modells = [LogisticRegression(max_iter=1000), RandomForestClassifier(),
KNeighborsClassifier(),
          GradientBoostingClassifier(), XGBClassifier()]
```

```
name = ['LogisticRegression', 'RandomForsetClassifier', 'KNeighborsClassifier',
      'GradientBoostingClassifier', 'XGBClassifier']

models= dict(zip(name,modells))
accuracy_scores=[]
for key,value in models.items():
   value.fit(X_train,y_train)
   y_pred= value.predict(X_test)
   accuracy= accuracy_score(y_test, y_pred)
   accuracy_scores.append(accuracy)
   print(key)
   print(accuracy)


# In[ ]:


# Comparing all the models and plotting
sns.barplot(x= ['LR','RF','KNN','GBC','XGB'],y=accuracy_scores)


# In[ ]:


# Hyperparameter tuning Logistic Regression
model1 = LogisticRegression(max_iter=1000)
solvers = ['newton-cg', 'lbfgs', 'liblinear']
penalty = ['l2']
c_values = [100, 10, 1.0, 0.1, 0.01]

params1 = dict(solver=solvers,penalty=penalty,C=c_values)
grid_search1 = RandomizedSearchCV(estimator=model1,  cv=3,
param_distributions=params1, n_iter=5)
grid_result1 = grid_search1.fit(X, y)
```

```
print("Best: %f using %s" % (grid_result1.best_score_, grid_result1.best_params_))



# In[ ]:



# Hyperparameter tuning Random Forest
model2 = RandomForestClassifier()
n_estimators = [10, 100, 1000]
max_features = ['sqrt', 'log2']

params2 = dict(n_estimators=n_estimators,max_features=max_features)
grid_search2 = RandomizedSearchCV(estimator=model2,  cv=3,
param_distributions=params2, n_iter=5)
grid_result2 = grid_search2.fit(X, y)

print("Best: %f using %s" % (grid_result2.best_score_, grid_result2.best_params_))



# In[ ]:



# Hyperparameter tuning K-Nearest Neighbours
model3 = KNeighborsClassifier()
n_neighbors = range(1, 21, 2)
weights = ['uniform', 'distance']
metric = ['euclidean', 'manhattan', 'minkowski']

params3 = dict(n_neighbors=n_neighbors,weights=weights,metric=metric)
grid_search3 = RandomizedSearchCV(estimator=model3,  cv=3,
param_distributions=params3, n_iter=5)
grid_result3 = grid_search3.fit(X, y)

print("Best: %f using %s" % (grid_result3.best_score_, grid_result3.best_params_))
```

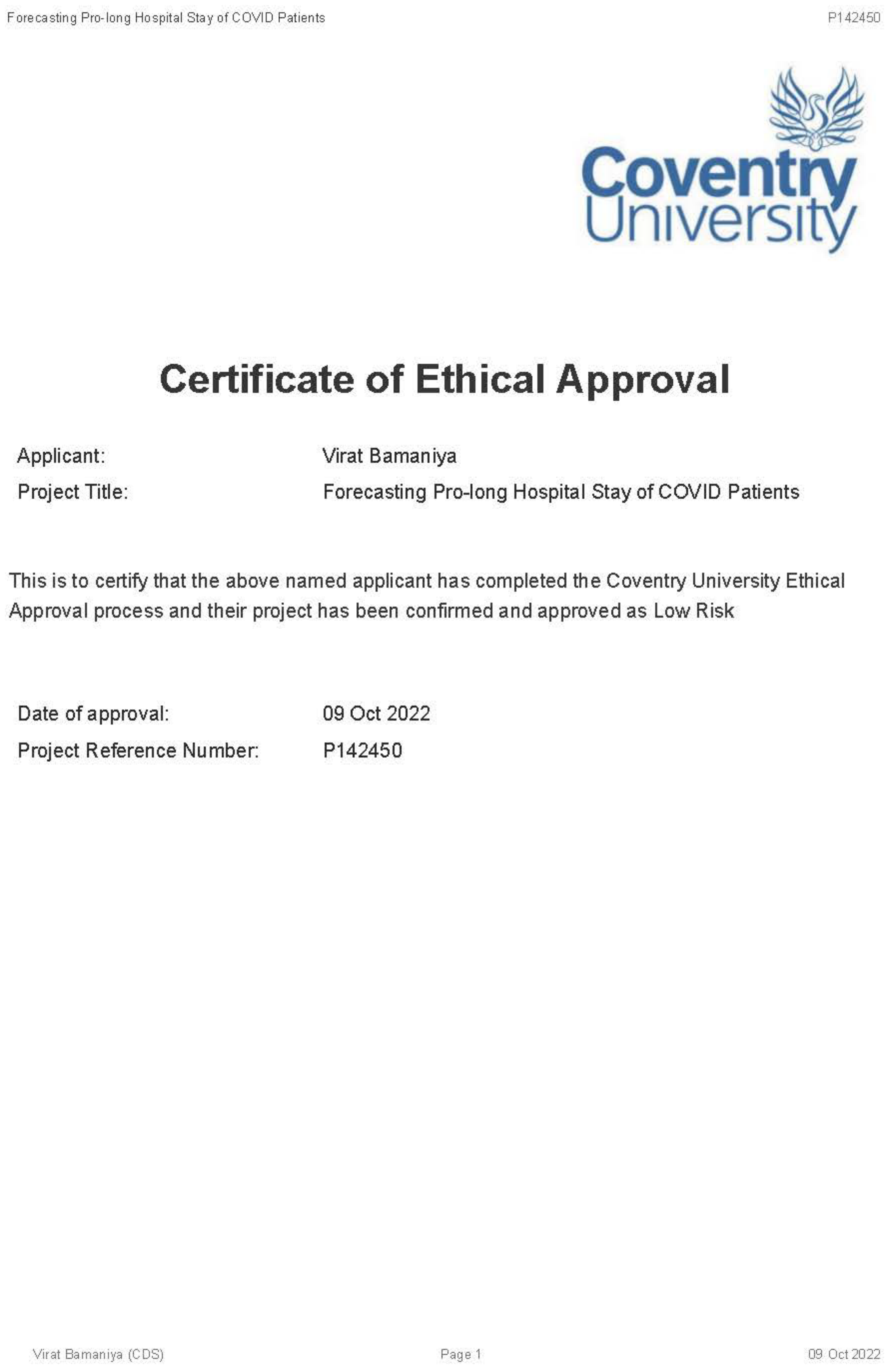
```
# In[ ]:


# Hyperparameter tuning Gradient Boosting, this is sooooo slow
model4 = GradientBoostingClassifier()
n_estimators2 = [10, 100, 1000]
learning_rate = [0.001, 0.01, 0.1]
subsample = [0.5, 0.7, 1.0]
max_depth = [3, 7, 9]

params4 = dict(learning_rate=learning_rate, n_estimators=n_estimators2,
subsample=subsample, max_depth=max_depth)
grid_search4 = RandomizedSearchCV(estimator=model4,  cv=3,
param_distributions=params4, n_iter=5)
grid_result4 = grid_search4.fit(X, y)

print("Best: %f using %s" % (grid_result4.best_score_, grid_result4.best_params_))


# In[ ]:


# Hyperparameter tuning Extreme Gradient Boosting
model5 = XGBClassifier(n_estimators=1000)
objective = ['binary:logistic']
max_depth2 = [3,4,5,6]
min_child_weight = [1,5,10,12]
subsample2 = [0.6,0.8,1.0]
colsample_bytree = [0.6,0.8,1.0]
gamma = [0.5,1,1.5,2]

params5 = dict(objective=objective, max_depth=max_depth2,
min_child_weight=min_child_weight, subsample=subsample2,
        colsample_bytree=colsample_bytree, gamma=gamma)
```

```
grid_search5 = RandomizedSearchCV(estimator=model5,  cv=3,
param_distributions=params5, n_iter=5)
grid_result5 = grid_search5.fit(X, y)

print("Best: %f using %s" % (grid_result5.best_score_, grid_result5.best_params_))


# In[ ]:


# Experimenting to see performance of all models (can't run it, crashes my system)
from lazypredict.Supervised import LazyClassifier
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split


clf = LazyClassifier(verbose=0,ignore_warnings=True, custom_metric=None)
models,predictions = clf.fit(X_train, X_test, y_train, y_test)

print(models)
```

Appendix B – Meetings with Supervisor

Meetings with my supervisor were quite fruitful. Every week, we had a meeting on either Tuesday, Wednesday, or Friday. I value the insightful criticism Long Chen provided me with regarding my project. When I was performing something incorrectly, he taught me the way to go. We spoke about a few topics and whether they were significant or not. Since this was my first fully developed thesis-like paper, whatever comments I received helped me put forth my best effort. Early on, I had a lot of time and was patiently working on my idea. I knew it would take me a while to finish a paper, so I started giving the paper more time and effort to better my paper. Overall, the sessions I had with my supervisor were really beneficial to creating the best version of my project.

Appendix C – Certificate of Ethics Approval



Forecasting Pro-long Hospital Stay of COVID Patients P142450

Coventry University

Certificate of Ethical Approval

Applicant:	Virat Bamaniya
Project Title:	Forecasting Pro-long Hospital Stay of COVID Patients

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:	09 Oct 2022
Project Reference Number:	P142450

Virat Bamaniya (CDS) Page 1 09 Oct 2022