

Prediction Of Admission Of Confirmed Covid-19 Cases To ICU

A PROJECT REPORT

for

DATA MINING TECHNIQUES (SWE2009)

in

M.Tech (Integrated) Software Engineering

by

SIREESHA K (20MIS0009)

BHARATH A (20MIS0413)

Under the Guidance of

Dr. SENTHILKUMAR N C

Associate Professor, SITE



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering

April, 2023

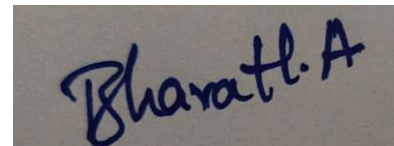
DECLARATION BY THE CANDIDATE

We hereby declare that the project report entitled “**PREDICTION OF ADMISSION OF CONFIRMED COVID-19 CASES TO ICU**” submitted by us to Vellore Institute of Technology; Vellore in partial fulfillment of the requirement for the award of the course **Data Mining Techniques (SWE2009)** is a record of bonafide project work carried out by us under the guidance of **Dr. Senthilkumar N C**. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other course.

Place: Vellore

Signature

Date: 30/03/2023

Handwritten signature in blue ink, reading "Sreeshak".Handwritten signature in black ink, reading "Bharath. A".



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology & Engineering [SITE]

CERTIFICATE

This is to certify that the project report entitled **“PREDICTION OF ADMISSION OF CONFIRMED COVID-19 CASES TO ICU”** submitted by **SIREESHA K (20MIS0009)** to Vellore Institute of Technology, Vellore in partial fulfillment of the requirement for the award of the course **Data Mining Techniques (SWE2009)** is a record of bonafide work carried out by them under my guidance.

Dr. Senthilkumar N C

Associate Professor, SITE

Title of the Project

Abstract

Huge amounts of data is being transferred ever since the evolution of internet and social media. By observing that data in an efficient way we can predict a lot of things early based on the past data. Many data mining techniques and algorithms are available to obtain results from data. The day when covid 19 was declared as pandemic by W.H.O, everyone was surged to their personal protection. Number of patients who were affected by the virus was increasing eventually. Medical facilities were not enough to treat everyone in hospitals. Manpower is also not in sufficient number to take care of the patients. To overcome this issue ,we are going to use the data mining techniques to find if a person affected with Covid-19 need to get admitted in ICU, simply it predicts the severity of the infection This helps the hospitals to take care for the right person in correct time. This prediction aids to save lives and also reduces the burden on the medical staff. We collected the patients dataset from Kaggle to compute results.

Keywords – Logistic regression, Gaussian Naïve Bayes(NB), XGB regrssor, SDG classifier, ensemble model.

I. INTRODUCTION

With the beginning of 2020, the World Health Organisation (WHO) declared the COVID-19 disease caused by the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) as a global pandemic that transmits rapidly between individuals. The COVID-19 pandemic has become a great challenge for healthcare systems due to the urgent need for ICUs that exceeded their capacity. Determining critical patients that require ICU transfer early will be valuable in optimising ICU resources and triage the patients. We have used data mining techniques and compared their accuracy and performance . This comparative analysis provides us with knowledge about which algorithm will have a better accuracy and performance rate , with this conclusion we will be able to predict the ICU admission of patients which will help the hospitals to act accordingly to satisfy the needs of patients also this prediction acts as a warning to both hospitals and the government as it provides a heads-

up regarding the upcoming need for ICUs and hence oxygen cylinders. We could say implementation of this project is very beneficial as it helps in handling the COVID situation seamlessly .

II. BACKGROUND

In this section we will discuss the preliminary concepts used for our project work.

Some of the concepts include – Python programming language for coding, python packages like numpy, pandas for computing the statistics. The data mining algorithms utilized are – logistic regression, Gaussian naïve bayes, SDG classifier, XGB regressor. Data visualization and data mining alogorithms are two preliminary concepts utilized for the implementation of this project.

III. Literature Survey

A numerous algorithms have been discussed and proposed in COVID -19 ICU prediction domain.

[1]. The growing interest in creating artificial intelligence (AI) applications has addressed a number of medical issues. However, given the high potential threat posed by this virus to global public health, such applications remain insufficient. This systematic review focuses on automated AI applications for detecting and diagnosing COVID-19 that are based on data mining and machine learning (ML) algorithms. Also, the paper provides an overview of this critical virus and discusses the limitations of using data mining and ML algorithms, and provide the health sector with the benefits of various classification techniques.

[2]. This study employs machine learning to build a model for analysing risk factors and predicting mortality in COVID-19 ICU patients. Individually, 100 potential risk factors were screened using significance tests, correlation analysis, and factor analysis. The risk prediction model for the prognosis of COVID-19 patients in the ICU was built using conventional logistic regression methods and four machine learning algorithms. To ensure the stability and reliability of the risk prediction model, calibration curves, SHapley Additive Explanations (SHAP), Local Interpretable Model-Agnostic Explanations (LIME), and other methods were

used to interpret and evaluate it. The outcome was determined using ICU deaths from the database.

[3]. In this study, data mining models were created to predict the recovery of COVID-19 infected patients. To develop the models, the decision tree, support vector machine, naive Bayes, logistic regression, random forest, and K-nearest neighbour algorithms were applied directly on the dataset using the Python programming language. The model predicted a minimum and maximum number of days for COVID-19 patients to recover from the virus, as well as the age group of patients who are at high risk of not recovering from the COVID-19 pandemic, those who are likely to recover, and those who are likely to recover quickly. The study's findings indicate that the model developed using the decision tree data mining algorithm is more effective at predicting the possibility of recovery.

[4]. The data mining predictive modelling method for data handling and forecasting the spread of the COVID-19 virus was investigated. This research focuses on predicting or forecasting using fbprophet. Prophet is a Python library package ,It works best with time series that have a strong seasonal effect and data from multiple seasons. The model aids in the interpretation of public sentiment patterns regarding the dissemination of related health information, as well as the assessment of political and economic influence of the spread of the virus.

[5]. To aid Indonesia's development of information and communication technology, this study discovered a correlation between weather and Covid-19. Using a Data Mining Algorithm to process government data sources to produce knowledge that can be used to make decisions. It is accomplished through the use of the ID3 algorithm. ID3 is a decision tree-building supervised learning algorithm. The resulting tree will be used to categorise future samples. According to the study's findings, the weather had an impact on the number of Covid-19 patients. The average duration of sunlight (hour) ,average temperature and average humidity have the highest order of correlation.

[6]. Data mining techniques has recently gained popularity for treating massive amounts of infectious disease data. Cluster analysis, one of the data mining techniques used in this study, is used to classify real groups of infectious disease COVID-19 data sets from different states and union territories (UTs) in India based on their high similarity to each other. The primary

goal of clustering in this study is to optimise monitoring techniques in infected states and UTs in India, which will be very useful to the government, doctors, police, and others involved in understanding the seriousness of COVID-19 spread in order to improve government policies, decisions, medical facilities treatment, and so on to reduce the number of infected and deceased persons.

[7]. The purpose of this paper was to estimate the length of COVID-19 incubation and describe its public health implications. A group analysis of confirmed COVID-19 cases was carried out. Patient demographics were collected, as well as the dates and times of possible exposure, symptom onset, fever onset, and hospitalisation. According to the study, 101 out of every 10,000 cases will develop symptoms after 14 days of active monitoring or quarantine, and there is additional evidence for a median incubation period for COVID-19 of about 5 days, similar to SARS.

[8]. Based on the findings of the research, it can be concluded that the data processing to determine the infected status of COVID-19 using the naïve Bayes algorithm has good outcomes and can be used as a guide for those who are familiar with the determination of the infected status of COVID-19. From the results of data processing manual calculation, rapid miner and web-based applications, it can be concluded that the results of the prediction of the classification of data processing status are infected with covid-19 virus.

[9]. This study is centred on looking at how geographic latitude affects our understanding of how the COVID 19 virus spreads in cases that have been verified. result: To aid scientists and researchers who are still trying to find a vaccine, they should consider the areas where the Covid-19 virus is most likely to spread. It is advantageous to use classification methods, linear regression, and data mining to extract knowledge from data using the Kaggle dataset.

[10]. The goal of this study is to gather and process COVID-19 data from Kaggle using the K-Means Clustering Data Mining technique. As a result, the Data Mining approach of clustering can be utilised to process big data like this. There will be three different ways to handle the data in this research for K-Means Clustering, including using the Weka and KNIME Data Mining programmes as well as the Microsoft Excel software. Two data clusters are identified from the results of the data processing, with cluster 2 having a higher number of confirmed cases and deaths than cluster 1 and hence requiring priority care.

[11]. To begin, regression analysis is used to model the number of confirmed cases and deaths. Experiments show that autoregressive integrated moving average (ARIMA) can reliably model and predict the increase in the number of confirmed cases. Second, a number of classifiers are used to predict whether a COVID-19 patient should be admitted to an ICU or semi-ICU for which Classification algorithms are used. Results show that random forest and hard voting classifiers achieve the highest classification accuracy values near 98%, and the highest recall value of 98% in predicting the need for admission into ICU/semi ICU units.

[12]. This paper describes how to create a predictive model for estimating the risk of ICU admission or mortality in COVID-19 patients hospitalised and provides a user-friendly tool to help clinicians make decisions. Different techniques and machine learning (ML) algorithms were used to build the predictive models, including multilayer perceptron, random forest, and extreme gradient boosting (XGBoost). A reduction dimensionality procedure was used. The model was validated both internally and externally, with calibration and results providing an analysis of the optimal cut-off points depending on the metric to be optimized. The final model achieved good discrimination for the external validation set and accurate calibration.

[13]. The ability of data mining and machine learning, two advanced forms of artificial intelligence, to predict severe COVID-19 pneumonia based on routine laboratory tests was investigated in this study. Following curation, data was processed to identify the most influential features. The Multilayer Perceptron, a feed forward neural network algorithm, or a supervised learning algorithm that generated a decision tree, successfully confirmed the best fit of variables. Finally, a complex bivariate Pearson's correlation matrix combined with advanced hierarchical clustering (dendrograms) were conducted for knowledge discovery was presented.

[14]. The current chemical-informatics-based data analysis approach is an attempt to accelerate the search for current SARS-CoV-2 Mpro inhibitors by using previous activity data of SARS-CoV main protease (Mpro) inhibitors. The study design consisted of three major components they are classification QSAR-based data mining of diverse SARS-CoV Mpro inhibitors, identification of favourable and/or unfavourable molecular features/fingerprints/substructures regulating Mpro inhibitory properties, and data mining-based prediction to validate recently reported natural virtual hits against SARS-CoV-2 Mpro

enzyme. The paper aims to construct an interpretable QSAR model, gradient boosting machine (GBM), random forest (RF), support vector machine (SVM) and k-nearest neighbor (kNN) and establishes using structural and physico-chemical interpretation (SPCI) analysis.

[15]. The effect of preventive strategies on COVID-19 spread was studied in this paper, and a model based on supervised data mining algorithms was presented, with the best algorithm suggested based on accuracy. Three classifiers (Naive Bayes, Multilayer Perceptron, and J48) in this model were based on questionnaires filled out by Basra City respondents. The questionnaires included 25 questions covering areas most relevant to and impacting COVID-19 prevention, such as demographic, psychological, health management, cognitive, awareness, and preventive factors. Weka 3.8 was used to create this model. The findings revealed that quarantine played an important role in limiting disease spread. J48 was determined to be the best algorithm by comparing the accuracy of the algorithms used.

[16]. The authors built a model to analyse and predict the presence of COVID-19 using the COVID-19 Symptoms and Presence dataset from Kaggle and several supervised machine learning and data mining algorithms. WEKA machine learning software was used to apply the J48 Decision Tree, Random Forest, Support Vector Machine, K-Nearest Neighbors, and Nave Bayes algorithms. According to the results, Support Vector Machine with Pearson VII universal kernel outperforms other algorithms.

[17]. This paper describes RAGA research (Rule Acquisition with a Genetic Algorithm). RAGA is a hybrid genetic algorithm and genetic programming designed for supervised and certain types of unsupervised data mining tasks. It computes using statistical factors such as support and confidence. It has predictive accuracy, data coverage, and the ability to generate more scalable rule hierarchies.

[18]. In this study, the COVID-19 was diagnosed using the Chi-square approach, and eight different data mining methods were utilised, including the multilayer perceptron (MLP), J-48, Bayesian net (Bayes Net), logistic regression, K-star, random forest, Ada-boost, and sequential minimal optimization (SMO). Finally, evaluating the effectiveness of the chosen algorithms allowed for the identification of the diagnostic model that is most suitable for COVID-19.

[19]. This study compares customer preferences before and after the outbreak using data mining methods. The authors first obtain reasonably clean comment data by cleaning and sorting the acquired data. The study uses this information to perform word frequency statistics and word cloud mapping and examine customers' psychological behaviour and changes in response to public health situations.

[20]. This study compares different data mining methods such Support Vector Machine, Back Propagation Neural Network, Decision Tree, and K-means clustering, among others, and evaluates how well they perform on the COVID-19 dataset. The purpose of this investigation is to collect and process COVID-19 data from Kaggle using the K-Means Clustering Data Mining technique. As a result, the Data Mining approach of clustering can be used to process such large amounts of data.

[21]. This study's Methodology Corner piece aimed to provide a review of the methodological advancements in big data analytics research and how they can be applied to analyse current organisational and management problems resulting from the global pandemic caused by COVID-19 as well as other grand challenges of the modern era.

[22]. The purpose of this study was to see how well the Quick COVID-19 Severity Index (qCSI) and the Brescia-COVID Respiratory Severity Scale (BCRSS) predicted ICU admissions and in-hospital mortality in patients with coronavirus disease 2019 (COVID-19) pneumonia. The area under the receiver operating characteristic curve (AUC) was used to compare the discriminatory power of the qCSI score and BCRSS prediction rule for predicting mortality and intensive care unit admission to the CURB-65 score. The qCSI score had the highest numerical AUC for ICU admission when compared to the BCRSS prediction rule and the CURB-65 score. The qCSI score and the BCRSS prediction rule showed a good performance for predicting ICU admission.

[23]. The purpose of this study was to see if the traditional critical care severity scores qSOFA, SOFA, APACHE-II, and SAPS-II could predict which patients admitted from an emergency department would eventually require intensive care. Chart abstraction was used to perform clinical phenotyping, and the correlation of the qSOFA, SOFA, APACHE-II, and SAPS-II scores for the primary endpoint of ICU admission and secondary endpoint of in-hospital mortality was assessed. The APACHE-II score performed the best of the three for

identifying patients who would require ICU admission. The paper provides a way to risk-stratification of patients safe to treat in alternative health care settings and prognostic enrichment to accelerate clinical trials of COVID-19 therapies.

[24]. The goal of this study was to see if admission albumin levels predict outcomes in COVID-19 patients. The following outcomes were recorded for the study: venous thromboembolism (VTE), acute respiratory distress syndrome (ARDS), ICU admission, discharge with new or higher home oxygen supplementation, readmission within 90 days, in-hospital mortality, and total adverse events. A multivariate modified Poisson regression analysis was then used to identify significant predictors of increased adverse events in COVID-19 pneumonia patients. Higher albumin levels on admission were associated with significantly fewer adverse outcomes, including fewer VTE events, ARDS development, ICU admissions, and readmissions within 90 days, according to the study.

[25]. The number of confirmed Covid-19 cases and deaths caused by this virus in Southeast Asia was increasing and quite alarming. This study looks at the clustering of COVID-19 cases and deaths in Southeast Asia. The K-Means Clustering Data Mining method was used. The obtained data can be grouped into several clusters using this method, which employs the K-Means Clustering Process and RapidMiner tools. The data is divided into three clusters: high (C1), medium (C2), and low (C3) (C3). The results show that four countries have a high level cluster, one country has a moderate level cluster, and six countries have a low level cluster. This can be a factor for each country to consider.

[26]. Making clusters of countries with similar types of health care quality provides insight into health care quality in different countries. The K-means clustering algorithm is commonly used in machine learning and data science to create clusters based on similarity. In this paper, an efficient K-means clustering method for determining cluster initial centroids was proposed. The proposed method was used to determine health care quality clusters of countries using the COVID-19 datasets. When compared to the traditional k-means clustering algorithm, the experimental results show that the proposed method reduces the number of iterations and execution time required to analyse COVID-19.

[27]. The goal of this study was to create and compare models for predicting ICU admission in COVID-19 patients at the time of hospitalisation. The Chi-square test and logistic

regression were used to identify the most important variables. The selected variables were used to train seven decision tree (DT) algorithms: Hoeffding tree (HT), LMT, J-48, random forest (RF), random tree (RT), and REP-Tree. Finally, the performance of the models was assessed. In addition, an external dataset was used to validate the prediction models. Implementing such models has the potential to inform clinicians and managers on the best policy to adopt and prepare for the COVID-19 time-sensitive and resource-constrained period.

[28]. This survey's main objective is to identify the most popular data mining techniques and knowledge gaps from existing literature. This study identified prominent COVID-19 researchers, institutions, countries, and publications and provided a systematic review of an exhaustive overview of integrated AI-based DM and machine learning algorithms with the CoV family. It can also help the research community identify and prioritise research needs. In this study, the data mining techniques employed in global pandemics were examined.

[29]. This paper study the epidemiological models and data mining have comparable tasks, this paper has highlighted the significance of data mining correlation. These comprise: describing and forecasting the behaviour of a disease. extracting pertinent information or finding patterns. Considering that mathematical models' degrees of complexity might vary, just like those of other data mining methods. researching the short- and long-term pandemic evolution estimating values using past information.

[30]. This study found that following Pfizer-BioNTech Dose 2, there were post-vaccination clusters of unknown adverse effects, frequent vaccine reactions, and, for the mRNA vaccinations, chest pain and palpitations as well as myocarditis/pericarditis. The untargeted character of this data mining and its built-in compensation for the variety of diagnoses and risk intervals screened are its distinctive advantages.

IV. DATASET DESCRIPTION & SAMPLE DATA

This dataset contains anonymized data from Hospital Sírío-Libanês, São Paulo and Brasilia. It consists Patient demographic information, Patient previous grouped diseases, Blood results , Vital signs. In total there are 54 features, expanded when pertinent to the mean, median, max, min, diff and relative diff.

[LINK](#) –

https://docs.google.com/spreadsheets/d/17dprb1R2xDdjjGH1ifxps8mEzWND17Qt/edit?usp=share_link&ouid=115197573204533587252&rtpof=true&sd=true

SAMPLE DATA -

A1		PATIENT_VISIT_IDENTIFIER																												
T	ISE_ABOVE_P	CENT	GENDER	ISE	GROUPSE	GROUPSE	GROUPSE	GROUPSE	GROUPSE	GROUPSE	HTN	NOCOMP	PRC	OTHER	UMIN_ME	BUMIN_ME	BUMIN_M	BUMIN_ML	BUMIN_D	TERIAL_ML	TER									
1	0	1 60th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
2	0	1 60th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
3	0	1 60th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
4	0	1 60th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
5	0	1 60th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
6	0	1 60th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
7	1	1 90th	1	0	0	0	0	0	0	0	0	0	1	1	-0.21053	-0.21053	-0.21053	-0.21053	-1	-1										
8	1	1 90th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
9	1	1 90th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
10	1	1 90th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
11	1	1 90th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
12	2	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
13	2	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
14	2	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
15	2	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
16	2	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
17	3	0 40th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
18	3	0 40th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
19	3	0 40th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
20	3	0 40th	1	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
21	3	0 40th	1	0	0	0	0	0	0	0	0	0	0	1	-0.26316	-0.26316	-0.26316	-0.26316	-1	-1										
22	4	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
23	4	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
24	4	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
25	4	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
26	4	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
27	5	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
28	5	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
29	5	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
30	5	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
31	5	0 10th	0	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
32	6	1 70th	1	0	0	0	0	0	0	0	0	0	1	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
33	6	1 70th	1	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
34	6	1 70th	1	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
35	6	1 70th	1	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
36	6	1 70th	1	0	0	0	0	0	0	0	0	0	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1										
37	7	0 20th	0	0	0	0	0	0	0	0	0	0	1	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1									
38	7	0 20th	0	0	0	0	0	0	0	0	0	0	1	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1									
39	7	0 20th	0	0	0	0	0	0	0	0	0	0	1	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1									
40	7	0 20th	0	0	0	0	0	0	0	0	0	0	1	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1									
41	7	0 20th	0	0	0	0	0	0	0	0	0	0	1	0	1	0.605263	0.605263	0.605263	0.605263	-1	-1									

	A1			PATIENT_VISIT_IDENTIFIER																			
	AW	AX	AY	AZ	BA	BB	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN	BO	GGT			
1	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED	CUM_MED			
2																							
3																							
4	0.183673	0.183673	0.183673	0.183673	-1	-0.86837	-0.86837	-0.86837	-0.86837	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.94509	-0.94509	-0.94509	-0.94509				
5																							
6	0.326531	0.326531	0.326531	0.326531	-1	-0.9264	-0.9264	-0.9264	-0.9264	-1	-0.85928	-0.85928	-0.85928	-0.85928	-1	-0.66939	-0.66939	-0.66939	-0.66939				
7																							
8	0.530612	0.530612	0.530612	0.530612	-1	-0.615	-0.615	-0.615	-0.615	-1	-0.76972	-0.76972	-0.76972	-0.76972	-1	-0.98248	-0.98248	-0.98248	-0.98248				
9																							
10	0.367347	0.367347	0.367347	0.367347	-1	-0.9084	-0.9084	-0.9084	-0.9084	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
11	0.530612	0.530612	0.530612	0.530612	-1	-0.43524	-0.43524	-0.43524	-0.43524	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
12	0.357143	0.357143	0.357143	0.357143	-1	-0.91224	-0.91224	-0.91224	-0.91224	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
13																							
14																							
15																							
16	0.367347	0.367347	0.367347	0.367347	-1	-0.90658	-0.90658	-0.90658	-0.90658	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
17																							
18																							
19																							
20																							
21	0.326531	0.326531	0.326531	0.326531	-1	-0.96886	-0.96886	-0.96886	-0.96886	-1	-0.19403	-0.19403	-0.19403	-0.19403	-1	-0.31659	-0.31659	-0.31659	-0.31659				
22	0.357143	0.357143	0.357143	0.357143	-1	-0.91366	-0.91366	-0.91366	-0.91366	-1	-0.82942	-0.82942	-0.82942	-0.82942	-1	-0.93808	-0.93808	-0.93808	-0.93808				
23																							
24	0.357143	0.357143	0.357143	0.357143	-1	-0.9084	-0.9084	-0.9084	-0.9084	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
25																							
26	0.357143	0.357143	0.357143	0.357143	-1	-0.91366	-0.91366	-0.91366	-0.91366	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
27	0.357143	0.357143	0.357143	0.357143	-1	-0.89101	-0.89101	-0.89101	-0.89101	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
28																							
29																							
30																							
31	0.489796	0.489796	0.489796	0.489796	-1	-0.88393	-0.88393	-0.88393	-0.88393	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
32	0.357143	0.357143	0.357143	0.357143	-1	-0.9448	-0.9448	-0.9448	-0.9448	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
33																							
34																							
35																							
36	0.357143	0.357143	0.357143	0.357143	-1	-0.93631	-0.93631	-0.93631	-0.93631	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
37	0.357143	0.357143	0.357143	0.357143	-1	-0.91224	-0.91224	-0.91224	-0.91224	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
38	0.357143	0.357143	0.357143	0.357143	-1	-0.9084	-0.9084	-0.9084	-0.9084	-1	-0.742	-0.742	-0.742	-0.742	-1	-0.95853	-0.95853	-0.95853	-0.95853				
39																							

V. PROPOSED ALGORITHM WITH FLOWCHART

To overcome the pitfalls of various algorithms we use a combination of binary classification models like logistic regression, Gaussian Navie Bayes, SDG(stochastic gradient descent) classifier and XGB(extreme gradient boosting) to train the data. We predict the result, compute evaluation metrics, accuracy, confusion metrics and ROC curve using an ensemble model, also the performance of individual model is compared.

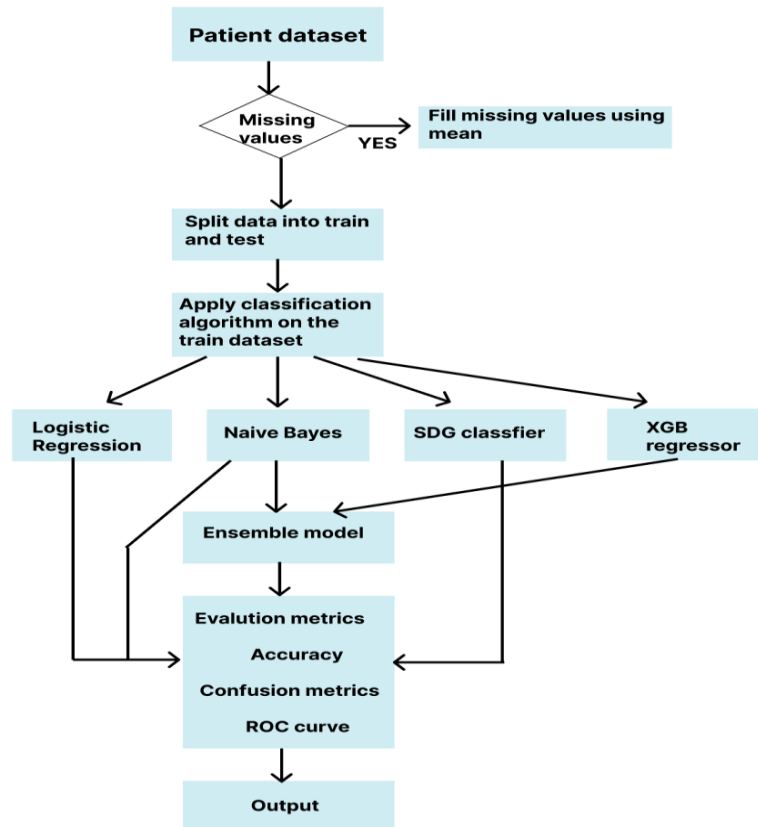
STEPS TO BE FOLLOWED

1. Import necessary packages and load the data set. Understand the data set with metadata.
2. Clean the null values in the data using mean.
3. Split the data set into train and test data.
4. Import the model regression and train model with given data.
5. Modulate data using k-fold techniques and train the models separately

Each type of regression model is trained separately two times once with straight data and again with modified data.

6. Perform evaluation metrics like accuracy, confusion matrix and roc curve is computed for each model.
7. Create an ensemble model using the regression types and train the ensemble model.
8. Conduct evaluation metrics for the ensemble model.

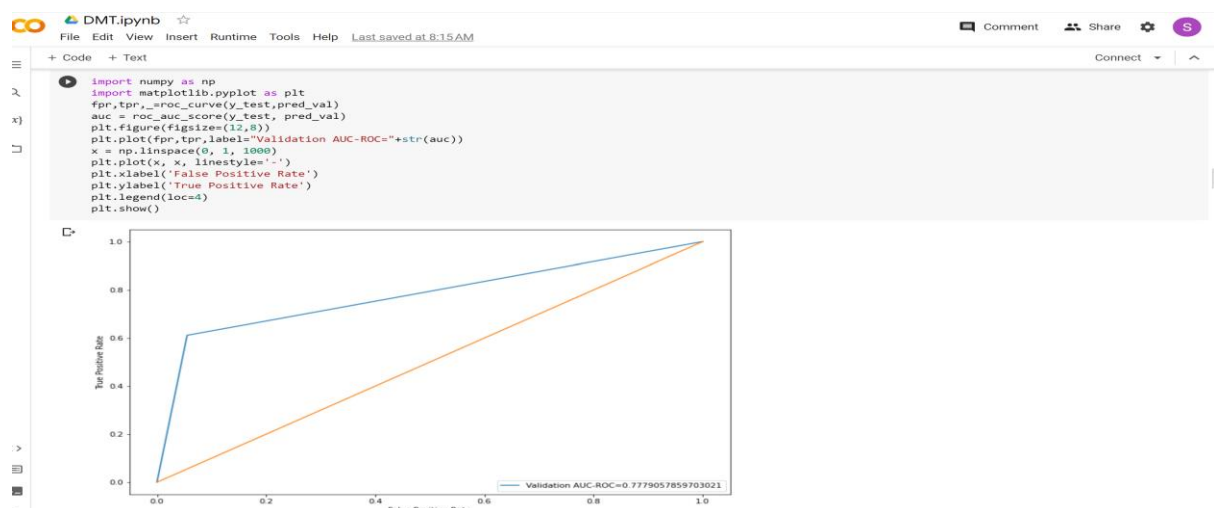
FLOWCHART



VI. EXPERIMENTS RESULTS

In our project we have computed logistic regression , Naïve bayes, SDG classifier, XGB regressor algorithms. We used an ensemble model to collaborate these algorithms, have computed the evaluation metrics, accuracy, confusion metrics and roc curve.

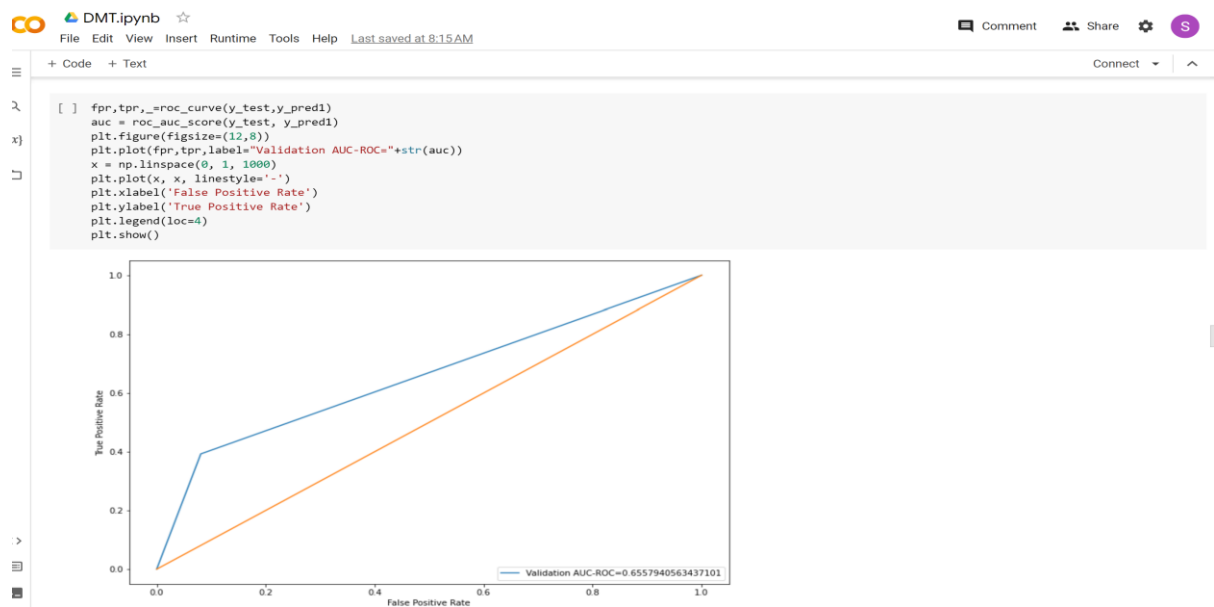
LOGISTIC REGRESSION



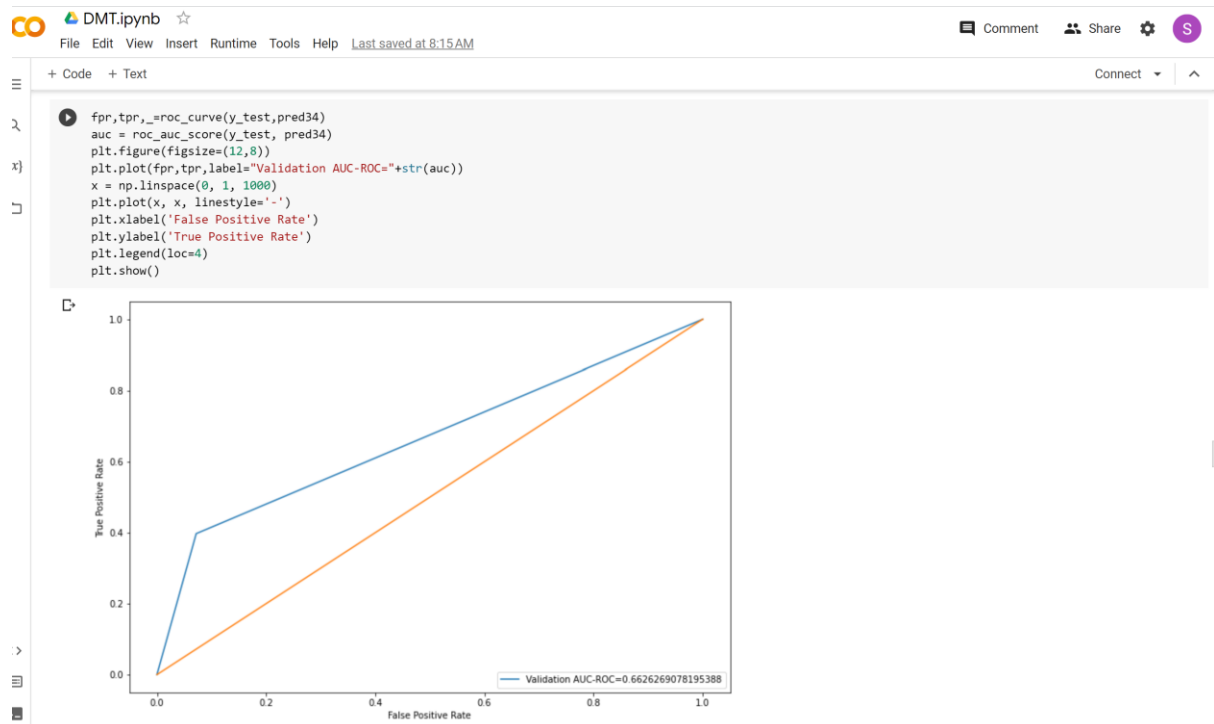
LOGISTIC REGRESSION WITH STRATIFIED K FOLD



GAUSSIAN NB



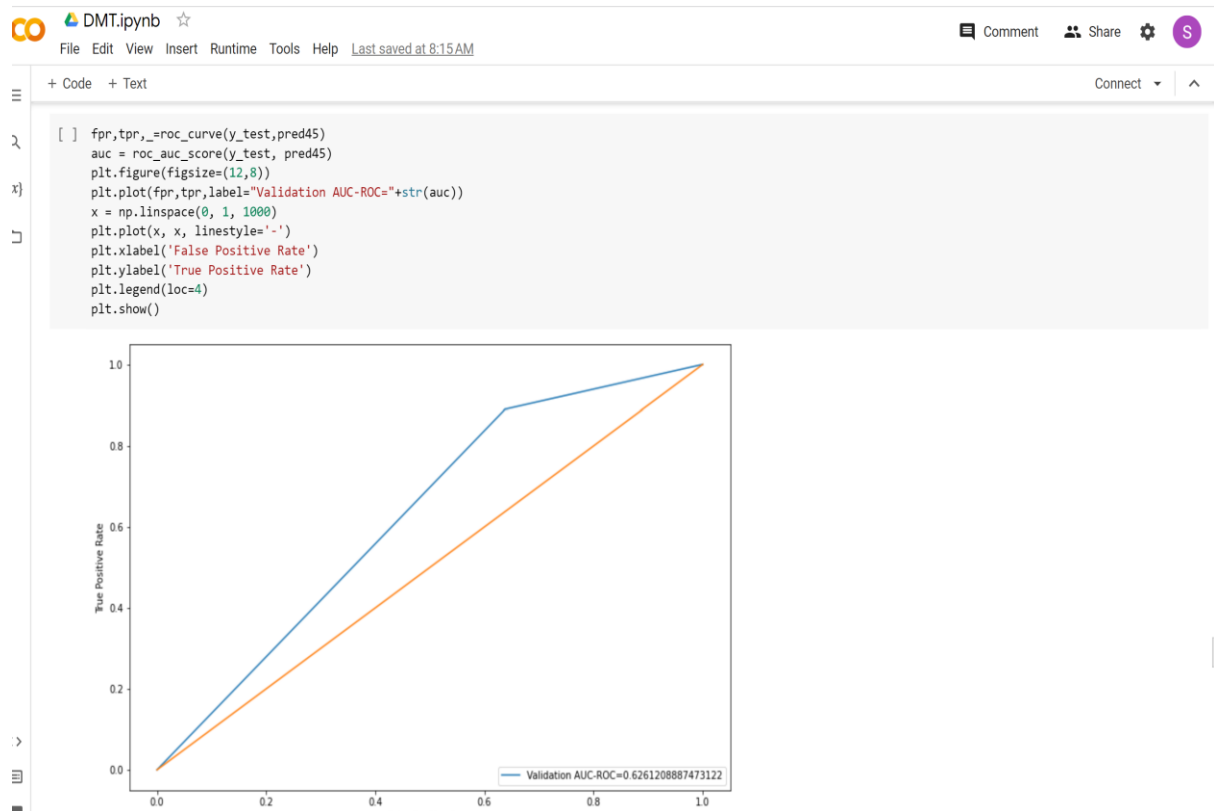
GAUSSIAN NB WITH STRATIFIED K FLOD



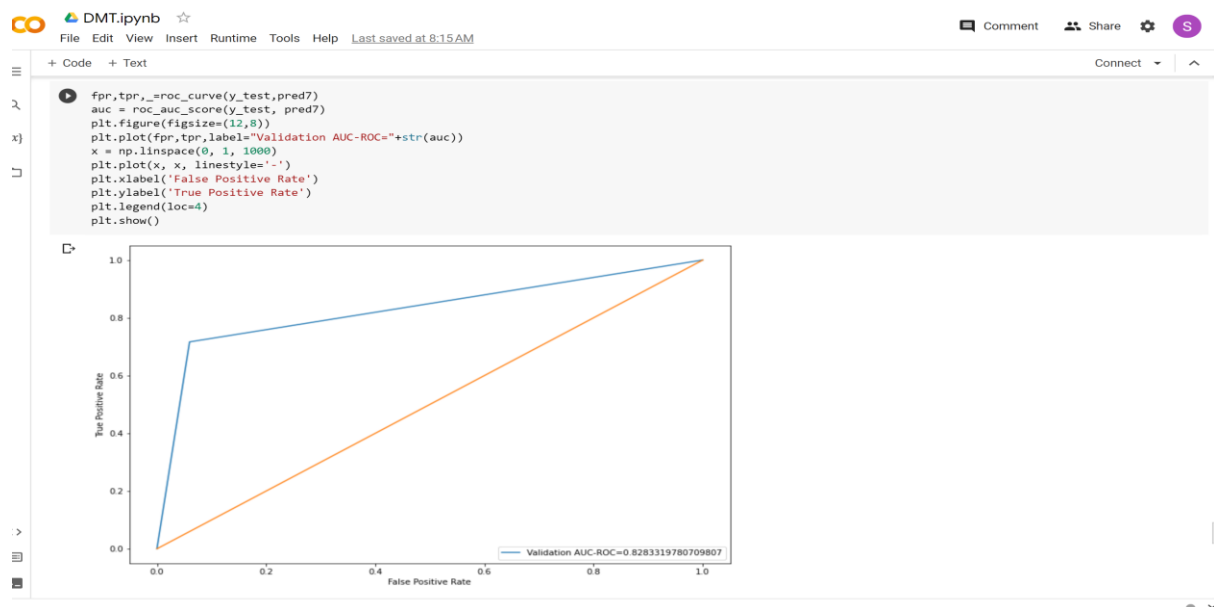
SDG CLASSIFIER



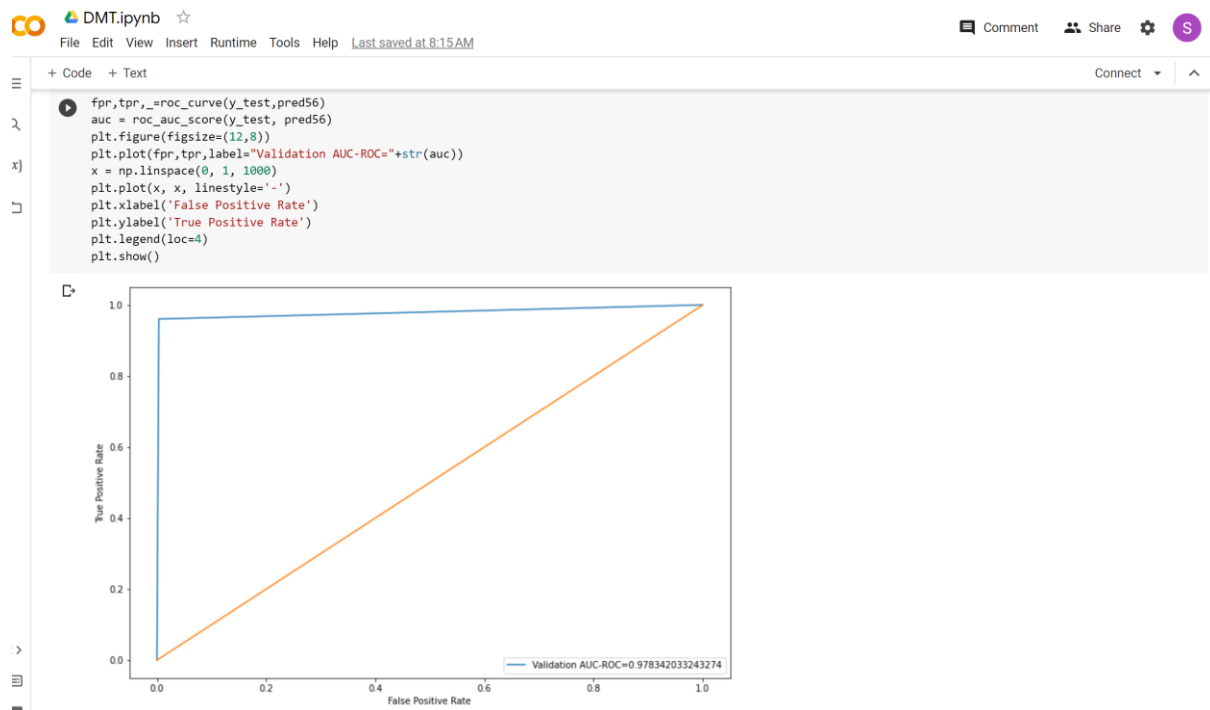
SDG CLASSIFIER WITH STRATIFIED K FOLD



XGB REGRESSOR



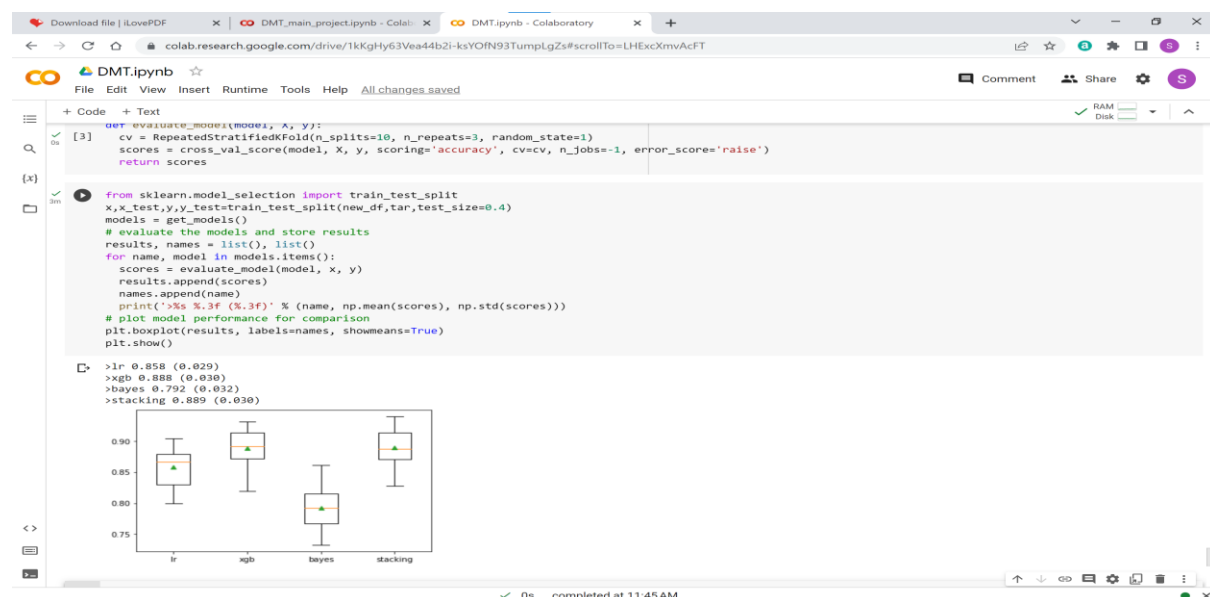
XGB REGRESSOR WITH STRATIFIED K FLOD



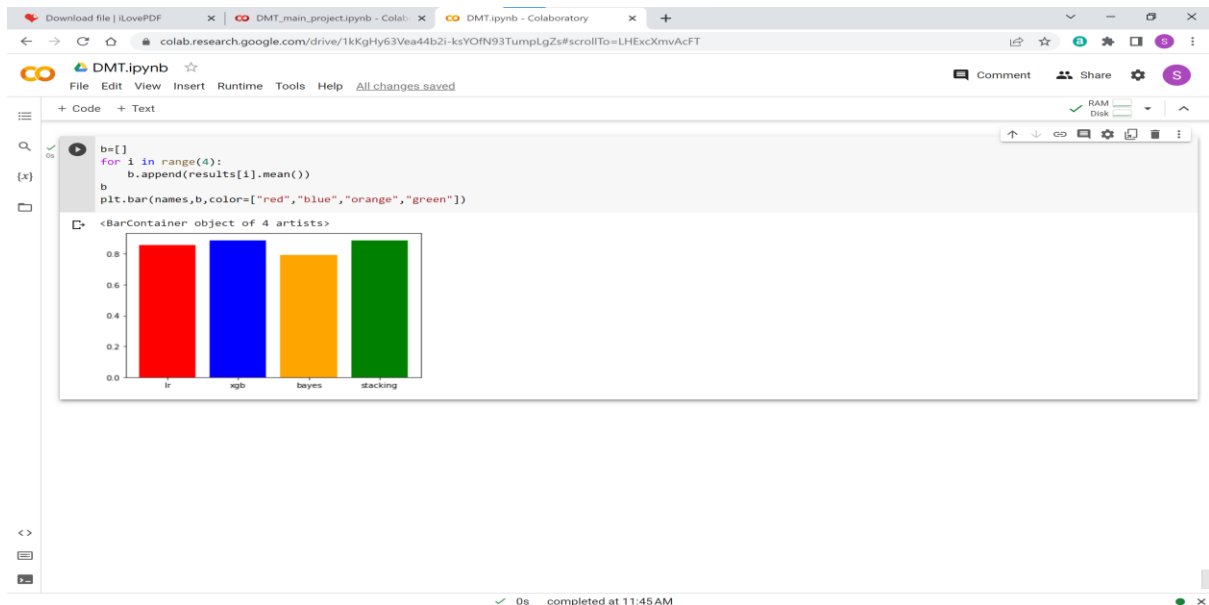
VII. COMPARATIVE STUDY / RESULTS AND DISCUSSION

We have performed comparison of the performance of logistic regression, Naïve Bayes, SDG classifier and XGB regressor. With the help of ensemble model we have combined the various models into one stack model.

The comparison is done using box plot and bar plot.



The graphical representation of the comparative study between the various algorithms is represented below



Algorithm	Accuracy
Logistic regression	86%
Gaussian NB	77%
SDG classifier	79%
XGB regressor	89%

VIII. CONCLUSION AND FUTURE WORK

We can say the XGB regressor algorithm has more prediction accuracy compared to logistic regression, Gaussian NB and SDG classifier. Comparatively we can say that Logistic regression and XGB has more accuracy rate than Gaussian NB and SDG classifier.

When we use the ensemble model to create a stack of these individual models we can conclude that the stacked output done with the help of ensemble model has more performance compared to individual models.

FUTURE SCOPE

We can create an interface to where we will be able to give csv file as input and predict if the person is 1(admitted to ICU) or 0(not need of admission to ICU), this could be done with the help for any individual algorithm which we computed earlier.

For example we can consider logistic regression algorithm as it has good accuracy rate and better performance. Using python modules like – pandas , tkinter, we can create an interface

to upload the csv file and predict if patient is admitted or not. To create an interface the some features need to be imported from tkinter module they are – fielddialog, messagebox, tkintertable and tablecanvas.

IX. REFERENCES

1. Role of biological data mining and machine learning techniques in detecting and diagnosing the novel coronavirus (COVID-19): a systematic review. Journal of medical systems – May, 2020.
2. Prognostic assessment of COVID-19 in the intensive care unit by machine learning methods: model development and validation : August,2020.
3. Predictive data mining models for novel coronavirus (COVID-19) infected patients' recovery : June,2020.
4. COVID-19 outbreak data analysis and prediction modeling using data mining technique : May,2020.
5. Simple correlation between weather and COVID-19 pandemic using data mining algorithms : December,2020.
6. Monitoring novel corona virus (COVID-19) infections in India by cluster analysis : May,2020.
7. The incubation period of coronavirus disease 2019 (COVID-19) from publicly reported confirmed cases: estimation and application : May,2020.
8. Application of Data Mining Classification for Covid-19 Infected Status Using Algorithm a Naïve Method: Application of Data Mining Classification for Covid-19 Infected Status Using Algorithm a Naïve Method: May,2020.
9. Study impact the latitude on Covid-19 spread virus by data mining algorithm: November,2020.
10. K-Means Clustering Data COVID-19: December,2020.

11. Forecasting the Spread of COVID-19 and ICU Requirements – March, 2021.
12. A clinical decision web to predict ICU admission or death for patients hospitalised with COVID-19 using machine learning algorithms : August, 2021.
13. Biomarkers of severe COVID-19 pneumonia on admission using data-mining powered by common laboratory blood tests-datasets : August 2021.
14. Chemical-informatics approach to COVID-19 drug discovery: Exploration of important fragments and data mining based prediction of some hits from natural origins as main protease (Mpro) inhibitors : August,2021.
15. Predictions of COVID-19 spread by using supervised data mining techniques : March,2021.
16. COVID-19 Prediction applying supervised machine learning algorithms with comparative analysis using WEKA. Algorithms :June,2021.
17. Supervised and unsupervised data mining with an evolutionary algorithm : May,2021.
18. Performance analysis of data mining algorithms for diagnosing COVID-19: November,2021.
19. The impact of COVID-19 on consumers' psychological behavior based on data mining for online user comments in the catering industry in China: April,2021.
20. Comparative Study of Various Data Mining Techniques Towards Analysis and Prediction of Global COVID-19 Dataset. Machine Learning for Healthcare Applications: April,2021.
21. COVID-19 pandemic in the new era of big data analytics: Methodological innovations and future research directions: October,2021.
22. Performance of the quick COVID-19 severity index and the Brescia-COVID respiratory severity scale in hospitalized patients with COVID-19 in a community hospital setting: January,2021.
23. Severity of illness scores at presentation predict ICU admission and mortality in COVID-19: January,2021.

24. Higher albumin levels on admission predict better prognosis in patients with confirmed COVID-19: March,2021.
25. Covid-19 cases and deaths in southeast Asia clustering using k-means algorithm: February,2021.
26. An efficient k-means clustering algorithm for analysing covid-19: April,2021.
27. Using decision tree algorithms for estimating ICU admission of COVID-19 patients : March 2022
28. Using Data Mining Techniques for COVID-19: A Systematic. Science and Technology : June,2022.
29. The State of the Art of Data Mining Algorithms for Predicting the COVID-19 Pandemic: May,2022.
30. A broad assessment of covid-19 vaccine safety using tree-based data-mining in the vaccine safety data link: January,2023.

Appendix

CODE

```
# -*- coding: utf-8 -*-
"""DMT.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1kKgHy63Vea44b2i-ksY0fN93TumpLgZs
"""

from google.colab import drive
drive.mount('/content/drive',force_remount=True)

from sklearn.impute import SimpleImputer
import pandas as pd
df=pd.read_excel("/content/drive/MyDrive/Kaggle_Sirio_Libanes_ICU_Prediction[1].xlsx")

df.columns
```

```

df.isnull().sum()

# Data Preparation
df['AGE_PERCENTIL'] = df['AGE_PERCENTIL'].str.replace('Above', '').str.extract(r'(.+?)th')
df['WINDOW'] = df['WINDOW'].str.replace('ABOVE_12', '12-more').str.extract(r'(.+?)-')

# Missingness as features
df['row_missingness'] = df.isnull().sum(axis=1)

# Mean imputation
mean_impute = SimpleImputer(strategy='mean')
imputed_data = mean_impute.fit_transform(df)
imputed_data = pd.DataFrame(imputed_data, columns = df.columns)

imputed_data.isnull().sum()

imputed_data.head(5)

target=["ICU"]
un=["row_missingness"]
numericals=list(set(imputed_data.columns.values)-set(target)-set(un))
len(numericals)

new_df=imputed_data[numericals]
new_df.shape
tar=imputed_data[target]
tar.shape
from sklearn.model_selection import train_test_split
x,x_test,y,y_test=train_test_split(new_df,tar,test_size=0.3)
x.shape

y.head(5)

import warnings
warnings.filterwarnings('ignore')
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x,y)

pred_val=lr.predict(x_test)

lr.predict_proba(x_test)[:,:1]

from sklearn.metrics import
accuracy_score,confusion_matrix,roc_auc_score,roc_curve

```



```

accuracy_score(y_test,pred_val)

confusion_matrix(y_test,pred_val)

import pickle
saved_model1=pickle.dumps(lr)

import numpy as np
import matplotlib.pyplot as plt
fpr,tpr,_=roc_curve(y_test,pred_val)
auc = roc_auc_score(y_test, pred_val)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

model23=LogisticRegression()

from sklearn.model_selection import StratifiedKFold
accuracy=[]
skf=StratifiedKFold(n_splits=10,random_state=None)
skf.get_n_splits(new_df,tar)

for train_index,test_index in skf.split(new_df,tar):
    x1_train,x1_test=new_df.iloc[train_index],new_df.iloc[test_index]
    y1_train,y1_test=tar.iloc[train_index],tar.iloc[test_index]
    model23.fit(x1_train,y1_train)
    prediction=model23.predict(x1_test)
    score=accuracy_score(prediction,y1_test)
    accuracy.append(score)
print(accuracy)

np.array(accuracy).mean()
pred23=model23.predict(x_test)
pred23

fpr,tpr,_=roc_curve(y_test,pred23)
auc = roc_auc_score(y_test, pred23)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')

```

```

plt.legend(loc=4)
plt.show()

confusion_matrix(y_test,pred23)

import pickle

saved_model2=pickle.dumps(model23)

from sklearn.model_selection import train_test_split
x,x_test,y,y_test=train_test_split(new_df,tar,test_size=0.4)

from sklearn.naive_bayes import GaussianNB
nb=GaussianNB()
nb.fit(x,y)

y_pred1=nb.predict(x_test)
y_pred1

from sklearn.metrics import accuracy_score

accuracy_score(y_test,y_pred1)

confusion_matrix(y_test,y_pred1)

fpr,tpr,_=roc_curve(y_test,y_pred1)
auc = roc_auc_score(y_test, y_pred1)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

import pickle
saved_model3=pickle.dumps(nb)

from sklearn.model_selection import StratifiedKFold
acc=[]
skf=StratifiedKFold(n_splits=10,random_state=None)
skf.get_n_splits(new_df,tar)

from sklearn.naive_bayes import GaussianNB
nb2=GaussianNB()
for train_index,test_index in skf.split(new_df,tar):
    x1_train,x1_test=new_df.iloc[train_index],new_df.iloc[test_index]

```

```

y1_train,y1_test=tar.iloc[train_index],tar.iloc[test_index]
nb2.fit(x1_train,y1_train)
prediction=nb2.predict(x1_test)
score=accuracy_score(prediction,y1_test)
acc.append(score)
print(acc)

np.array(acc).mean()

pred34=nb2.predict(x_test)

fpr,tpr,_=roc_curve(y_test,pred34)
auc = roc_auc_score(y_test, pred34)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

confusion_matrix(y_test,pred34)

saved_model4=pickle.dumps(nb2)

from sklearn.model_selection import train_test_split
x,x_test,y,y_test=train_test_split(new_df,tar,test_size=0.3)
from sklearn.linear_model import SGDClassifier
sgd=SGDClassifier(loss='modified_huber',shuffle=True,random_state=101)
sgd.fit(x,y)
pred=sgd.predict(x_test)
pred

accuracy_score(y_test,pred)

confusion_matrix(y_test,pred)

fpr,tpr,_=roc_curve(y_test,pred)
auc = roc_auc_score(y_test, pred)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

```

```

saved_model5=pickle.dumps(sgd)

from sklearn.model_selection import StratifiedKFold
acc2=[]
skf=StratifiedKFold(n_splits=10,random_state=None)
skf.get_n_splits(new_df,tar)

from sklearn.linear_model import SGDClassifier
sgd2=SGDClassifier(loss='modified_huber',shuffle=True,random_state=101)
for train_index,test_index in skf.split(new_df,tar):
    x1_train,x1_test=new_df.iloc[train_index],new_df.iloc[test_index]
    y1_train,y1_test=tar.iloc[train_index],tar.iloc[test_index]
    sgd2.fit(x1_train,y1_train)
    prediction=nb2.predict(x1_test)
    score=accuracy_score(prediction,y1_test)
    acc2.append(score)
print(acc2)
saved_model6=pickle.dumps(sgd2)

np.array(acc2).mean()

pred45=sgd2.predict(x_test)

fpr,tpr,_=roc_curve(y_test,pred45)
auc = roc_auc_score(y_test, pred45)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

confusion_matrix(y_test,pred45)

from sklearn.model_selection import train_test_split
x,x_test,y,y_test=train_test_split(new_df,tar,test_size=0.4)
from xgboost import XGBClassifier
model7=XGBClassifier()
model7.fit(x,y)

pred7=model7.predict(x_test)

pred7

accuracy_score(pred7,y_test)

```

```

confusion_matrix(y_test,pred7)

fpr,tpr,_=roc_curve(y_test,pred7)
auc = roc_auc_score(y_test, pred7)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='-')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

saved_model7=pickle.dumps(model7)

from sklearn.model_selection import StratifiedKFold
acc3=[]
skf=StratifiedKFold(n_splits=10,random_state=None)
skf.get_n_splits(new_df,tar)

from xgboost import XGBClassifier
model8=XGBClassifier()
for train_index,test_index in skf.split(new_df,tar):
    x1_train,x1_test=new_df.iloc[train_index],new_df.iloc[test_index]
    y1_train,y1_test=tar.iloc[train_index],tar.iloc[test_index]
    model8.fit(x1_train,y1_train)
    prediction=model8.predict(x1_test)
    score=accuracy_score(prediction,y1_test)
    acc3.append(score)
print(np.array(acc3).mean())

pred56=model8.predict(x_test)

fpr,tpr,_=roc_curve(y_test,pred56)
auc = roc_auc_score(y_test, pred56)
plt.figure(figsize=(12,8))
plt.plot(fpr,tpr,label="Validation AUC-ROC="+str(auc))
x = np.linspace(0, 1, 1000)
plt.plot(x, x, linestyle='-')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc=4)
plt.show()

confusion_matrix(y_test,pred56)

from sklearn.ensemble import StackingClassifier

```

```

from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.model_selection import cross_val_score
def get_stacking():
    # define the base models
    level0 = list()
    level0.append(('lr', LogisticRegression()))
    level0.append(('xgb', XGBClassifier()))
    level0.append(('bayes', GaussianNB()))
    # define meta learner model
    level1 = LogisticRegression()
    # define the stacking ensemble
    model = StackingClassifier(estimators=level0, final_estimator=level1,
cv=5)
    return model

def get_models():
    models = dict()
    models['lr'] = LogisticRegression()
    models['xgb'] = XGBClassifier()
    models['bayes'] = GaussianNB()
    models['stacking'] = get_stacking()
    return models

def evaluate_model(model, X, y):
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
    scores = cross_val_score(model, X, y, scoring='accuracy', cv=cv, n_jobs=-
1, error_score='raise')
    return scores

from sklearn.model_selection import train_test_split
x,x_test,y,y_test=train_test_split(new_df,tar,test_size=0.4)
models = get_models()
# evaluate the models and store results
results, names = list(), list()
for name, model in models.items():
    scores = evaluate_model(model, x, y)
    results.append(scores)
    names.append(name)
    print('>%s %.3f (%.3f)' % (name, np.mean(scores), np.std(scores)))
# plot model performance for comparison
plt.boxplot(results, labels=names, showmeans=True)
plt.show()

b=[]
for i in range(4):
    b.append(results[i].mean())
b
plt.bar(names,b,color=["red","blue","orange","green"])

```

Prediction

Code

```
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('ggplot')

from IPython.display import display_html
```

```
import os
print(os.listdir("../"))

['lib', 'input', 'working']
```

```
#Loading data
data = pd.read_excel("../input/Kaggle_Sirio_Libanes_ICU_Prediction.xlsx")

comorb_lst = [i for i in data.columns if "DISEASE" in i]
comorb_lst.extend(["HTN", "IMMUNOCOMPROMISED", "OTHER"])

demo_lst = [i for i in data.columns if "AGE_" in i]
demo_lst.append("GENDER")

vitalSigns_lst = data.iloc[:,193:-2].columns.tolist()

lab_lst = data.iloc[:,13:193].columns.tolist()
```

```
print(f"Number of Comorbidities features: {len(comorb_lst)}")
print(f"Number of Demographics features: {len(demo_lst)}")
print(f"Number of Vital Signs features: {len(vitalSigns_lst)}")
print(f"Number of Laboratory features: {len(lab_lst)}")

Number of Comorbidities features: 9
Number of Demographics features: 3
Number of Vital Signs features: 36
Number of Laboratory features: 180
```

```
# ID is a identification number for each patient.
print(f"Number of lines in the dataset: {len(data)}")
print(f"Number of inpatients: {len(data.PATIENT_VISIT_IDENTIFIER.unique())}")

Number of lines in the dataset: 1925
Number of inpatients: 385
```

```
#From hospital admission, there are five time windows. Please, observe that there is a flag pointing out whether the patient was or not admitted in the ICU as
print(data.WINDOW.unique())

Python

['0-2' '2-4' '4-6' '6-12' 'ABOVE_12']
```

```
data.groupby("PATIENT_VISIT_IDENTIFIER", as_index = False).agg({"ICU":(list), "WINDOW":list}).iloc[[13,14,15,41,0,2]]
```

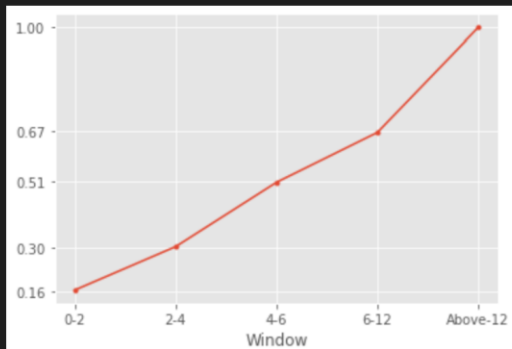
	PATIENT_VISIT_IDENTIFIER	ICU	WINDOW
13	13	[0, 0, 0, 0, 1]	[0-2, 2-4, 4-6, 6-12, ABOVE_12]
14	14	[0, 0, 1, 1, 1]	[0-2, 2-4, 4-6, 6-12, ABOVE_12]
15	15	[0, 0, 0, 0, 1]	[0-2, 2-4, 4-6, 6-12, ABOVE_12]
41	41	[1, 1, 1, 1, 1]	[0-2, 2-4, 4-6, 6-12, ABOVE_12]
0	0	[0, 0, 0, 0, 1]	[0-2, 2-4, 4-6, 6-12, ABOVE_12]
2	2	[0, 0, 0, 0, 1]	[0-2, 2-4, 4-6, 6-12, ABOVE_12]

```
### 0 stands for negative and 1 for positive.
* Patient with ID 13 experienced ICU admission in the first 2 hours from Hospital admission (0-2).
* Patient with ID 14 experienced ICU admission between 2 and 4 hours from Hospital admission (2-4).
* Patient with ID 15 experienced ICU admission between 4 and 6 hours from Hospital admission (4-6).
* Patient with ID 41 experienced ICU admission between 6 and 12 hours from Hospital admission (6-12).
* Patient with ID 0 experienced ICU admission after 12 hours from Hospital admission (ABOVE_12).
* Patient with ID 2 did not experienced ICU admission.
```

```
aux = abs(data.groupby("PATIENT_VISIT_IDENTIFIER")["ICU"].sum()-5)
aux = aux.value_counts().reset_index()
aux.sort_values(by = "index", inplace = True)
aux.reset_index(drop = True, inplace = True)
```

```
tot_icu_inpatients = aux.ICU[0:5].sum()
y = aux.ICU[0:5].cumsum()/tot_icu_inpatients
plt.plot(y, marker = ".")

plt.ylabel
plt.xlabel("Window")
plt.yticks(round(y,2) )
plt.xticks([0,1,2,3,4], ["0-2", "2-4", "4-6", "6-12", "Above-12"])
plt.show()
```



```
# out Of the 249 inpatients 124 $(49\%)$ experienced ICU admission,in which:
```

- * 28 patients in the 0-2 window \$(23\%)\$
- * 14 patients in the 2-4 window \$(28+14 \approx 34\%)\$
- * 20 patients in the 4-6 window \$(28+14+20 \approx 50\%)\$
- * 13 patients in the 6-12 window \$(28+14+20+13 \approx 60\%)\$


```
* 49 patients in the ABOVE_12 window $(28+14+20+13+49 \approx 100\%)$
```

```
* $249 - 124 = 125$ did not experienced ICU admission.
```

```
missing_df = data.groupby("WINDOW").count()/249
```

```
missing_df[vitalSigns_lst]
```

WINDOW	BLOODPRESSURE_DIASTOLIC_MEAN	BLOODPRESSURE_SISTOLIC_MEAN	HEART_RATE_MEAN	RESPIRATORY_RATE_MEAN	TEMPERATURE_MEAN	OXYGEN_SATURATION_M
0-2	0.550201	0.550201	0.534137	0.469880	0.489960	0.51
2-4	0.710843	0.710843	0.706827	0.618474	0.682731	0.69
4-6	0.871486	0.871486	0.867470	0.803213	0.867470	0.85
6-12	1.305221	1.305221	1.329317	1.293173	1.361446	1.36
ABOVE_12	1.542169	1.542169	1.542169	1.542169	1.542169	1.54

rows × 36 columns

Example (BLOODPRESSURE_DIASTOLIC_MEAN). Have at least one vital sign.

- 0-2 = 36%
- 2-4 = 45%
- 4-6 = 53%
- 6-12 = 82%
- Above-12 = 100%

```
df1_styler = missing_df[demo_lst].style.set_table_attributes("style='display:inline'").set_caption('Demographics')
df2_styler = missing_df[comorb_lst].style.set_table_attributes("style='display:inline'").set_caption('Comorbidities')

display_html(df1_styler._repr_html_()+df2_styler._repr_html_(), raw=True)
```

Demographics

WINDOW	AGE_ABOVE65	AGE_PERCENTIL	GENDER
0-2	1.546185	1.546185	1.546185
2-4	1.546185	1.546185	1.546185
4-6	1.546185	1.546185	1.546185
6-12	1.546185	1.546185	1.546185
ABOVE_12	1.546185	1.546185	1.546185

Comorbidities

WINDOW	DISEASE_GROUPING 1	DISEASE_GROUPING 2	DISEASE_GROUPING 3	DISEASE_GROUPING 4	DISEASE_GROUPING 5	DISEASE_GROUPING 6	HTN	IMMUNOCOMPROMISED	OTHER
0-2	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169
2-4	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169
4-6	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169
6-12	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169
ABOVE_12	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169	1.542169

There is no missing values for comorbidities and demographics features

```
#window example
data[data["PATIENT_VISIT_IDENTIFIER"] == 1]
#For instance, the inpatient 1 was ICU addmitted at window "0-2". So, if a
bigger window was selected lines 6 to 9 should **not** be used.
```

