Table of Content

Serial number	**Contents**	**Page number**
1	Abstract	3
2	Introduction	3-5
3	Literature Survey	6-10
4	Proposed Work	10-14
5	Results and Discussion	15-45
6	Conclusion and Future Scope	46-47
7	References	47-48

ABSTRACT

HealthCare domain is one of the largest and prominent domains in the world. Integrating Healthcare system with Technology can create a better healthcare system that can lead to provide better health care services to the people. The foremost objective of any healthcare discipline is Intensive Critical care of patients. In clinical practice, estimates of mortality risk can be useful in triage and resource allocation, in determining appropriate levels of care, and even in discussions with patients and their families around expected outcomes. Since ICUs have limited medical resources, an objective patient triage protocol and evaluation of the severity of a condition must be developed to minimize unnecessary use of ICUs and allow for efficient use of the limited healthcare resources. APACHE IV system provides typically useful and accurate predictions on in-hospital mortality and length of stay for patients in critical care. However, there are factors which may preclude APACHE IV from reaching its heights of predictive accuracy. In order to overcome the drawback of the existing system we have come up with an approach of developing a Smart-End version for Intensive Critical Care Mortality Prediction-an effective and accurate system compared to that of existing APACHE IV prediction system. Developing a better accurate mortality prediction system results in enabling the healthcare providers to identify patients at higher risk of mortality earlier, allowing for earlier interventions and potentially better outcomes.

Keywords: APACHE IV, Healthcare, Intensive Critical Care, Mortality, Smart-End Intensive Critical Care Mortality Prediction.

II. Introduction:

Healthcare sector is a multidisciplinary domain. The global healthcare industry is booming, and it isn't showing any signs of slowing down. Aging populations, a rise in the number of chronic diseases, and more health-conscious societies are driving innovations in technology, public health policy, and education. The Centralised focus of Healthcare Industry is to deliver best care to the patients. Although each sector, each department of healthcare industry has its own objective, at the end each of these objectives are designed in such a way to improve the efficacy in providing better services to the patients.

One of the Crucial sub-domains in Healthcare Industry is Clinical Practice. In Clinical Practice, there is high need of estimating mortality risk of the patients for assessing the condition of the patient well in advance before the case turns into serious and avail all the necessary treatment with required equipments and resources. Since we cannot avail guarantee of available of resources like Intensive Care units and other equipments for the treatment at the neck of the moment it is highly suggestable to estimate the mortality risk of the patient well in advance. Estimates of mortality risk are however based on studying aggregate data from large, heterogeneous groups of patients, and as such their validity in the context of any single patient encounter cannot be assured. Intensive-care units aka ICUs treat the most critically sick patients, which is complicated by the heterogeneity of the diseases that they encounter.

Severity scores based mainly on acute physiology measures are usually considered and collected at ICU admission are used to predict the survival, but are non-specific, and predictions for individual patients can be inaccurate. Prognostic scoring systems, such as the Acute Physiology and Chronic Health Evaluation (APACHE) score, Single Organ Failure Assessment SOFA and Mortality Probability Model (MPM) have been developed to predict ICU Patient mortality analysis.

The primary aim of our project is to determine which variables available within the first 24 hours of a patient's ICU stay may be indicative of the APACHE IV scoring system making occasional but potentially illuminating errors in predicting in-hospital mortality. The goal of this project is to create a model that uses data from the first 24 hours of intensive care to predict patient survival with better prediction probability of Mortality compared to Acute Physiology and Chronic Health Evaluation IV Score model. An approach of diminishing the non-vital APACHE features that reduces the accuracy of APACHE IV score model & developing multi-prediction models using Machine Learning & Deep Learning algorithms and choosing the most optimum Prediction model that has greater efficacy in predicting the Mortality of Critical care unit patients compared to that of APACHE IV Score model.

MOTIVATION FOR THE RESEARCH

The main motivation towards the research on this project is to improve patient outcomes by identifying patients who are at higher risk of mortality and providing more personalised care. The main crucial motivations of this project include improving patient outcomes, Addressing limitation of existing APACHE Mortality prediction system, Advancing the field of Healthcare Analytics beyond mortality prediction system using Machine Learning algorithms and cost saving by reducing unnecessary interventions or treatments for patients who are at lower risk of mortality In addition to improving patient outcomes, a better accurate mortality prediction system could have broader applications in the healthcare field. It could contribute to the development of personalized medicine, by identifying patients who may benefit from specific treatments based on their individual risk factors. Overall the main motto of the research on this project is to negotiate the practice of providing irrelevant interventions or treatment to the patients and take steps in developing the system of providing personalised treatment and diagnosis.

PROBLEM STATEMENT

In General, the initial 24 hours in an intensive care unit ICU are highly crucial cum critical for any patient. On Diagnosing the condition of the patient, the hospital must facilitate highly advanced personalized critical care tailored to respective patient’s needs to enhance the chances of survival. Perhaps there exists several critical challenges in the functioning and delivery of the ICU of a hospital system, which include the demand for sufficient and specialized intensivists - a board-certified physician who provides special care for critically ill patients, technologies, equipments, apparatus, dispensary etc. These challenges directly impact the patient’s survival. Over the past three decades severity scoring systems like **APACHE scoring system, the Simplified Acute Physiology Score** (SAPS), and the **Mortality Probability Model** (MPM), **Single Organ Failure Assessment (SOFA)** have been attempted to be used as a critical care triage criterion. But They often lack generalizability beyond the patients on whom the models were developed, and the models like APACHE scoring system are often used to proprietary, costly to use and suffer from opaque algorithms. Hence there arose the need of identifying and developing an efficient way to address the problems in optimistic way with existing severity of illness systems.

GOAL OF THE PROJECT

The main task of this project is to develop a Smart-End Version for Intensive Critical Care Mortality Prediction which can result in Long Term effective solution for the doctors in order to predict the survival chances of the patient and provide the optimal personalized required treatment and save the patient’s life. The intermediate task of this project includes identifying the top-tier risk factors associated with a high mortality rate, that will assist the medical system in understanding the criticalness of the patient’s condition and take quick appropriate actions such that we can negotiate the non-vital features of existing mortality rate prediction system which is APACHE IV in order to achieve better accuracy in predictions. Hence the better the accurate predictions the system predicts the better the scope of delivering effective treatment services to the patient and save their lives.

III. Literature Survey:

Intensive Care units are the most safest place for treating the patients who are in critical life conditions. These are considered to be the specialised units which holds high end advanced technologies that help in providing best & faster efficient treatment to the patients. The main aim of these Intensive care units is to take immediate high-risk decisions and provide the optimal treatment with efficient technological equipments and improve patient survival rate. Critical Care usually needed immediate high-risk decision making in the situation of uncertainty and patient outcome is related to wide range of factors which includes admission type, age, chronic diseases, acute physiological changes (Immediate response of patient's body to the treatment) etc. Therefore, real-time decision making based on the numerous and various amount of data in the ICUs is one of the most important challenges faced by clinicians. It strongly depends on efficiency of clinicians.

The prediction of patient survival in the Intensive Care Units is highly beneficial in supporting clinical and managerial tasks such as appropriate treatment planning, optimal resource allocation, determining workload, and evaluating the quality of care, and it can play vital role in providing deeper insights into the health status of patients and the ICU management for the clinical staff and managers of this ward.

Predicting hospital mortality for Intensive care unit patients: Timeseries analysis by Aya Awad, Mohamed Bader-El-Den, James McNicholas, et.al. focused on investigating how early hospital mortality can be predicted for intensive care unit patients. The authors conducted a thorough time-series analysis on the performance of different data mining methods during the first 48 h of intensive care unit admission. This research performed experimental investigation on ICU patient data using **DM classification techniques to predict mortality**. Earlier studies have defined early as the first 12 h of admission; others have defined it as 24, 48 or 72 h after admission. These assumptions triggered work done in this research to perform a timeseries analysis for mortality prediction over the first 48 h of ICU admission. **The Data** used for the challenge consisted of five general descriptors, including age, gender, height, ICU type and initial weight. The remaining variables are 36 time series measurements of vital signs and laboratory results from the first 48 h of the first available ICU stay of a patient's admission. Random Forest, PART "The algorithm name" (Partial Decision Trees) & Bayesian Networks (BN) algorithms are the algorithms they have employed. **Result:** The primary outcome was hospital mortality. Performance measures were calculated using cross-validated AUROC to minimize bias. **Limitations:** They have not developed a usable clinical tool in this work, we have shown that there exists rich information signal early in a critical care admission, which can provide guidance about likely individual outcome. We have shown this on a database with incomplete data.

Evaluating ICU Clinical Severity Scoring Systems and Machine Learning Applications: APACHE IV/IVa Case Study by Baran Balkan, Patrick Essay, and Vignesh Subbian had made an attempt to alleviate the general underlying limitations of scoring instruments specifically Acute Physiology and Chronic Health

Evaluation scoring system and demonstrate the utility of readily available medical databases, machine learning techniques were used to evaluate APACHE IV and IVa prediction measures in an **open-source, teleICU research database**. The researchers have constructed three random forest models with ten trees each and used the metrics that they have used to evaluate the APACHE models to examine how the random forests compare to APACHE IV and IVa. This allowed them to compare the predictive capabilities of the existing models with their ensemble method named Random Forest. They have compared the APACHE predictions with the regressors by applying the same metrics, however, a different set of metrics to assess the binary classifier had to be used because the regression task was no longer being examined. **Results: i) Length of Stay:** A decrease in performance was seen from the APACHE IV to the IVa model in predicting length of stay. Random forest model had shown good performance in predicting length of stay, which shows that set of features used by the APACHE model to derive predictions is insufficient for predicting length of stay. **ii) Mortality Prediction:** With respect to Mortality Prediction **APACHE IVa** exhibited higher performance metrics than the IV model, **Random Forest regression model** exhibits a drastic increase in mortality prediction scores for hospital mortality and perfect mortality prediction scores for the ICU. **Limitation:** Though they have observed improvements in predictive power in the random forest models compared to that of the APACHE models, the clinical significance of these findings is yet limited by the coarse-grained nature of their analysis.

The authors **Shuo Feng & Joel A. Dubin** have published an article titled **Identifying early-measured variables associated with APACHE IVa providing incorrect in-hospital mortality predictions for critical care patients** in which they have aimed to quantify both the number of false positive and false negatives of APACHE IVa in a large heterogeneous multi-hospital ICU database collected from over 200 hospitals in the United States. In addition, they have presented models that shows which features collected within the first 24 hrs of the ICU stay are associated with each of these two types of APACHE prediction error. That is, they have identified variables early in an ICU stay that might be predictive of APACHE IVa not doing as well in its outcome prediction of in-hospital mortality as expected. The **goal of this paper** was to use a **large heterogeneous multi-hospital ICU study** to attempt to identify some predictors of patient information collected in the first 24 h that might be predictive of when the APACHE prediction may go wrong regarding its speculation on in-hospital mortality. As there are two ways of APACHE leading to prediction errors, with these predictions likely leading to different approaches to patient care, they separated the two errors as: (i) **Type I errors**, or false positives (i.e., **APACHE predicting in-hospital mortality, but the patient is discharged alive**), and (ii) **Type II error**, or false negatives (i.e., **APACHE predicting being discharged alive, but the patient dies at some point during their hospital stay**). **Model & Methodology:** They have used an out-of-sample validation approach, following a model-building step using cross-validation, to model these two errors separately with multiple logistic regression with Lasso penalization. **Result:** The paper have determined the variables available within the first 24 h of a patient's ICU stay that may be indicative of the APACHE IVa scoring system's occasional prediction errors, through logistic regression modelling with Lasso penalization. **Limitations:** In this paper, the researchers wanted to directly investigate variables associated with APACHE's occasional failure for predicting in-hospital mortality. The key issue for this latter approach is then determining the appropriate APACHE prediction probability cut-off value for determining if APACHE is deciding death or survival, and this cut-off choice might need to be part of a tuning process for other datasets.

Survival prediction in intensive-care units based on aggregation of long-term disease history and acute physiology: a retrospective study of the Danish National Patient Registry and electronic patient records by Annelaura B Nielsen, Hans-Christian Thorsen-Meyer, Kirstine Belling, et.al. This paper predicted the patients survival time who were monitored in the ICU which treats most critical patients, each patient has different diseases but still this paper predicts every different diseased patient survival time. Here they have taken the big data, which is of the size 230000, which has the history of 1month, 3 months, 6 months, 1 year, 2.5 years, 5 years, 7.5 years, 10 years and 23 years before ICU admission. It is used for the neural network to predict the survival time. Firstly, data is splitted into training and testing data then they have used a fivefold cross validation technique to avoid overfitting of the model so that it can predict the accurate survival lifetime. Then they trained the neural network by using the training dataset which was splitted before. **Observations:** They found the morality prediction with the model based solely on disease history outperformed the Multimorbidity index. **Conclusion:** They concluded that disease spectrum wide data available before ICU admission were useful for modality prediction. To understand the patient long term and short-term events they have concluded that one can use the machine learning model which can interact with each other. Explainable machine learning models are critical in clinical settings, and the findings of this paper highlights how to proceed toward transforming sophisticated models into usable, transparent, and trustworthy therapeutic tools.

The Ability of the Acute Physiology and Chronic Health Evaluation (APACHE) IV Score to Predict Mortality in a Single Tertiary Hospital by Jae Woo Choi, Young Sun Park, Young Seok Lee, et.al. in this study also investigated the suitability of APACHE IV severity scores and MPMs in an ICU within a tertiary general hospital, by analysing the relationships among APACHE IV scores at the time of the admission, the predicted mortality rate, and the actual mortality rate comparing with APACHE II model. This study also verified the usefulness of the APACHE IV score as a standard triage protocol for admission in the ICU. **Observed Differences between APACHE II & APACHE IV Model:** APACHE II score is calculated based on 12 physiologic criteria and estimates risk based on data available within the first 24 hours of an ICU stay. APACHE IV was designed to assess the severity of illness as well as the prognosis in the ICU and has 17 physiological criteria, adding new variables such as mechanical ventilation and disease-specific subgroups etc to the existing APACHE III variables. Disease-specific scoring systems have been developed for several important subgroups treated in the ICU since an APACHE III model. **Methodologies: Cox proportional hazards regression was conducted to examine associations with death after adjustment for the APACHE IV score.** Hazard ratios (HRs) were used to quantify the relationship between risk factors and death. **Goal:** An ICU triage model, which predicts hospital mortality, was constructed by combining the APACHE IV score and the other risk factors identified above the Cox regression models. **Results:** The APACHE IV scoring system exhibited satisfactory discrimination and excellent calibration, but the result supposed that it was **not appropriate to be used as a single criterion for ICU admission.**

The research paper titled **"Survival Analysis of Elderly Patients in Intensive Care Units"** by the Authors Diego BonfadaMarquiony Marques dos SantosKenio Costa Lima had conducted survival analysis of elderly patients hospitalized in intensive care units by identifying the important predictors of mortality among that age group. A retrospective cohort study was performed with data from the medical records of 457 elderly patients hospitalized in an ICU located in the city of Natal in Brazil. Survival functions were estimated using the Kaplan-Meier estimator, and the Log-rank test was used for comparisons. In addition, a multiple Cox proportional hazards model was constructed to identify the independent effects of the predictors of survival. **Result:** It was found that the survival of elderly ICU patients declined due to factors such as increased hospitalization time, the presence of shock, pneumonia, septicaemia, fractures, a reduced state of consciousness, hospitalization for clinical reasons, being bedridden prior to hospitalization, fever, bradycardia, hypotension, cardiac arrest and the need for mechanical ventilation. The **multiple Cox proportional hazards model** revealed that variables such as shock, longevity, bradycardia, fractures, hospitalization in the public healthcare system and admission for clinical reasons remained significant as predictors of reduced survival in intensive care units. The have **concluded** that any initiative aimed at increasing survival of elderly ICU patients must look at individual & social issues and other factors related to healthcare network/domain.

After a detailed literature survey performed in order to know about Models for predicting Patient Survival Analysis in Intensive Care Units, we have come to a conclusion that although priorly many models were developed among which APACHE (Acute Physiology and Chronic Health Evaluation) System had been best performing model but it led to various limitations/errors which in-toto resulted in decreasing the efficacy & efficiency of the model's performance. These are often costly to use & suffer from opaque algorithms.

III. Proposed Work

About the Dataset:

Data for this project is taken from the initiative of MIT's GOSSIS community. Dataset of more than 90,000 hospital Intensive Care Unit (ICU) visits from patients, spanning a one-year timeframe. This data is part of a growing global effort and consortium spanning Argentina, Australia, New Zealand, Sri Lanka, Brazil, and more than 200 hospitals in the United States

The chosen dataset aims on determining patient outcomes based on data taken during their first 24 hours in the ICU. The response variable, hospital_death, is binary with a value of 1 indicating a patient's death and a value of 0 corresponding to a patient's survival.

Some of the data that is gathered pertains to a person's identity, demographics, vitals and labs from when they enter the ICU, APACHE covariates, APACHE comorbidities, and APACHE predictions and groupings. APACHE is meant to help with benchmarking especially for accurately predicting mortality.

Non-Sequential Features: -

Demographic feature includes variables such as "age", "bmi", "ethnicity" [Nationality], "gender", "height", "weight", "hospital_admit_source" [The location of the patient prior to being admitted to the hospital], ICU related features like "icu_admit_source" [The location of the patient prior to being admitted to the unit], "icu_stay_type" [readmit, transfer/admit], "icu_type" [A classification which indicates the type of care the unit is capable of providing] & "pre_icu_los_days" [The length of stay of patient between hospital admission & unit admission in days].

Prediction feature includes two variables "apache_4a_hospital_death_prob" & "apache_4a_icu_death_prob". Apache_4a_hospital_death_prob variable tells Probabilistic prediction of in-hospital Mortality given by APACHE 4a Prediction

System & “apache_4a_icu_death_prob” tells Probabilistic prediction of ICU Mortality given by APACHE 4a Prediction System.

Co-morbidity features are considered in APACHE II Prediction System & APACHE III-J Prediction System. APACHE II considered 6 co-morbidities whereas APACHE III considered 7 co-morbidities. Co-morbidities included the recors if person suffered with any of the chronic diseases among aids, cirrhosis, diabetes_mellitus, hepatic_failure, immunosuppression, leukemia, lymphoma & solid_tumor with metastasis. Co-morbidity feature contains the information if the person admitted in ICU had any of the chronic diseases.

Different Diagnoses have been considered by different APACHE prediction systems. APACHE_3j_bodysystem features include the list of diagnoses like cardiovascular, neurological, Sepsis, Respiratory, Gastrointestinal, Metabolic, Trauma, Genitourinary, Musculoskeletal/Skin, Haematological & Gynaecological diagnoses. Whereas APACHE_2j_bodysystem included all the above diagnoses except Sepsis & it contains another diagnosis apart from the diagnoses list of APACHE_3j_bodysystem which is termed as undefined diagnosis.

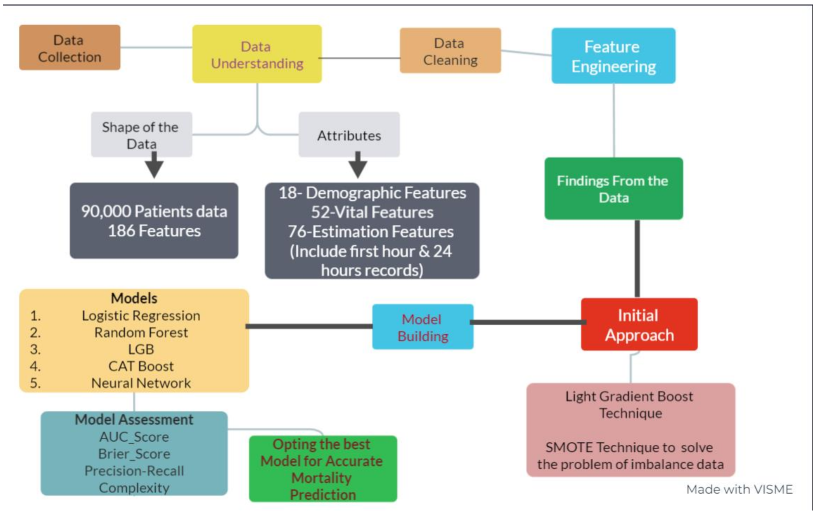
Sequential Features: -

LABS categorized features contains the variables that are recorded from laboratory test results. It contains around 15 lab tests of which both max value & min Value are recorded on basis of 1hour & 24 hours after admitting in ICU. The 15 lab test variables include albumin, bilirubin, bun, calcium, creatinine, glucose, bicarbonate (hco3), hemoglobin, hematocrit, inr, lactate, platelets, potassium, sodium & White Blood Cells (WBC) The notation **d1** represents recording per day (**24 hrs.**) & **h1** represent recording **per hour.**

VITALS categorized features include the variables of vital signs that include Temperature, Respiratory rate, Heart Rate, spo2, diasbp [Average Diastolic Blood Pressure], mbp (**Mean Blood Pressure** = diasbp +1/3[sysbp-diasbp]), sysbp [Systolic Blood Pressure]. Amon these vital features each measurement is classified into Invasive measurement method & Non-Invasive Measurement Method. Measurement of intracranial pressure (ICP) can be invaluable in the management of critically ill patients. **Invasive methods** remain the most accurate at measuring ICP, but they are prone to a variety of complications including infection, hemorrhage and neurological deficits. **Noninvasive methods** for measuring and evaluating ICP have been developed and classified in five broad categories but have not been reliable enough to use on a routine basis.

Other APACHE Features which led to increase in APACHE III Score. The Features include variables like **"gcs_unable_apache"** [Whether the Glasgow Coma Scale was unable to be assessed due to patient sedation], **"eye_apache"** [Assessing the condition of eye movement of the patient based on Glasgow coma Scale], **"Verbal_apache"** [Assessing the voice of the patient based on GCS], **"Motor_apache"** [Assessing the body response & movements to the commands based on GCS], **"Intubated_apache"** [Whether the patient was intubated at the time of the highest scoring arterial blood gas used in the oxygenation score], **"ventilated_apache"** [Whether the patient was invasively ventilated at the time of the highest scoring arterial blood gas using the oxygenation scoring algorithm, including any mode of positive pressure ventilation delivered through a circuit attached to an endo-tracheal tube or tracheostomy], **"arf_apache"** [Whether the patient had acute renal failure during the first 24 hours of their unit stay, defined as a 24 hour urine output <410ml, creatinine >=133 micromol/L and no chronic dialysis], "**PostOperative_apache", "Urine_Output"** [The total urine output for the first 24 hours], **"fio2_apache"** [The fraction of inspired oxygen from the arterial blood gas taken during the first 24 hours of unit admission which produces the highest APACHE III score for oxygenation], **"map_apache"** [The mean arterial pressure measured during the first 24 hours which results in the highest APACHE III score].

WORKFLOW:



Challenges of this Project:

Our Project has too many challenges which include.

- Minimum Domain Knowledge,
- High Dimensional data,
- Plenty of Missing Variables which are hard to impute &

- Highly Imbalanced dataset.

Proposed Methodology:

In order to have better understanding of the dataset using feature importance & Shapley Explanator and deal with missing values we have come up with idea of implementing **Baseline Model**: "**Light Gradient Boost"** & apply **SMOTE Technique** (Synthetic Minority oversampling technique) to solve the imbalance problem by randomly increasing the minority class examples by replicating them.

Why Light Gradient Boost Preferred over XG Boost for Baseline Model:

For our project LGB is more specifically preferred compared to XG Boost because LG Boost is

- LGB is highly compatible for categorical variable: binning continuous variable to discrete variable based on histogram.
- Filter out the data instances for finding a split value; can reduce more loss than the level wise algorithm (XGBoost & CATBoost, resulting in much better accuracy which can rarely be achieved by any of the existing boosting algorithms.
- Compatibility with large data set
- Reduce significant training time as compared to XGBOOST.

PROPOSED MODELS:

i) Logistic Regression - Binning, Imputing, PCA

ii) RF-Binning

iii) LGBoost-Imputing,

iv) CATBoost

v) Neural Network With PCA

DATA ANALYTICS MODELS:

A. Merits and Demerits of the Models:

1. **Logistic Regression: -** Logistic regression is a popular statistical technique used for modelling the relationship between a binary response variable (0/1) and one or more predictor variables.

Merits of Logistic Regression: *It is a simple and easy-to-understand method that requires few assumptions about the data and can be implemented quickly. It is a powerful tool for analyzing and predicting binary outcomes which can be interpreted easily and used for decision making.*

Demerits of Logistic Regression: *It can be sensitive to outliers, and the results may be biased if the data is not normally distributed. Logistic regression assumes that the predictors are independent, which may not be true in some cases. It can be difficult to interpret the coefficients, especially if the predictors are highly correlated.*

2. **Random Forest: -** Random Forest is a popular machine learning algorithm that is used for classification, regression, and feature selection.

   Merits of Random Forest: *Since the algorithm works by combining the results of multiple decision trees, and therefore, it is less likely to be affected by outliers. It is robust to noise and outliers in the data. It is a non-parametric algorithm which means it does not make any assumption about underlying distribution of data.*

   Demerits of Random Forest: *It is Computationally expensive. It is difficult to understand hoe the algorithm is making predictions. Random Forest is less prone to overfitting. Random Forest may not perform well on imbalanced datasets. Due to complexity of algorithm, Random Forest may not be easily interpretable.*

3. **LGBoost:** LightGBM uses gradient-based techniques to build an ensemble of decision trees, which can be used for both regression and classification tasks.

   Merits of LGBoost: *It is designed to be highly efficient and can handle large datasets quickly. It can scale to millions of samples & features & supports distributed computing. LGBoost has shown to perform well in a variety of benchmarks.*

   Demerits of LGBoost: *LGBoost can still consume a significant amount of memory, especially when dealing with large datasets. LightGBM can be difficult to interpret and explain especially when dealing with complex models. It requires significant tuning to achieve optimal performance.*

4. **CAT Boost: -** CAT Boost is a machine learning algorithm that is specifically designed to handle categorical features. It is an extension of gradient boosting that uses a novel algorithm to handle categorical variables.

Merits of CAT Boost: *One of the main advantages of CAT Boost is its ability to handle categorical variables natively without the need for one-hot encoding or other pre-processing techniques. It is known for its high accuracy and has been shown to outperform other popular algorithms like XGBoost and Light GBM in certain scenarios. It is also known for its fast-training speed.*

Demerits of CAT Boost: *CAT Boost can be computationally expensive, especially for large datasets. It is susceptible to overfitting if not properly tuned. The algorithm has several hyperparameters that need to be tuned for optimal performance.*

5. **Neural Network: -** Neural networks are a type of machine learning model that are designed to recognize patterns and relationships in complex data. They have become increasingly popular in recent years due to their ability to perform tasks such as image recognition, natural language processing, and speech recognition.

   Merits of Neural Network: *Neural networks can learn from large amounts of data and can automatically adapt their model to new information, making them ideal for tasks such as classification, prediction, and decision-making. Robust to noise and can handle incomplete/missing data making them useful for real-world applications. Neural networks can model non-linear relationships between variables, perform parallel processing and can generalise from training data.*

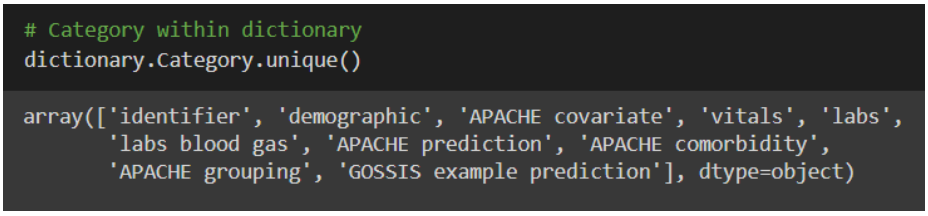
   Demerits of Neural Network: *Neural networks can overfit the training data, meaning they may become too complex and perform poorly on new data that they have not seen before. It si often considered black box models because they can be difficult to interpret, making it hard to understand why a certain decision was made. It can lack Transparency in terms of how they arrive at their decisions. Can be vulnerable to adversarial attacks,*

IV. Results and Discussions

Exploratory Data Analytics & The Approach

Part A: - Data Understanding

- ***Overviewing the list of categories present in dataset:***

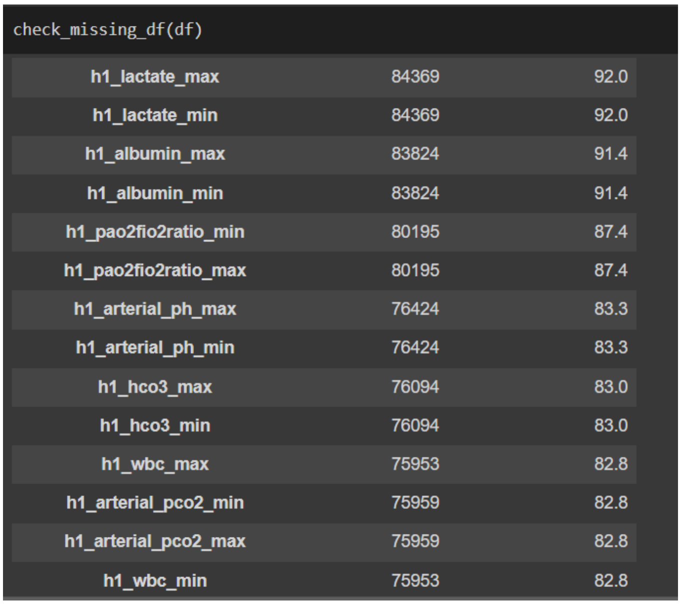


```
# Category within dictionary
dictionary.Category.unique()
```

```
array(['identifier', 'demographic', 'APACHE covariate', 'vitals', 'labs',
       'labs blood gas', 'APACHE prediction', 'APACHE comorbidity',
       'APACHE grouping', 'GOSSIS example prediction'], dtype=object)
```

We can see that the dataset contains the variables which are categorized into 10 categories that are displayed above.

- ***Checking Missing Values:***



```
check_missing_df(df)
```

h1_lactate_max	84369	92.0
h1_lactate_min	84369	92.0
h1_albumin_max	83824	91.4
h1_albumin_min	83824	91.4
h1_pao2fio2ratio_min	80195	87.4
h1_pao2fio2ratio_max	80195	87.4
h1_arterial_ph_max	76424	83.3
h1_arterial_ph_min	76424	83.3
h1_hco3_max	76094	83.0
h1_hco3_min	76094	83.0
h1_wbc_max	75953	82.8
h1_arterial_pco2_min	75959	82.8
h1_arterial_pco2_max	75959	82.8
h1_wbc_min	75953	82.8

<u>Category wise Missing Values Interpretation</u>

Observations

a. Demographics - hospital_admission_source attribute has higher percentage of missing values. "bmi, heaight & weight" attributes are often missing together.

b. Vitals - Any attribute with invasive measurement have lot of missing values.

c. Labs - Most of the data recorded within 1 hour od admission are missing. The attributes "inr,lactate,albumin, and bilirubin", a lot of missing values, even using 24hours results

d. Labs Blood Gas - Most of the attributes are missing, so we need to know the feature importance of the attributes belonging to category.

e. APACHE Covariate - "albumin_apche, bilirubin_apache, fio2_apache, paco2_apache, paco2_for_ph_apache, pao2_apache, ph_apache & urineoutput_apache" attributes have high % of missing values.

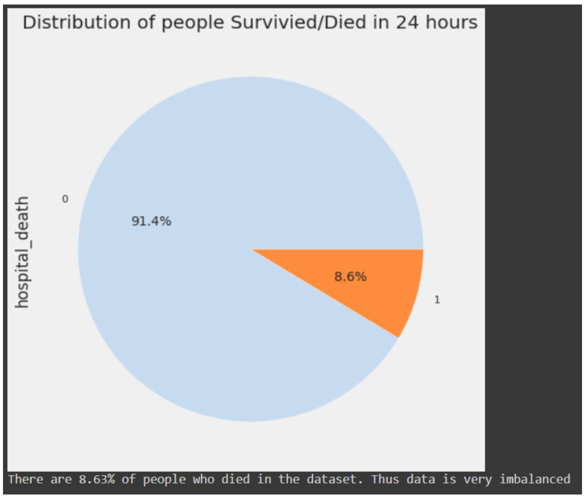
f. **APACHE Comorbidity** - Very Few Missing Values are present, hence they can be imputed.

g. **APACE Prediction** - If one of the 2 features has missing values, the other also has missing values

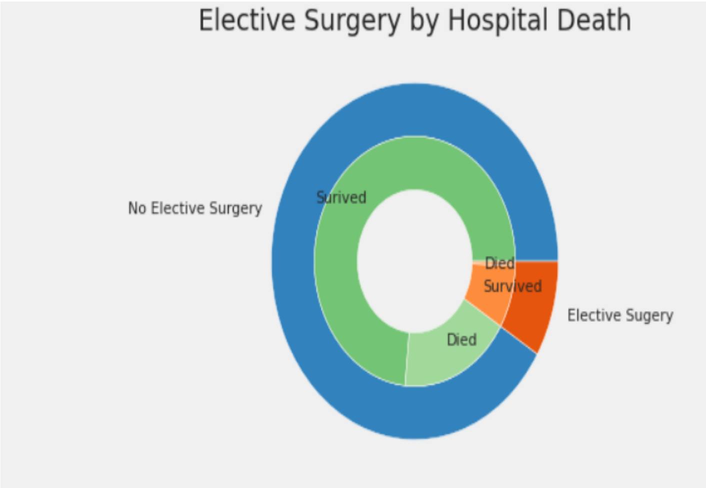
h. **APACHE Grouping** - apache2_bodysystem and apache3_bodysystem are missing together. We need to check the difference between these two attributes & analyse are both required in modelling or any one is sufficient.

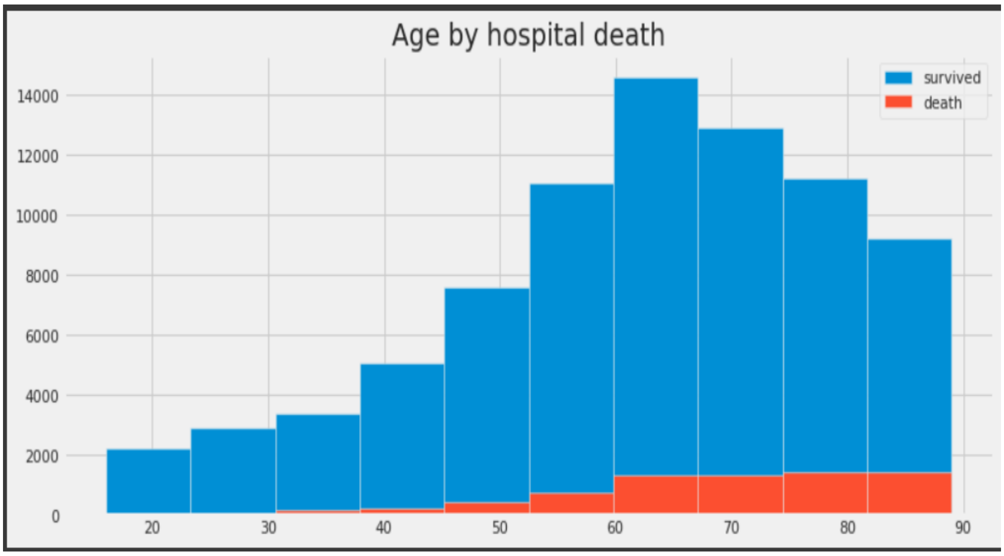
• Initial Findings: -

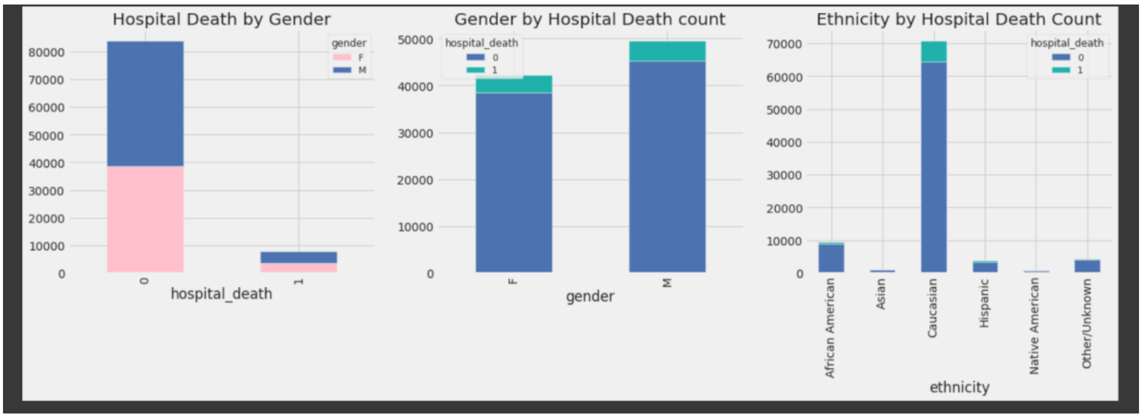
1) Demographic Features distribution with Hospital Death which contains Age, Gender, Ethnicity.

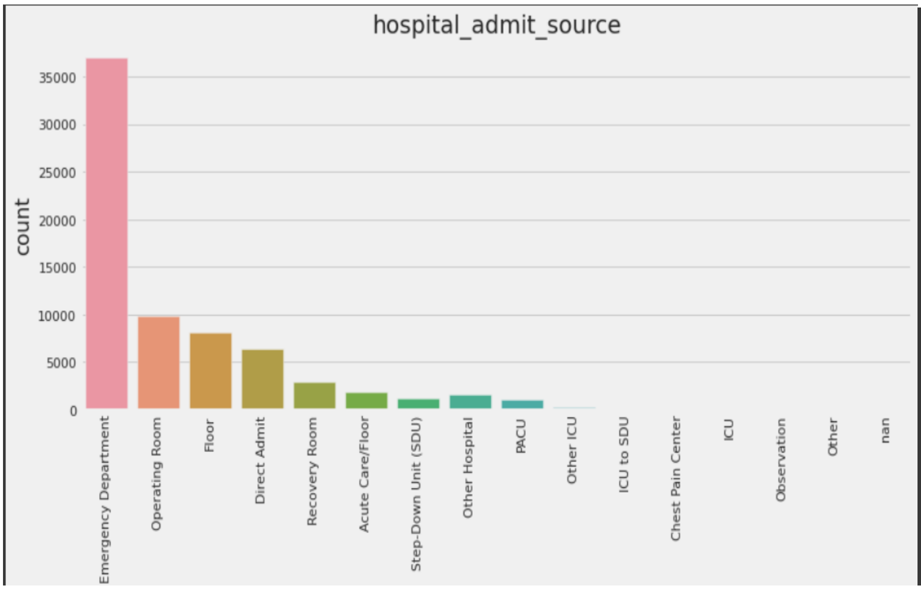


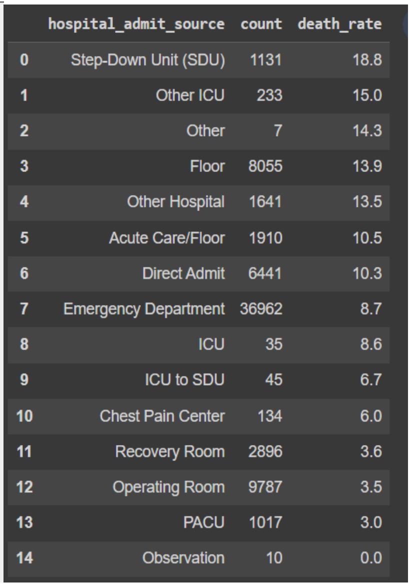
```
There are 8.63% of people who died in the dataset. Thus data is very imbalanced
```



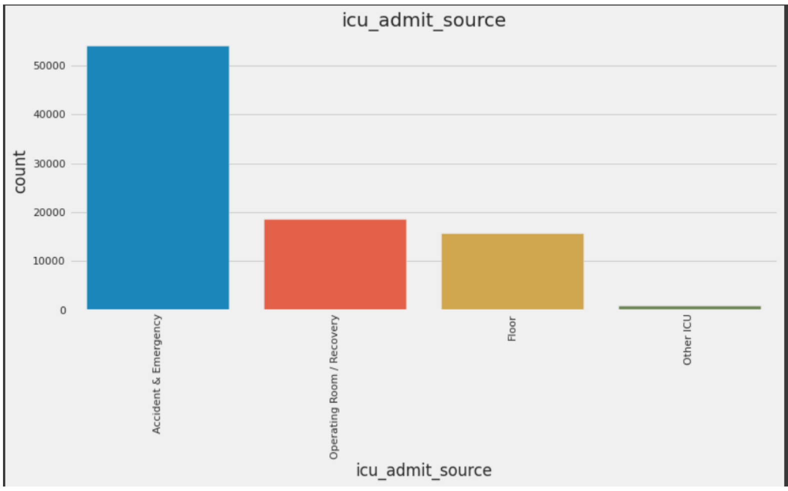


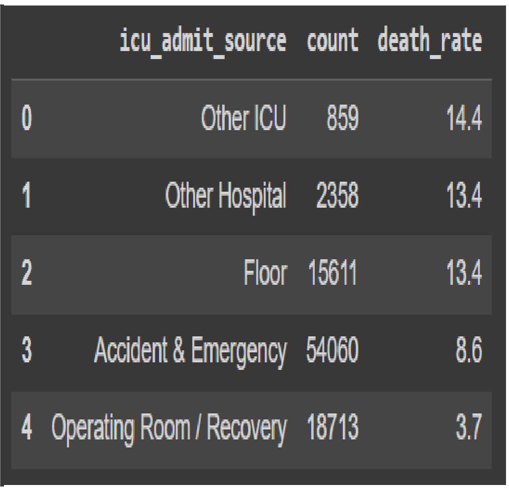






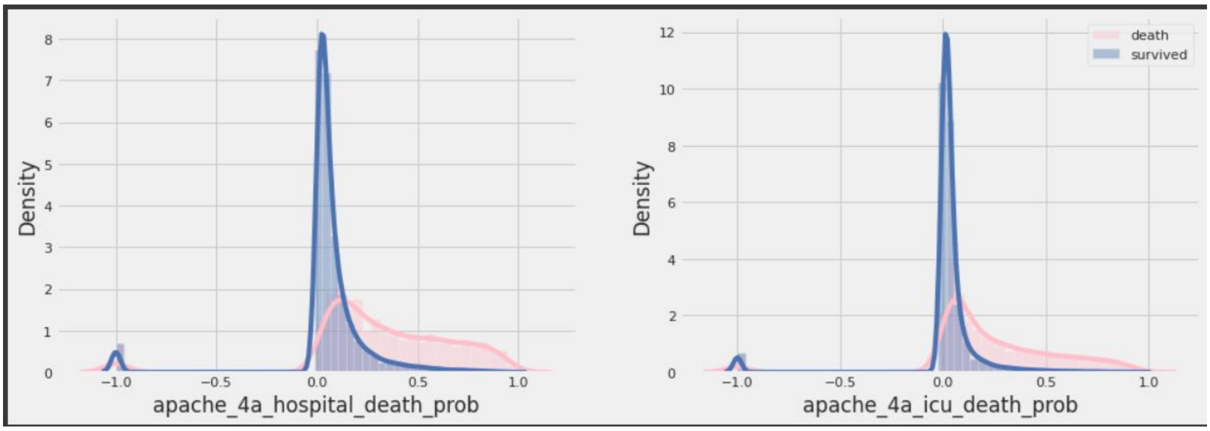
	hospital_admit_source	count	death_rate
0	Step-Down Unit (SDU)	1131	18.8
1	Other ICU	233	15.0
2	Other	7	14.3
3	Floor	8055	13.9
4	Other Hospital	1641	13.5
5	Acute Care/Floor	1910	10.5
6	Direct Admit	6441	10.3
7	Emergency Department	36962	8.7
8	ICU	35	8.6
9	ICU to SDU	45	6.7
10	Chest Pain Center	134	6.0
11	Recovery Room	2896	3.6
12	Operating Room	9787	3.5
13	PACU	1017	3.0
14	Observation	10	0.0



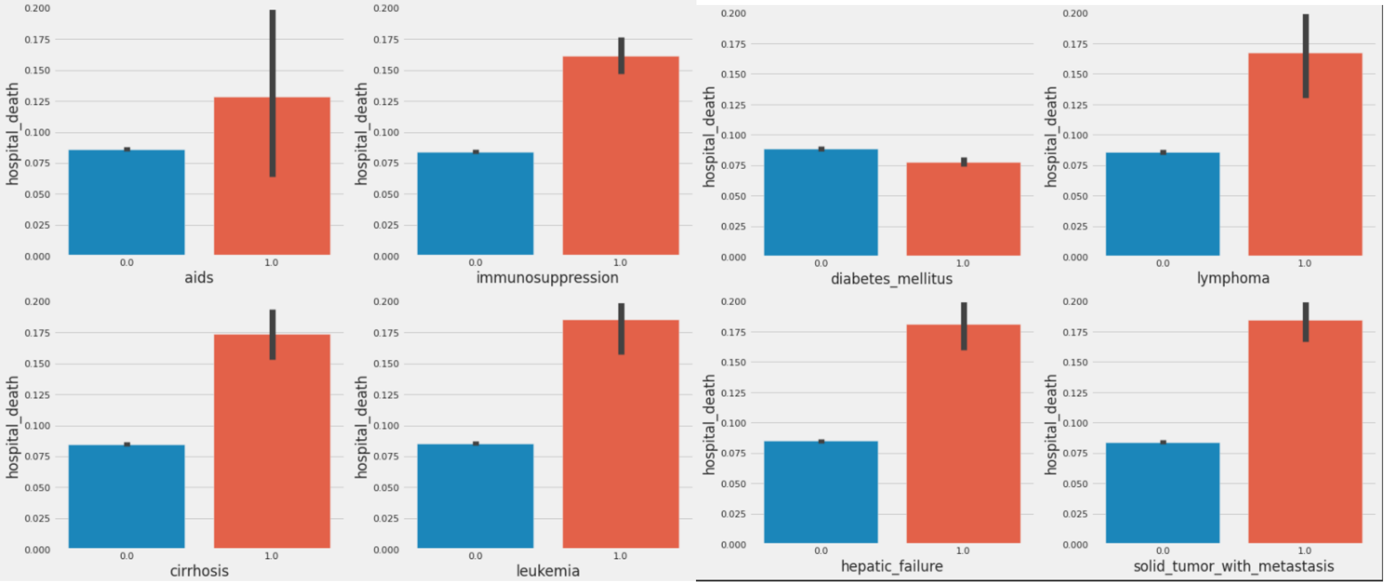


	icu_admit_source	count	death_rate
0	Other ICU	859	14.4
1	Other Hospital	2358	13.4
2	Floor	15611	13.4
3	Accident & Emergency	54060	8.6
4	Operating Room / Recovery	18713	3.7

2) APACHE Prediction Findings



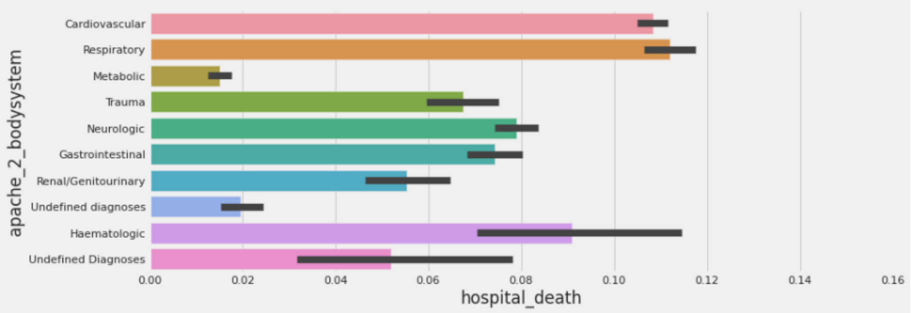
3) APACHE Co-Morbidity: -

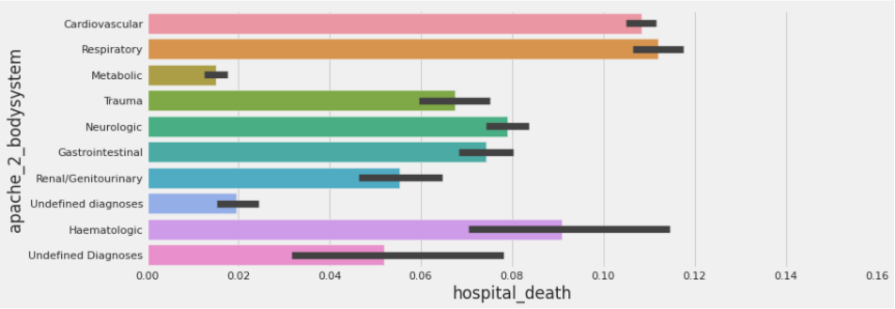


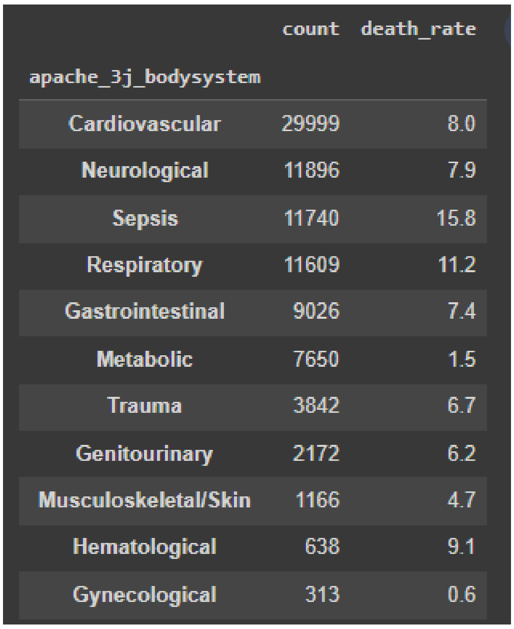
KEY FINDINDS:

- 31% of patients had at least 1 chronic disease. Diabetes accounted for the highest proportion, followed by hepatic_failure
- Patients with 1 of the chronic disease (aids, cirrhosis, hepatic_failure, luekemia, lymphoma, solid_tumor, immunosuppression) are likely to have higher chance of death as compared to those without such diseases. However, it is not the case for diabetes_melitus (equal probability of death)
- The result suggests no correlation between the number of chronic diseases and hospital death.

4) APACHE Grouping: -



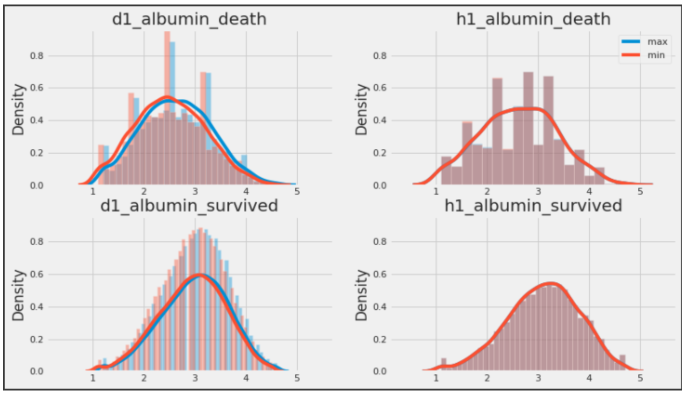




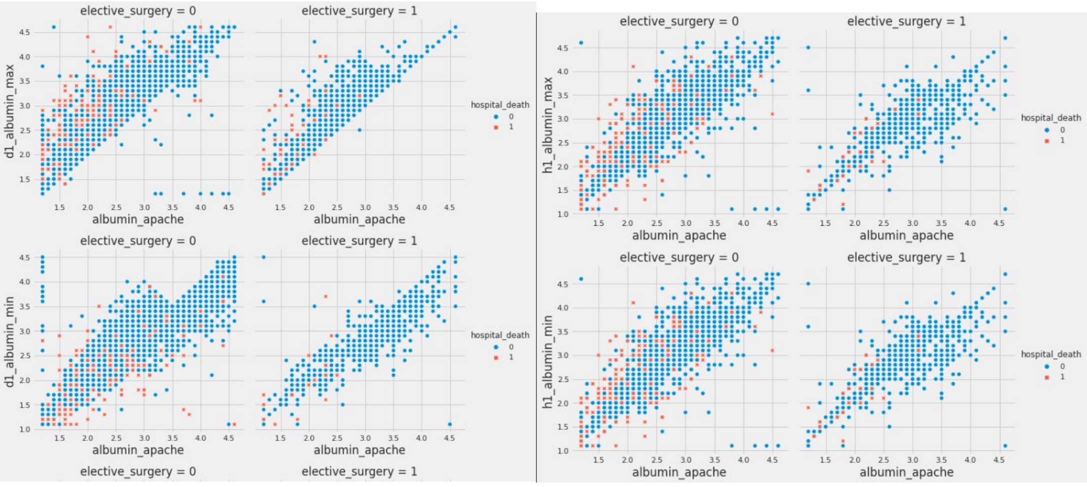
apache_3j_bodysystem	count	death_rate
Cardiovascular	29999	8.0
Neurological	11896	7.9
Sepsis	11740	15.8
Respiratory	11609	11.2
Gastrointestinal	9026	7.4
Metabolic	7650	1.5
Trauma	3842	6.7
Genitourinary	2172	6.2
Musculoskeletal/Skin	1166	4.7
Hematological	638	9.1
Gynecological	313	0.6

5) LABS: -

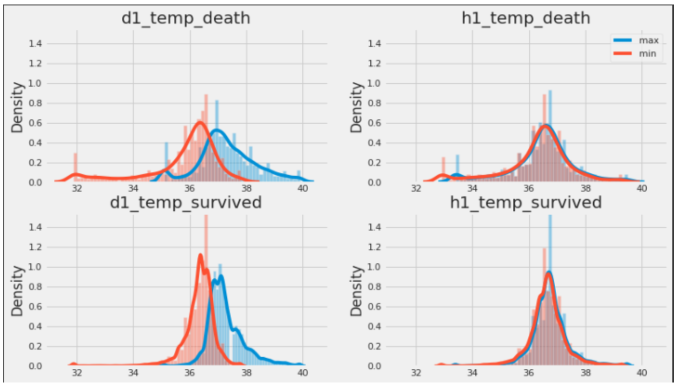
➔ ***Mortality Prediction based on hourly basis & daily basis Albumin value reported by lab.***



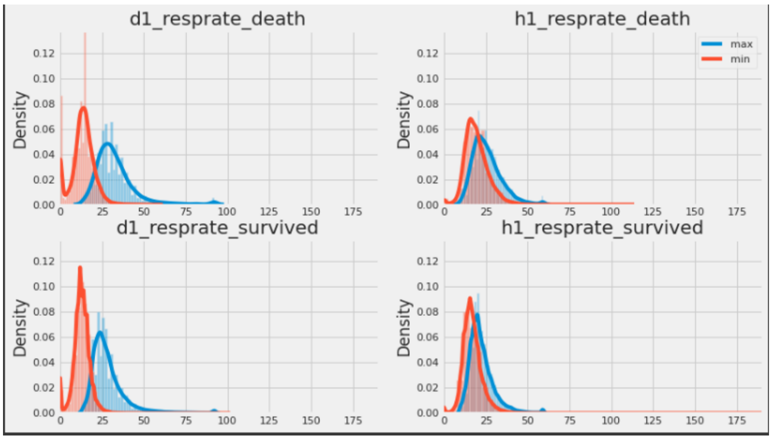
- ***Mortality Prediction given by APACHE III Scoring System based on Albumin levels recorded.***



6) *Vitals: -*



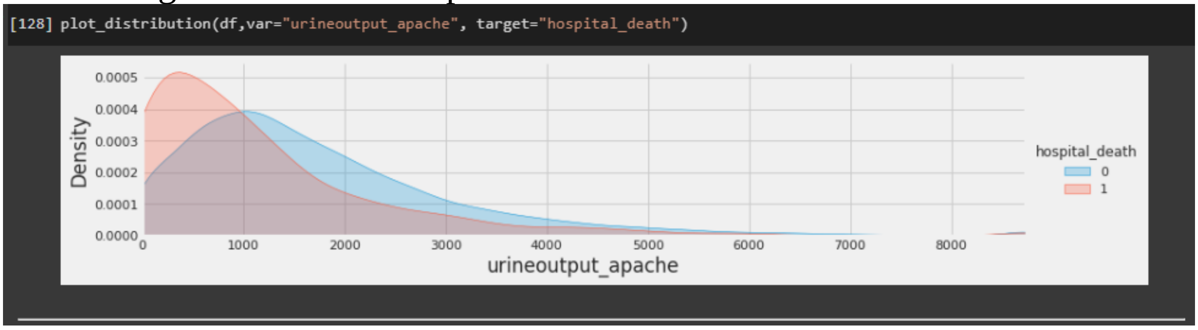
***Inference** - Higher chance of death for patients with temp under 34 in 1st 24hrs*



Inference **-** Higher prob of death when respiration rate min = 0. However, this is not the case for patients who might have aspiration pneumonia or similar disease. This may explain why some patients with resprate_min = 0 but they still alive!

7) Other APACHE Features: -

There exist various other APACHE features based on which death rate is predicted. One among them is Urine output.



```
[128] plot_distribution(df,var="urineoutput_apache", target="hospital_death")
```

Inference: Patients with urine output less than 500 have higher prob of death

Similarly for all the other features data visualization & understanding has been performed.

Baseline Model & Feature Importance: -

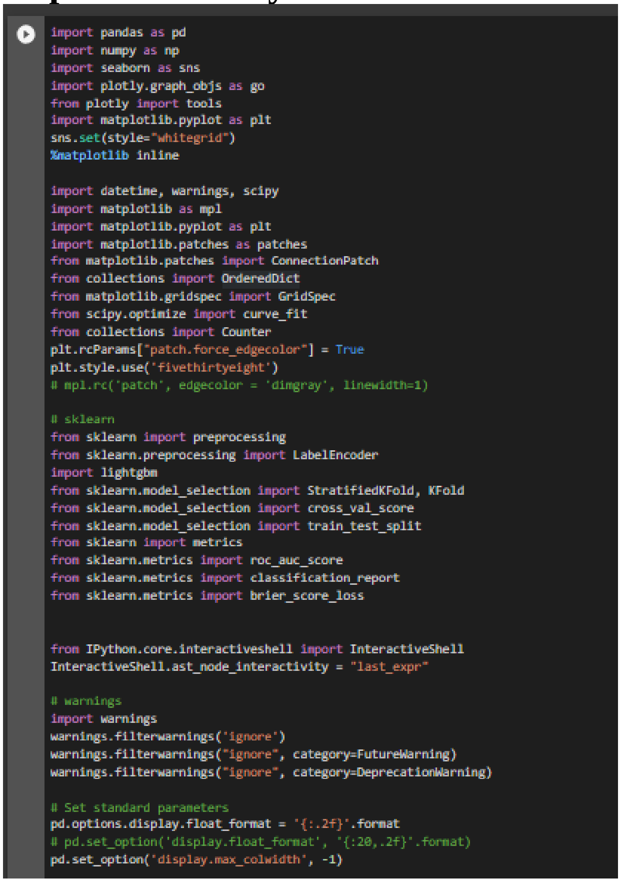
- Before Data Imputation & Binning, we will run the baseline model to have better data understanding & use **“Light Gradient Boosting”:** to deal with missing values & Categorical Variables.

BASELINE MODEL - Light gradient boost

- To have a better understanding of the dataset - using feature importance & Shapley Explanator
- Can deal with missing data.
- Oversampling – SMOTE to address imbalanced data problem.

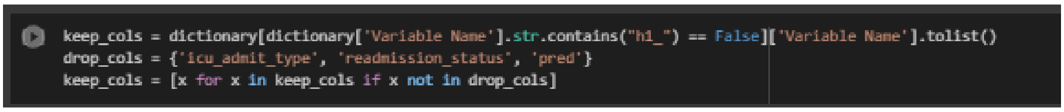
- Initial Step is to Clean the data, remove unnecessary columns that create noise to the output prediction, next Split the data, check data distribution of train & test, Perform Label Encoding, Compute ROC_AUC score from which finally plot feature importance.

Import Necessary Libraries:



```
import pandas as pd
import numpy as np
import seaborn as sns
import plotly.graph_objs as go
from plotly import tools
import matplotlib.pyplot as plt
sns.set(style="whitegrid")
%matplotlib inline

import datetime, warnings, scipy
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from matplotlib.patches import ConnectionPatch
from collections import OrderedDict
from matplotlib.gridspec import GridSpec
from scipy.optimize import curve_fit
from collections import Counter
plt.rcParams["patch.force_edgecolor"] = True
plt.style.use('fivethirtyeight')
# mpl.rc('patch', edgecolor = 'dimgray', linewidth=1)

# sklearn
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder
import lightgbm
from sklearn.model_selection import StratifiedKFold, KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import roc_auc_score
from sklearn.metrics import classification_report
from sklearn.metrics import brier_score_loss


from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "last_expr"

# warnings
import warnings
warnings.filterwarnings('ignore')
warnings.filterwarnings("ignore", category=FutureWarning)
warnings.filterwarnings("ignore", category=DeprecationWarning)

# Set standard parameters
pd.options.display.float_format = '{:.2f}'.format
# pd.set_option('display.float_format', '{:20,.2f}'.format)
pd.set_option('display.max_colwidth', -1)
```

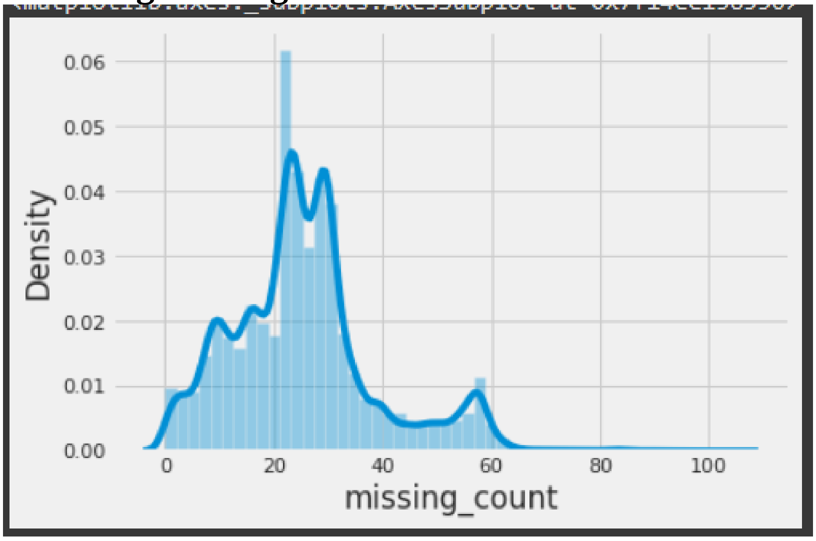
Drop Unnecessary Columns:



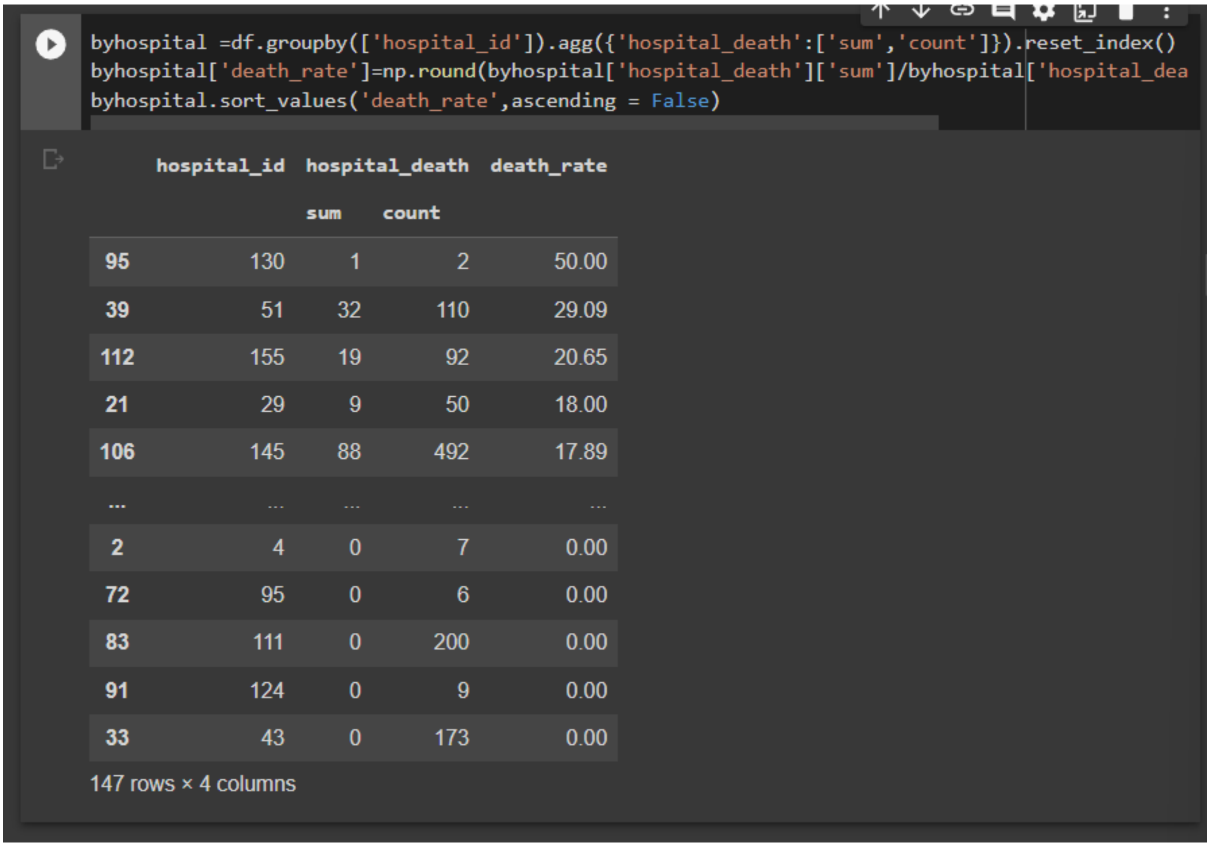
```
keep_cols = dictionary[dictionary['Variable Name'].str.contains("h1_") == False]['Variable Name'].tolist()
drop_cols = {'icu_admit_type', 'readmission_status', 'pred'}
keep_cols = [x for x in keep_cols if x not in drop_cols]
```

Now Store the clean data after removing unnecessary columns in df_clean & store it to a csv file.

Checking Missing Count in clean data frame.



Analyzing Hospital_ID Column to check how important the feature is to fit in the models.



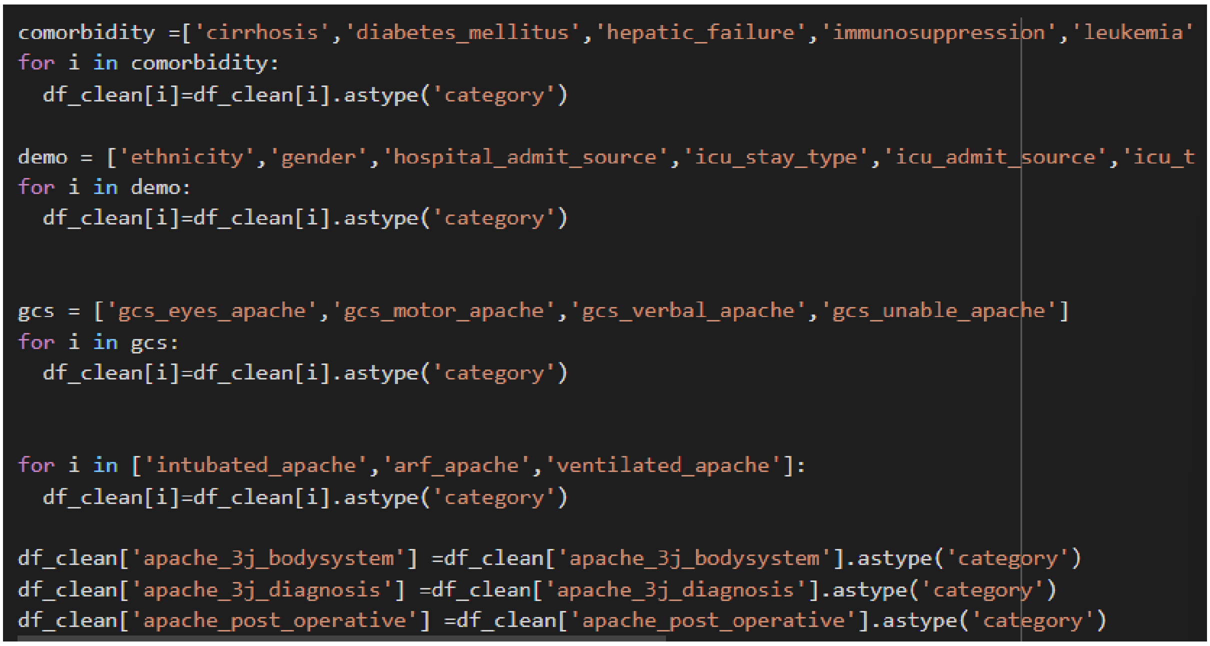
```
byhospital =df.groupby(['hospital_id']).agg({'hospital_death':['sum','count']}).reset_index()
byhospital['death_rate']=np.round(byhospital['hospital_death']['sum']/byhospital['hospital_dea
byhospital.sort_values('death_rate',ascending = False)
```

	hospital_id	hospital_death		death_rate
		sum	count	
95	130	1	2	50.00
39	51	32	110	29.09
112	155	19	92	20.65
21	29	9	50	18.00
106	145	88	492	17.89
...	...	...	...	...
2	4	0	7	0.00
72	95	0	6	0.00
83	111	0	200	0.00
91	124	0	9	0.00
33	43	0	173	0.00

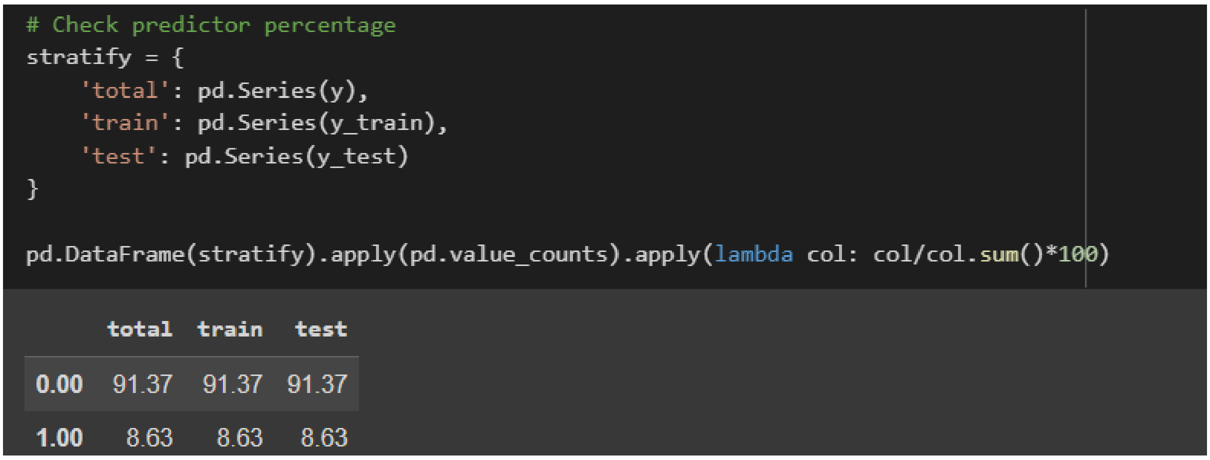
147 rows × 4 columns

- **We can see that some hospitals only have less than 10 patients >> make noise for the prediction if we keep this feature.**

Split Data & Check Distribution of Train vs Test:

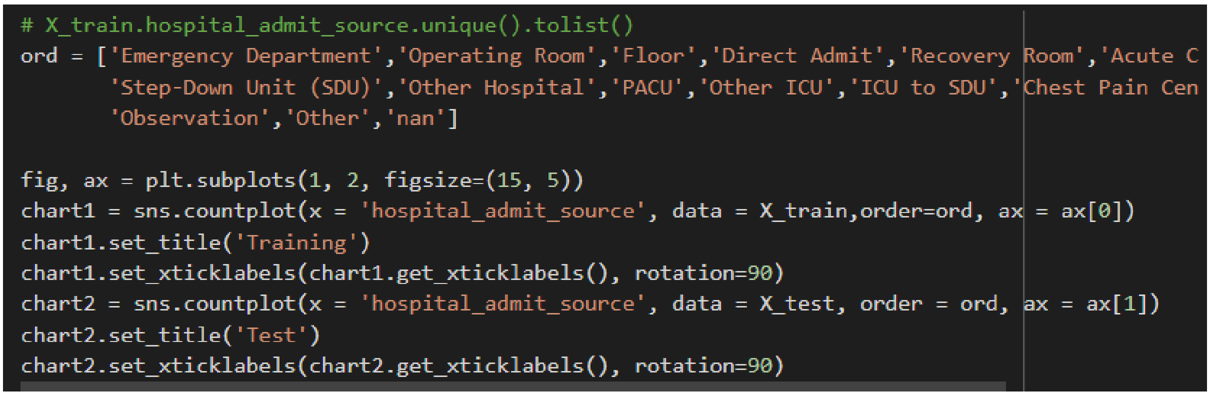


```
comorbidity =['cirrhosis','diabetes_mellitus','hepatic_failure','immunosuppression','leukemia'
for i in comorbidity:
  df_clean[i]=df_clean[i].astype('category')

demo = ['ethnicity','gender','hospital_admit_source','icu_stay_type','icu_admit_source','icu_t
for i in demo:
  df_clean[i]=df_clean[i].astype('category')


gcs = ['gcs_eyes_apache','gcs_motor_apache','gcs_verbal_apache','gcs_unable_apache']
for i in gcs:
  df_clean[i]=df_clean[i].astype('category')


for i in ['intubated_apache','arf_apache','ventilated_apache']:
  df_clean[i]=df_clean[i].astype('category')

df_clean['apache_3j_bodysystem'] =df_clean['apache_3j_bodysystem'].astype('category')
df_clean['apache_3j_diagnosis'] =df_clean['apache_3j_diagnosis'].astype('category')
df_clean['apache_post_operative'] =df_clean['apache_post_operative'].astype('category')
```

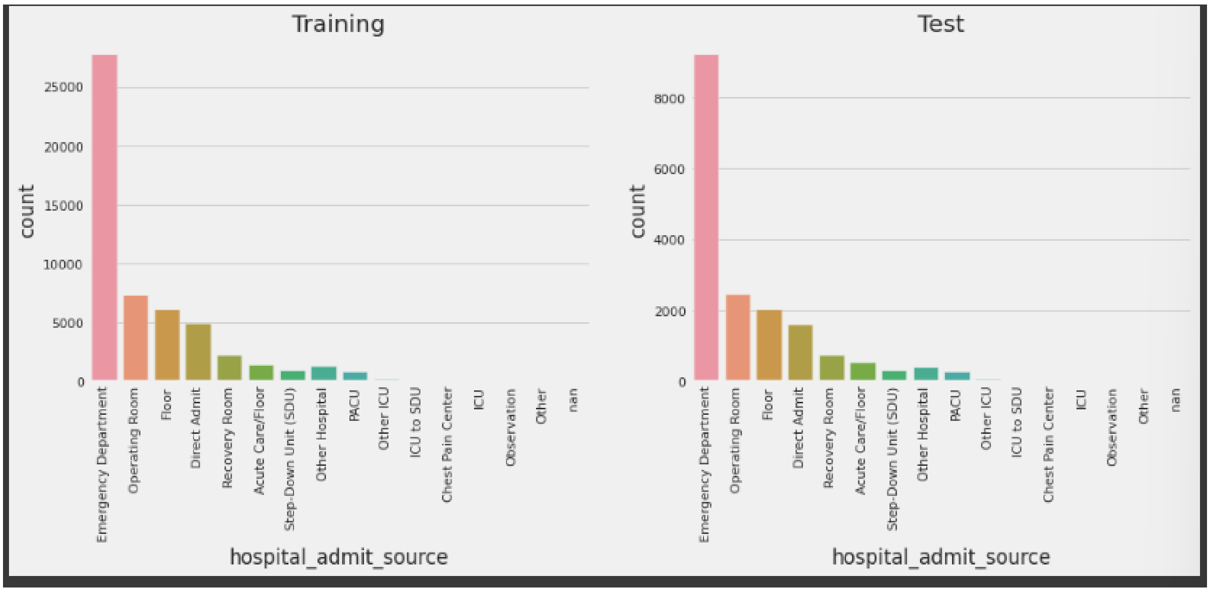


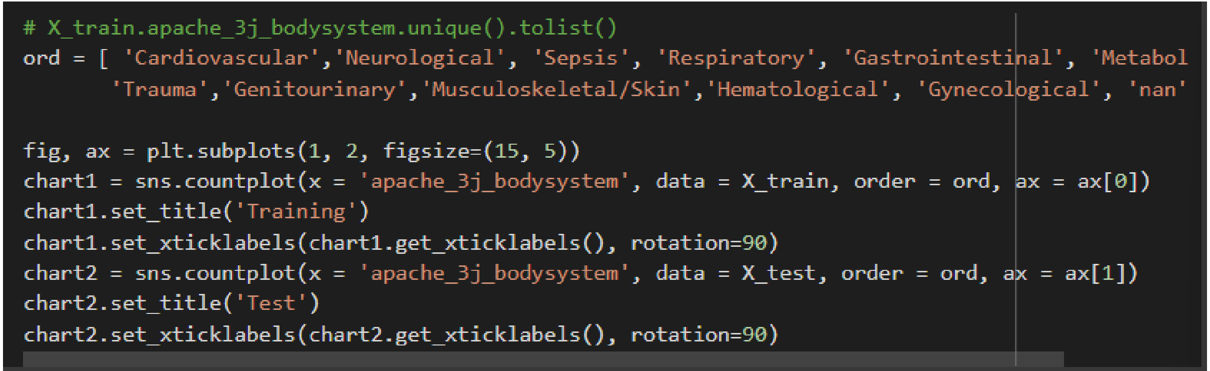
```
# Check predictor percentage
stratify = {
    'total': pd.Series(y),
    'train': pd.Series(y_train),
    'test': pd.Series(y_test)
}

pd.DataFrame(stratify).apply(pd.value_counts).apply(lambda col: col/col.sum()*100)
```

```
      total  train  test
0.00  91.37  91.37  91.37
1.00   8.63   8.63   8.63
```

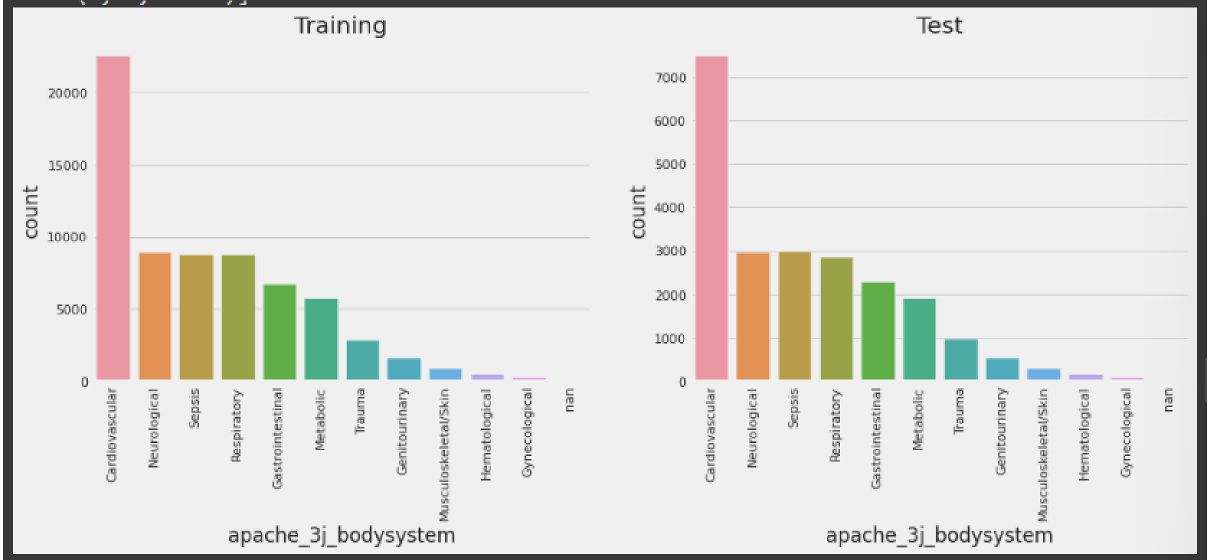


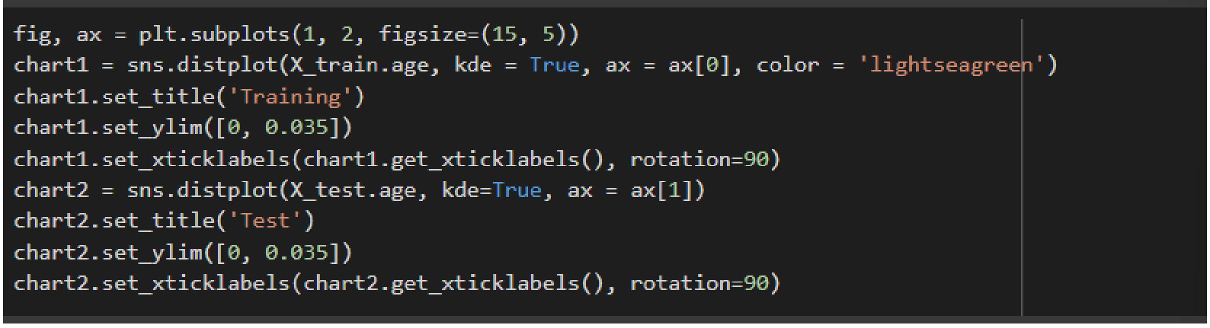
```
# X_train.hospital_admit_source.unique().tolist()
ord = ['Emergency Department','Operating Room','Floor','Direct Admit','Recovery Room','Acute C
       'Step-Down Unit (SDU)','Other Hospital','PACU','Other ICU','ICU to SDU','Chest Pain Cen
       'Observation','Other','nan']

fig, ax = plt.subplots(1, 2, figsize=(15, 5))
chart1 = sns.countplot(x = 'hospital_admit_source', data = X_train,order=ord, ax = ax[0])
chart1.set_title('Training')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90)
chart2 = sns.countplot(x = 'hospital_admit_source', data = X_test, order = ord, ax = ax[1])
chart2.set_title('Test')
chart2.set_xticklabels(chart2.get_xticklabels(), rotation=90)
```



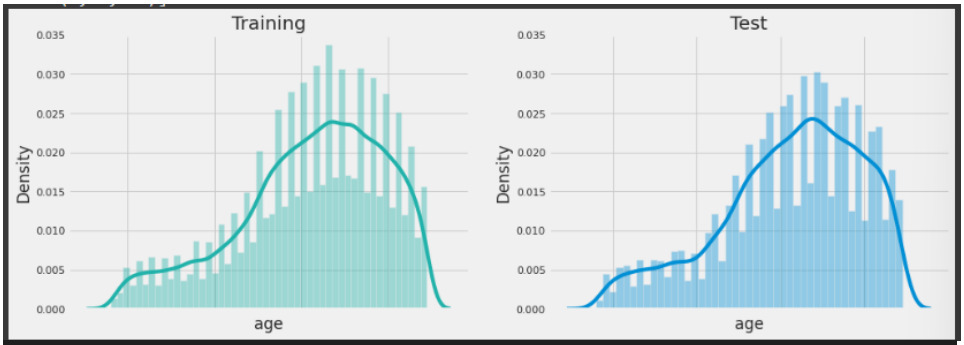


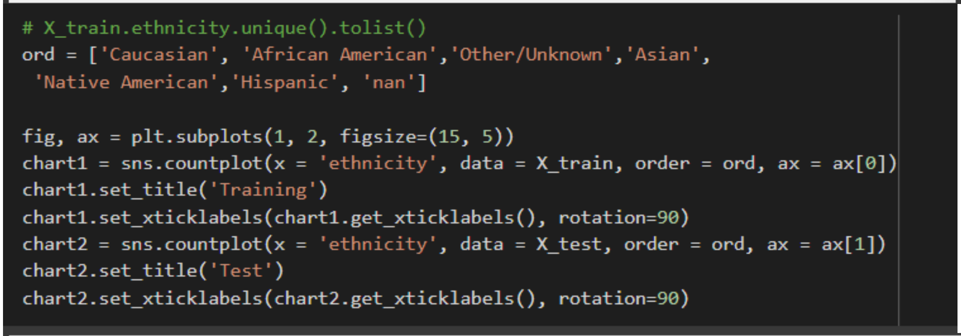
```
# X_train.apache_3j_bodysystem.unique().tolist()
ord = [ 'Cardiovascular','Neurological', 'Sepsis', 'Respiratory', 'Gastrointestinal', 'Metabol
       'Trauma','Genitourinary','Musculoskeletal/Skin','Hematological', 'Gynecological', 'nan'

fig, ax = plt.subplots(1, 2, figsize=(15, 5))
chart1 = sns.countplot(x = 'apache_3j_bodysystem', data = X_train, order = ord, ax = ax[0])
chart1.set_title('Training')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90)
chart2 = sns.countplot(x = 'apache_3j_bodysystem', data = X_test, order = ord, ax = ax[1])
chart2.set_title('Test')
chart2.set_xticklabels(chart2.get_xticklabels(), rotation=90)
```



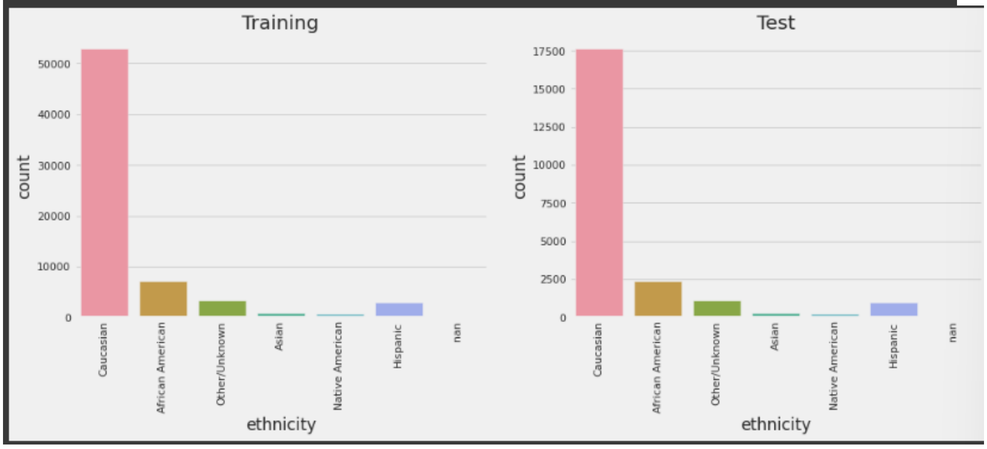


```
fig, ax = plt.subplots(1, 2, figsize=(15, 5))
chart1 = sns.distplot(X_train.age, kde = True, ax = ax[0], color = 'lightseagreen')
chart1.set_title('Training')
chart1.set_ylim([0, 0.035])
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90)
chart2 = sns.distplot(X_test.age, kde=True, ax = ax[1])
chart2.set_title('Test')
chart2.set_ylim([0, 0.035])
chart2.set_xticklabels(chart2.get_xticklabels(), rotation=90)
```

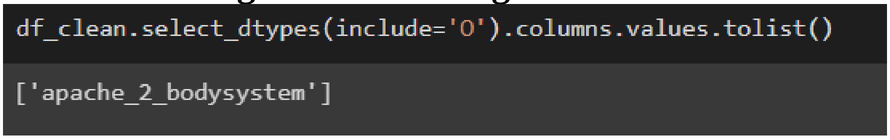




```
# X_train.ethnicity.unique().tolist()
ord = ['Caucasian', 'African American','Other/Unknown','Asian',
 'Native American','Hispanic', 'nan']

fig, ax = plt.subplots(1, 2, figsize=(15, 5))
chart1 = sns.countplot(x = 'ethnicity', data = X_train, order = ord, ax = ax[0])
chart1.set_title('Training')
chart1.set_xticklabels(chart1.get_xticklabels(), rotation=90)
chart2 = sns.countplot(x = 'ethnicity', data = X_test, order = ord, ax = ax[1])
chart2.set_title('Test')
chart2.set_xticklabels(chart2.get_xticklabels(), rotation=90)
```



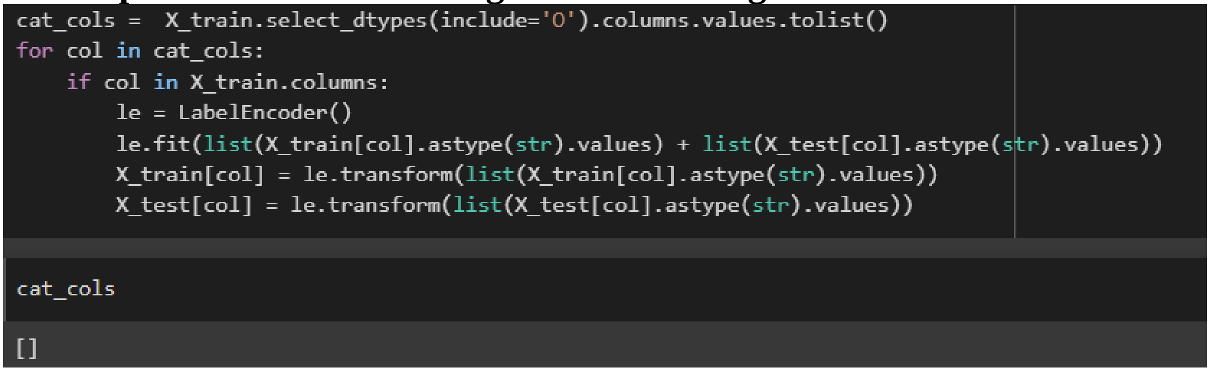
Label Encoding: Convert Categorical Variables into numerical variables.



```
df_clean.select_dtypes(include='O').columns.values.tolist()
```

```
['apache_2_bodysystem']
```

We can see “apache_2_bodysystem” is the variable with categorical data type. We should perform Label encoding & convert categorical to numerical.



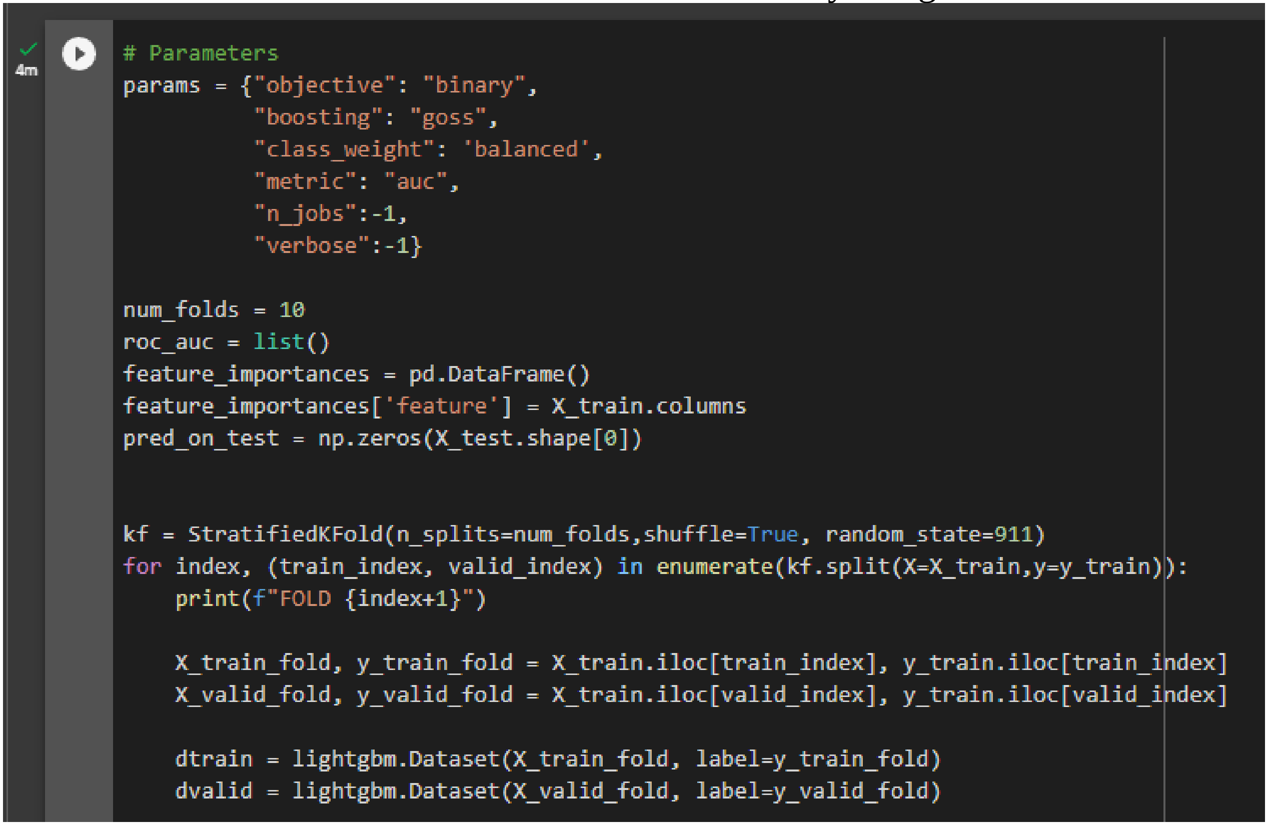
```
cat_cols =  X_train.select_dtypes(include='O').columns.values.tolist()
for col in cat_cols:
    if col in X_train.columns:
        le = LabelEncoder()
        le.fit(list(X_train[col].astype(str).values) + list(X_test[col].astype(str).values))
        X_train[col] = le.transform(list(X_train[col].astype(str).values))
        X_test[col] = le.transform(list(X_test[col].astype(str).values))
```

```
cat_cols
```

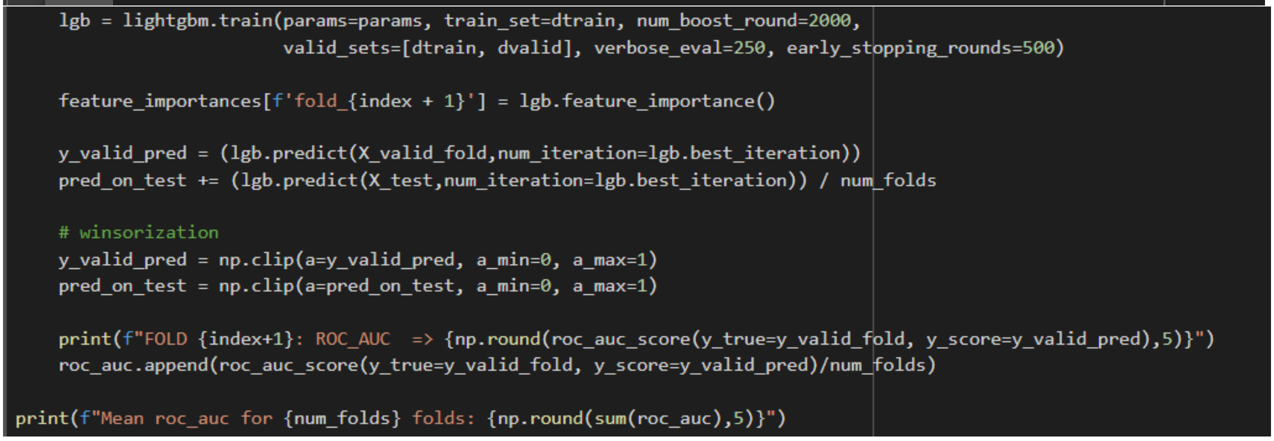
```
[]
```

Light Gradient Boosting: - Build Light Gradient Boosting Model & Evaluate the model using K-Fold Cross Validation Technique & Apply Feature importance using Light Gradient Boosting Model.

For the predictions made using Light Gradient boosting & validated the model using K-Fold cross validation, we have applied winsorization technique. Winsorization is the process of replacing the extreme values of statistical data in order to limit the effect of the outliers on the calculations or the results obtained by using that data.

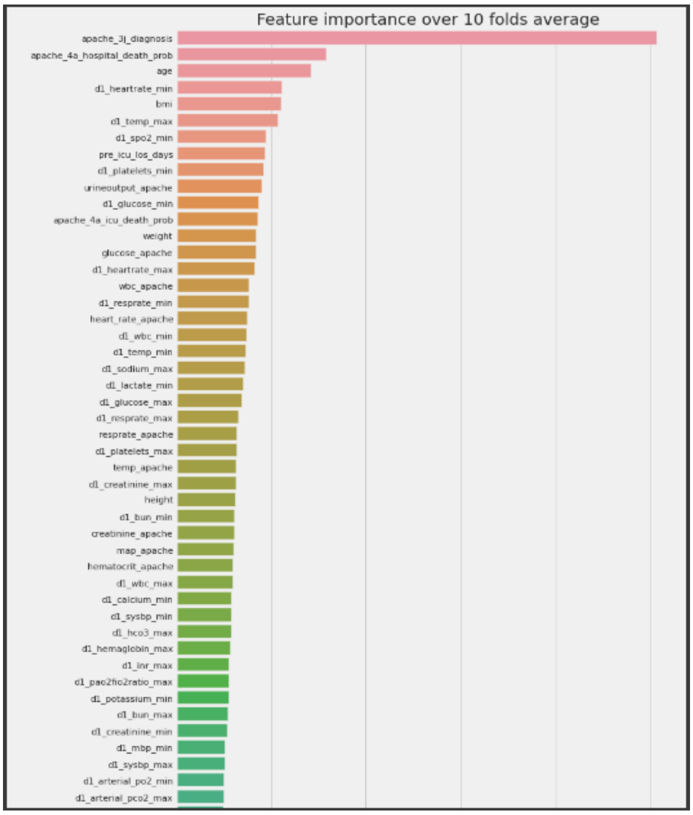


```
# Parameters
params = {"objective": "binary",
          "boosting": "goss",
          "class_weight": 'balanced',
          "metric": "auc",
          "n_jobs":-1,
          "verbose":-1}

num_folds = 10
roc_auc = list()
feature_importances = pd.DataFrame()
feature_importances['feature'] = X_train.columns
pred_on_test = np.zeros(X_test.shape[0])


kf = StratifiedKFold(n_splits=num_folds,shuffle=True, random_state=911)
for index, (train_index, valid_index) in enumerate(kf.split(X=X_train,y=y_train)):
    print(f"FOLD {index+1}")

    X_train_fold, y_train_fold = X_train.iloc[train_index], y_train.iloc[train_index]
    X_valid_fold, y_valid_fold = X_train.iloc[valid_index], y_train.iloc[valid_index]

    dtrain = lightgbm.Dataset(X_train_fold, label=y_train_fold)
    dvalid = lightgbm.Dataset(X_valid_fold, label=y_valid_fold)
```



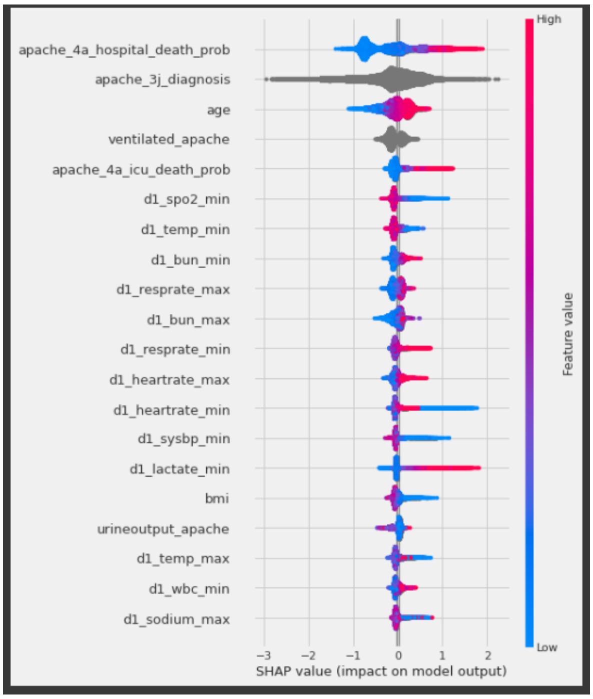
```
    lgb = lightgbm.train(params=params, train_set=dtrain, num_boost_round=2000,
                         valid_sets=[dtrain, dvalid], verbose_eval=250, early_stopping_rounds=500)

    feature_importances[f'fold_{index + 1}'] = lgb.feature_importance()

    y_valid_pred = (lgb.predict(X_valid_fold,num_iteration=lgb.best_iteration))
    pred_on_test += (lgb.predict(X_test,num_iteration=lgb.best_iteration)) / num_folds

    # winsorization
    y_valid_pred = np.clip(a=y_valid_pred, a_min=0, a_max=1)
    pred_on_test = np.clip(a=pred_on_test, a_min=0, a_max=1)

    print(f"FOLD {index+1}: ROC_AUC  => {np.round(roc_auc_score(y_true=y_valid_fold, y_score=y_valid_pred),5)}")
    roc_auc.append(roc_auc_score(y_true=y_valid_fold, y_score=y_valid_pred)/num_folds)

print(f"Mean roc_auc for {num_folds} folds: {np.round(sum(roc_auc),5)}")
```

Plotting Feature Importance using 10 fold Cross Validation



- Apache_3j_diagnosis feature is highly important.
- Features such as Elective_surgery, imputed_apache, hepatic_failure, cirrhosis, apache_post_operative, apache_3j_bodysystem, leukemia, lymphomia, aids & paco2_for_ph_apache have no feature importance.

SHAPELY EXPLANATOR:



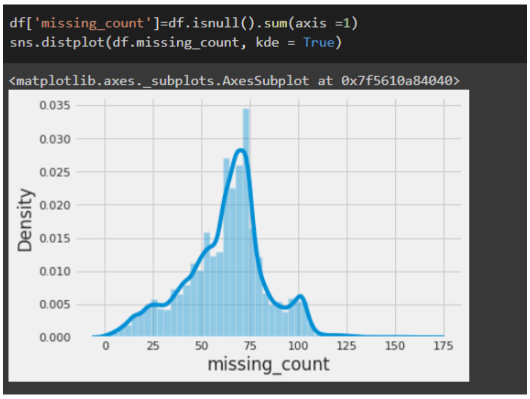
- ## Min-Max Dealing: -

We have built the baseline model using Light Gradient Boosting without proper data preprocessing, i.e., we have just cleaned the missing values & converted categorical variable to numerical variable & built LGBoost Model, Evaluated the model & plotted feature importance using K-Fold Cross Validation Technique .

*We cannot confidently tell that this model is highly efficient than existing APACHE prediction system. Hence, now we need to build **LGB (**without SMOTE, SMOTE, Trying Different Features, Adjust Threshold) **LOGISTIC REGRSSION** (Non SMOTE, SMOTE, PCA, Adjust Threshold) **CATBOOST** (Adjust Threshold), **NEURAL NETWORK** (With PCA, Adjust Threshold) for which we need to perform **DATA IMPUTATION** for better data cleaning & to obtain effective & efficient results in unbiased estimates, providing more validity than ad hoc approaches to missing data & we also need to build **Logistic Regression:** 10 versions, in which we tried different subsets: drop or add in features & **Random Forest**: 2 versions for which we need to perform **DATA BINNING** for Preprocessing the data.*

The overall Goal of this Process is to use the least features as possible as well generalize at a time to obtain better predictions than existing APACHE Prediction model.

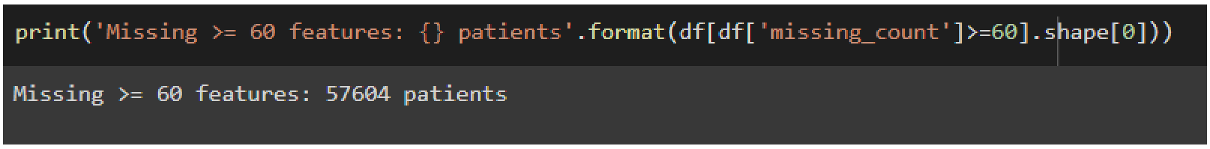
- *MIN-MAX Dealing*
  - ***Basic Imputing for Demographic info Variable***
  - ***Checking number of missing features by patient & how it is associated with hospital_death.***
  - ***Imputing other Features to deal with Max-min problem which means min value is higher than max value.***
- ***Exploring number of missing features by each patient***



```
df['missing_count']=df.isnull().sum(axis =1)
sns.distplot(df.missing_count, kde = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5610a84040>
```

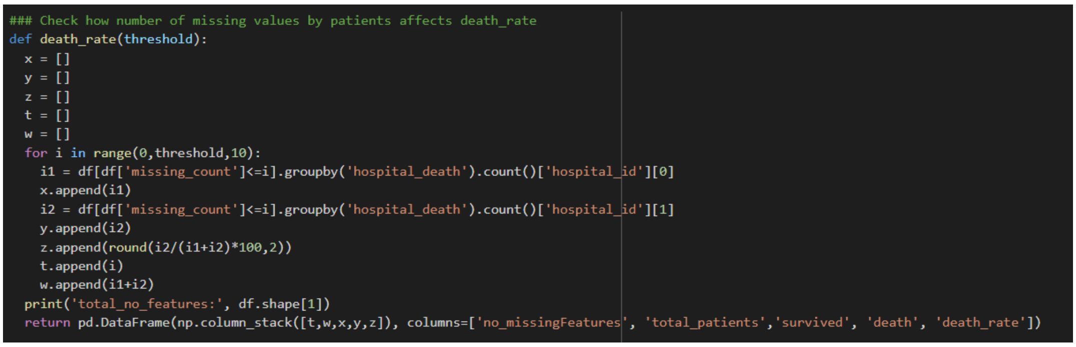
- ***Count of patients who have missing features more than 60.***



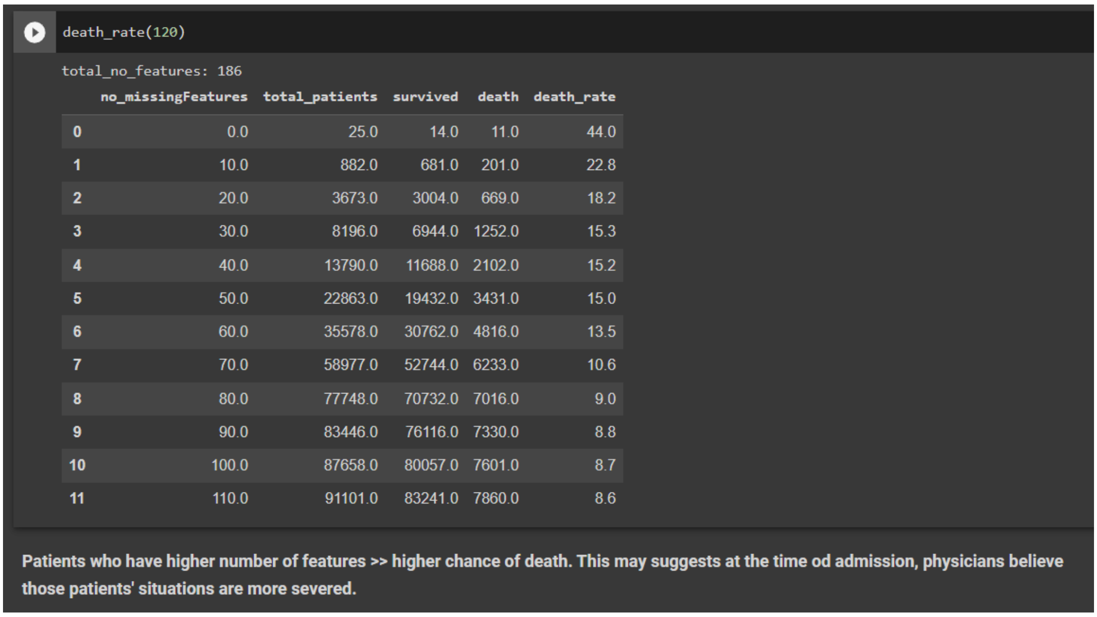
```
print('Missing >= 60 features: {} patients'.format(df[df['missing_count']>=60].shape[0]))
```

```
Missing >= 60 features: 57604 patients
```

- **Checking How the number of missing features by patients affects death rate**



```
### Check how number of missing values by patients affects death_rate
def death_rate(threshold):
  x = []
  y = []
  z = []
  t = []
  w = []
  for i in range(0,threshold,10):
    i1 = df[df['missing_count']<=i].groupby('hospital_death').count()['hospital_id'][0]
    x.append(i1)
    i2 = df[df['missing_count']<=i].groupby('hospital_death').count()['hospital_id'][1]
    y.append(i2)
    z.append(round(i2/(i1+i2)*100,2))
    t.append(i)
    w.append(i1+i2)
  print('total_no_features:', df.shape[1])
  return pd.DataFrame(np.column_stack([t,w,x,y,z]), columns=['no_missingFeatures', 'total_patients','survived', 'death', 'death_rate'])
```



```
death_rate(120)
```

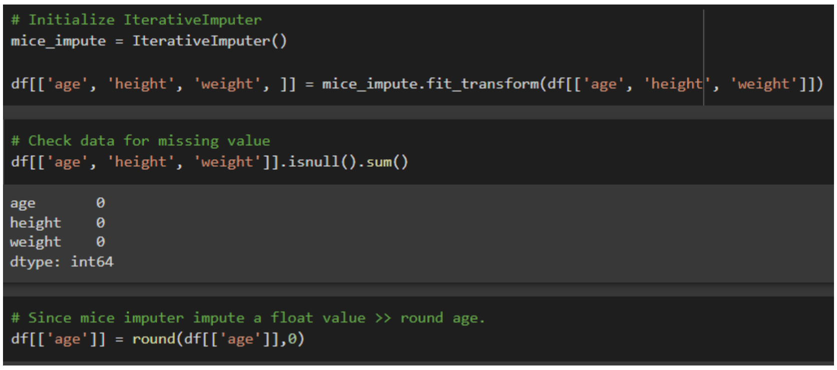
```
total_no_features: 186
```

	no_missingFeatures	total_patients	survived	death	death_rate
0	0.0	25.0	14.0	11.0	44.0
1	10.0	882.0	681.0	201.0	22.8
2	20.0	3673.0	3004.0	669.0	18.2
3	30.0	8196.0	6944.0	1252.0	15.3
4	40.0	13790.0	11688.0	2102.0	15.2
5	50.0	22863.0	19432.0	3431.0	15.0
6	60.0	35578.0	30762.0	4816.0	13.5
7	70.0	58977.0	52744.0	6233.0	10.6
8	80.0	77748.0	70732.0	7016.0	9.0
9	90.0	83446.0	76116.0	7330.0	8.8
10	100.0	87658.0	80057.0	7601.0	8.7
11	110.0	91101.0	83241.0	7860.0	8.6

Patients who have higher number of features >> higher chance of death. This may suggests at the time od admission, physicians believe those patients' situations are more severed.

- ➢ Impute **Demographic Info:** Age/Weight/Height, Ethnicity, BMI & Hospital_Admit_source, ICU_Admit_Source with respect to Death_Probability. [**MICE IMPUTER to impute categorical variables**].

  - MICE imputation for Age, Height & weight Columns



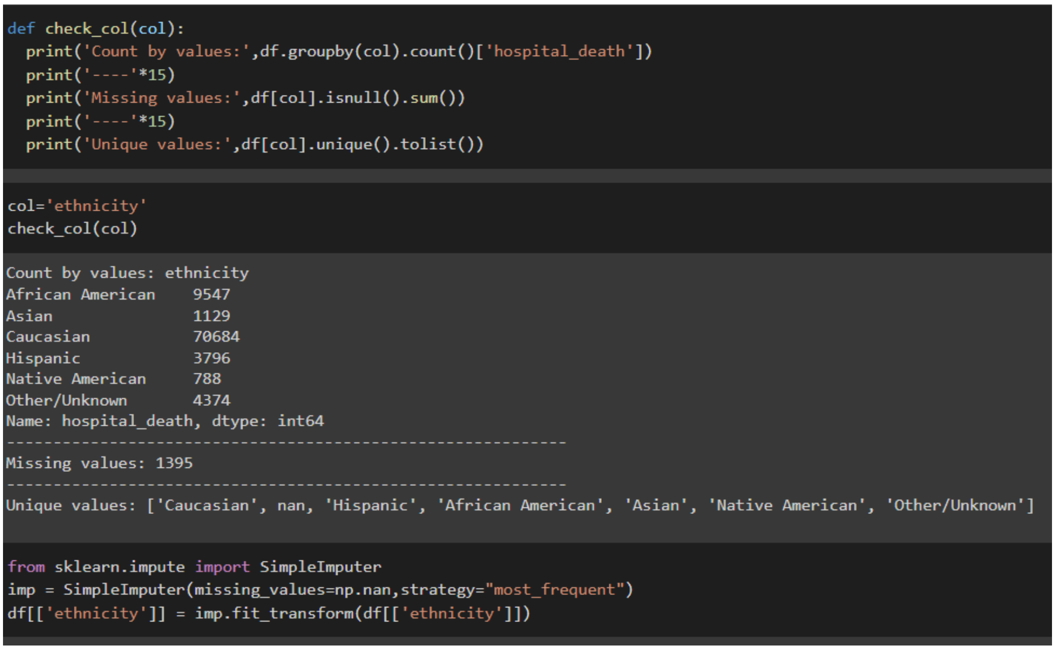
```
# Initialize IterativeImputer
mice_impute = IterativeImputer()

df[['age', 'height', 'weight', ]] = mice_impute.fit_transform(df[['age', 'height', 'weight']])
```

```
# Check data for missing value
df[['age', 'height', 'weight']].isnull().sum()
```

```
age       0
height    0
weight    0
dtype: int64
```

```
# Since mice imputer impute a float value >> round age.
df[['age']] = round(df[['age']],0)
```

- Mode imputation for Ethnicity Column

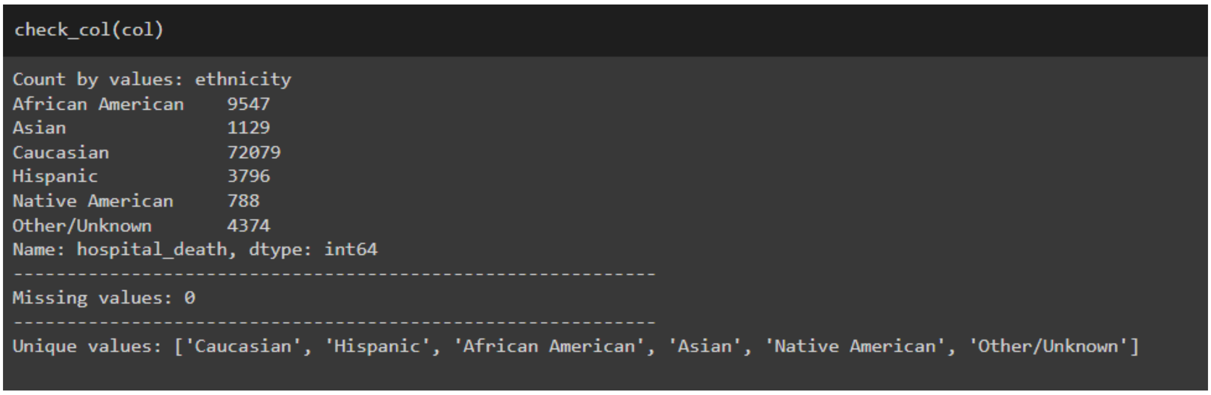


```
def check_col(col):
  print('Count by values:',df.groupby(col).count()['hospital_death'])
  print('----'*15)
  print('Missing values:',df[col].isnull().sum())
  print('----'*15)
  print('Unique values:',df[col].unique().tolist())
```

```
col='ethnicity'
check_col(col)
```

```
Count by values: ethnicity
African American    9547
Asian               1129
Caucasian           70684
Hispanic            3796
Native American     788
Other/Unknown       4374
Name: hospital_death, dtype: int64
------------------------------------------------------------
Missing values: 1395
------------------------------------------------------------
Unique values: ['Caucasian', nan, 'Hispanic', 'African American', 'Asian', 'Native American', 'Other/Unknown']
```

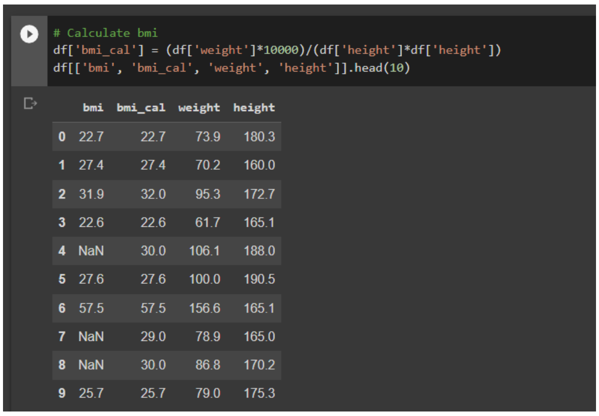
```
from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=np.nan,strategy="most_frequent")
df[['ethnicity']] = imp.fit_transform(df[['ethnicity']])
```



```
check_col(col)
```

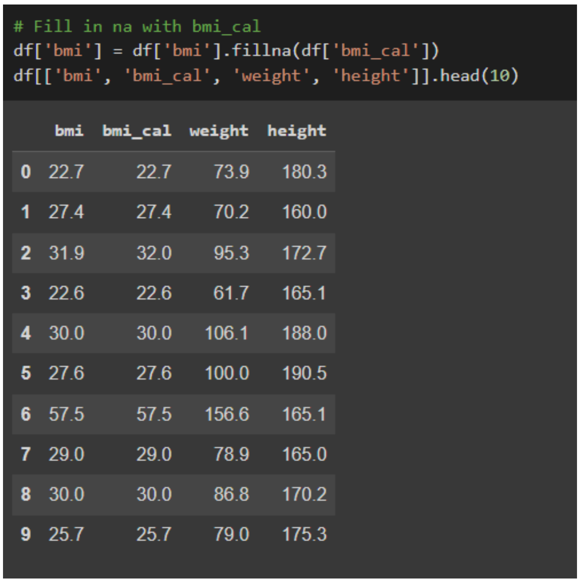
```
Count by values: ethnicity
African American    9547
Asian               1129
Caucasian           72079
Hispanic            3796
Native American     788
Other/Unknown       4374
Name: hospital_death, dtype: int64
------------------------------------------------------------
Missing values: 0
------------------------------------------------------------
Unique values: ['Caucasian', 'Hispanic', 'African American', 'Asian', 'Native American', 'Other/Unknown']
```

- Statistical Imputation for BMI column using other two columns of Height & Weight.



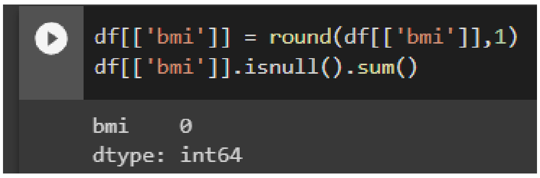
```
# Calculate bmi
df['bmi_cal'] = (df['weight']*10000)/(df['height']*df['height'])
df[['bmi', 'bmi_cal', 'weight', 'height']].head(10)
```

	bmi	bmi_cal	weight	height
0	22.7	22.7	73.9	180.3
1	27.4	27.4	70.2	160.0
2	31.9	32.0	95.3	172.7
3	22.6	22.6	61.7	165.1
4	NaN	30.0	106.1	188.0
5	27.6	27.6	100.0	190.5
6	57.5	57.5	156.6	165.1
7	NaN	29.0	78.9	165.0
8	NaN	30.0	86.8	170.2
9	25.7	25.7	79.0	175.3



```
# Fill in na with bmi_cal
df['bmi'] = df['bmi'].fillna(df['bmi_cal'])
df[['bmi', 'bmi_cal', 'weight', 'height']].head(10)
```

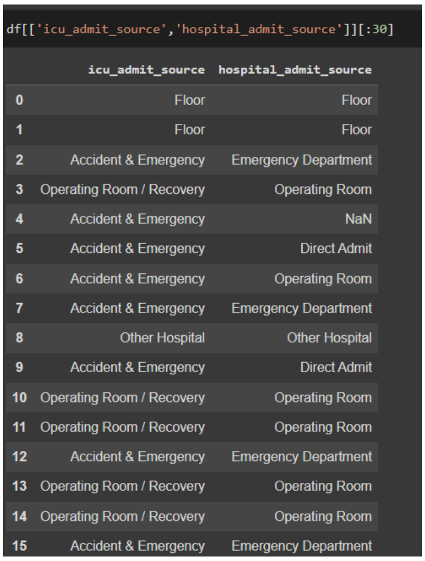
	bmi	bmi_cal	weight	height
0	22.7	22.7	73.9	180.3
1	27.4	27.4	70.2	160.0
2	31.9	32.0	95.3	172.7
3	22.6	22.6	61.7	165.1
4	30.0	30.0	106.1	188.0
5	27.6	27.6	100.0	190.5
6	57.5	57.5	156.6	165.1
7	29.0	29.0	78.9	165.0
8	30.0	30.0	86.8	170.2
9	25.7	25.7	79.0	175.3



```
df[['bmi']] = round(df[['bmi']],1)
df[['bmi']].isnull().sum()
```

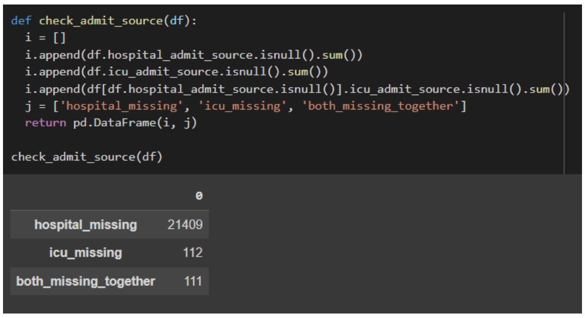
```
bmi    0
dtype: int64
```

- Mode imputation for “Hospital_Admit_Source” & “ICU_Admit_Source”



```
df[['icu_admit_source','hospital_admit_source']][:30]
```

	icu_admit_source	hospital_admit_source
0	Floor	Floor
1	Floor	Floor
2	Accident & Emergency	Emergency Department
3	Operating Room / Recovery	Operating Room
4	Accident & Emergency	NaN
5	Accident & Emergency	Direct Admit
6	Accident & Emergency	Operating Room
7	Accident & Emergency	Emergency Department
8	Other Hospital	Other Hospital
9	Accident & Emergency	Direct Admit
10	Operating Room / Recovery	Operating Room
11	Operating Room / Recovery	Operating Room
12	Accident & Emergency	Emergency Department
13	Operating Room / Recovery	Operating Room
14	Operating Room / Recovery	Operating Room
15	Accident & Emergency	Emergency Department



```
def check_admit_source(df):
  i = []
  i.append(df.hospital_admit_source.isnull().sum())
  i.append(df.icu_admit_source.isnull().sum())
  i.append(df[df.hospital_admit_source.isnull()].icu_admit_source.isnull().sum())
  j = ['hospital_missing', 'icu_missing', 'both_missing_together']
  return pd.DataFrame(i, j)

check_admit_source(df)
```

	0
hospital_missing	21409
icu_missing	112
both_missing_together	111

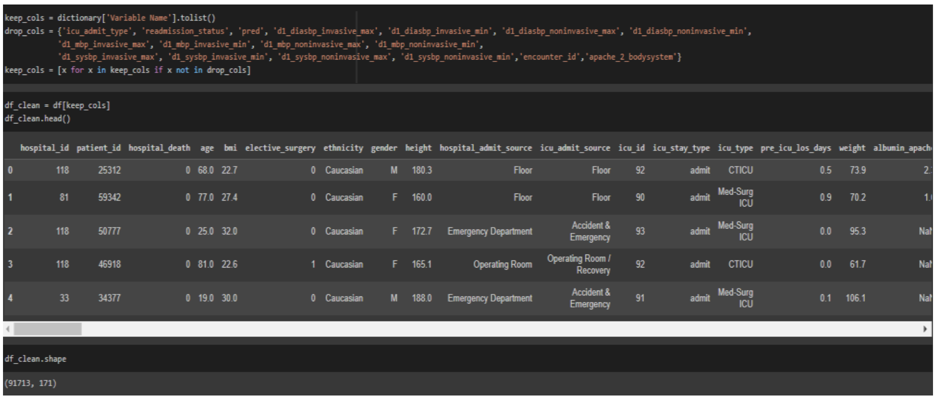
Strategy: using ICU to impute hospital

Recheck profile of those who are missing both icu and hospital admit_source

- All the apache_grouping columns are missing togerther >> fill na by 0 (N) based on ANZICS CORE dictionary also)
- Add-in col: Number of diseases/patient
- Filled na in apache_3j_body_system with 'Undefined' since they do not have post operative and there is no clear cut of how to define if they was diagnosis with any disease at time of admission.
- Save the preprocessed data to a data frame and read the data frame to a csv file.

➢ Impute **VITALS, LABS & APACHE COVARIATE Features** to deal with max-min problem.

- **VITALS-** Solved problem with max > min. For missing value, need to combine with other analysis and will create another dataset.
- **LABS** - All of feature inside labs tests have the problem with min > max >> fixed one by one. Dealing with nan later.
- **LAB Blood GAS –** Check the features having max>min & impute those features measured hourly, per day & using Apache model.
- **APACHE Covariate** check the code given for the ailment & map with ailments.



```
keep_cols = dictionary['Variable Name'].tolist()
drop_cols = {'icu_admit_type', 'readmission_status', 'pred', 'd1_diasbp_invasive_max', 'd1_diasbp_invasive_min', 'd1_diasbp_noninvasive_max', 'd1_diasbp_noninvasive_min',
          'd1_mbp_invasive_max', 'd1_mbp_invasive_min', 'd1_mbp_noninvasive_max', 'd1_mbp_noninvasive_min',
          'd1_sysbp_invasive_max', 'd1_sysbp_invasive_min', 'd1_sysbp_noninvasive_max', 'd1_sysbp_noninvasive_min','encounter_id','apache_2_bodysystem'}
keep_cols = [x for x in keep_cols if x not in drop_cols]
```

```
df_clean = df[keep_cols]
df_clean.head()
```

	hospital_id	patient_id	hospital_death	age	bmi	elective_surgery	ethnicity	gender	height	hospital_admit_source	icu_admit_source	icu_id	icu_stay_type	icu_type	pre_icu_los_days	weight	albumin_apach
0	118	25312	0	68.0	22.7	0	Caucasian	M	180.3	Floor	Floor	92	admit	CTICU	0.5	73.9	2.
1	81	59342	0	77.0	27.4	0	Caucasian	F	160.0	Floor	Floor	90	admit	Med-Surg ICU	0.9	70.2	1.
2	118	50777	0	25.0	32.0	0	Caucasian	F	172.7	Emergency Department	Accident & Emergency	93	admit	Med-Surg ICU	0.0	95.3	Na
3	118	46918	0	81.0	22.6	1	Caucasian	F	165.1	Operating Room	Operating Room / Recovery	92	admit	CTICU	0.0	61.7	Na
4	33	34377	0	19.0	30.0	0	Caucasian	M	188.0	Emergency Department	Accident & Emergency	91	admit	Med-Surg ICU	0.1	106.1	Na

```
df_clean.shape
```

```
(91713, 171)
```

Output: *After performing feature selection through baseline model & min-max dealing of all the features by imputation technique the final preprocessed data consists of 91713 attributes in 171 variables.*

DATA CLEANING, FEATURE ENGINEERING & MODEL BUILDING:

- In order to handle features with missing values we need to apply any of the missing value dealing methods like data imputation technique or data binning technique. Here in this project as we are going to develop various models for mortality rate prediction, we will be performing data imputation and data binning considering the preprocessed dataset twice separately and use the binned data for building models with Logistic Regression and Random Forest and imputed data for building model with other chosen algorithms.

  - ***Data Imputation** – Impute features with less than 50% of missing values. For Features classified as vitals & Labs, the results are different by diseases, hence we will be using apache_3j_bodysystem as an indicator to impute.*

The preprocessed data after complete **dealing with min max problem** & **data imputation** for features less than 50% & for the other features which have different results for different diseases, we will be doing following process:

a) Plot Correlation Map between the features in 3 categories: VITALS, LABS & APACHE to understand the correlation between the variables.

b) Perform Feature Engineering

c) Perform Modelling:

I. LGB – without SMOTE, with SMOTE, trying diff features, Adjust Threshold.

II. Logistic Regression – NON SMOTE, SMOTE, PCA, Adjust Threshold.

III. CATBOOST – Adjust Threshold

IV. Neural Network – With PCA, Adjust Threshold.

  - ***Data Binning** - Create Binning for Each Feature. Binning works better for Building Logistic Regression & Random Forest Models. Extreme values in binning quantile can help the model perform better.*

The preprocessed data after complete dealing with min max problem & **Data Binning** for all the features, we will be building:

I. Logistic Regression – we will be performing different versions by trying different feature subsets by adding/adding features. [Which helps us to consider only few features for predicting hospital mortality with high accuracy than existing APACHE Prediction model.

II. Random Forest – we will be performing different versions even in this model just like the above Logistic regression model.

Finally deciding the best efficient model which gives higher accuracy & considering only important features than the existing APACHE Model.

MODELLING:

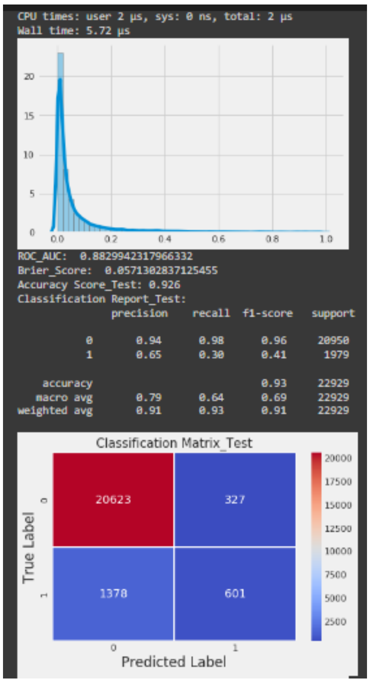
- **Results of Modelling in Imputing Approach and Binning Approach.**

I. LOGOSTIC REGRESSION: -

A) Logistic Regression – Binning:

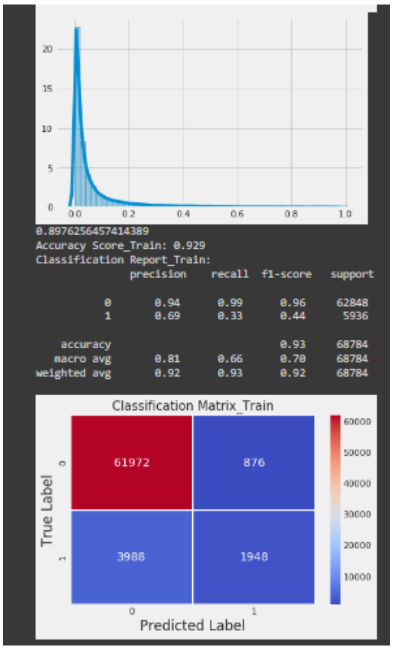
Without Smote:

Train data **Test Data**



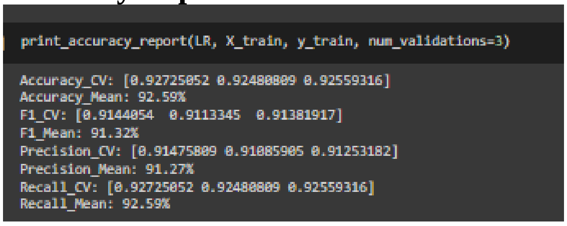
```
CPU times: user 2 µs, sys: 0 ns, total: 2 µs
Wall time: 5.72 µs
```

```
ROC_AUC:  0.8829942317966332
Brier_Score:  0.0571302837125455
Accuracy Score_Test: 0.926
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.94      0.98      0.96     20950
           1       0.65      0.30      0.41      1979

    accuracy                           0.93     22929
   macro avg       0.79      0.64      0.69     22929
weighted avg       0.91      0.93      0.91     22929
```



```
0.8976256457414389
Accuracy Score_Train: 0.929
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.94      0.99      0.96     62848
           1       0.69      0.33      0.44      5936

    accuracy                           0.93     68784
   macro avg       0.81      0.66      0.70     68784
weighted avg       0.92      0.93      0.92     68784
```

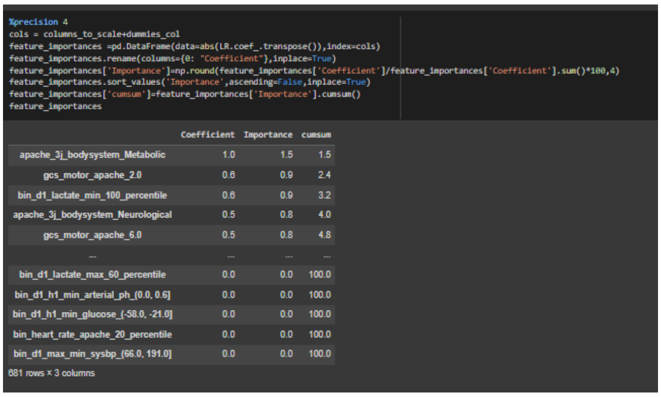
Accuracy Report:



```
print_accuracy_report(LR, X_train, y_train, num_validations=3)
```

```
Accuracy_CV: [0.92725052 0.92480809 0.92559316]
Accuracy_Mean: 92.59%
F1_CV: [0.9144054  0.9113345  0.91381917]
F1_Mean: 91.32%
Precision_CV: [0.91475809 0.91085905 0.91253182]
Precision_Mean: 91.27%
Recall_CV: [0.92725052 0.92480809 0.92559316]
Recall_Mean: 92.59%
```

Feature Importance:

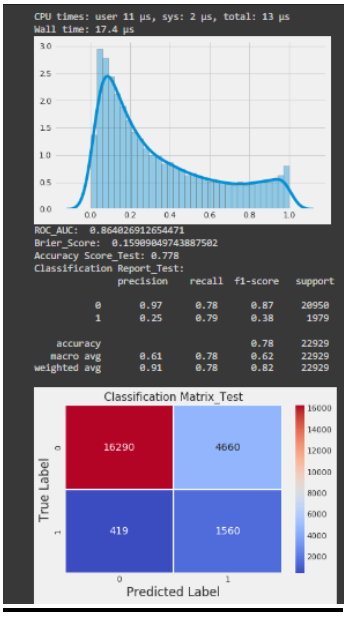


```
%precision 4
cols = columns_to_scale+dummies_col
feature_importances =pd.DataFrame(data=abs(LR.coef_.transpose()),index=cols)
feature_importances.rename(columns={0: "Coefficient"},inplace=True)
feature_importances['Importance']=np.round(feature_importances['Coefficient']/feature_importances['Coefficient'].sum()*100,4)
feature_importances.sort_values('Importance',ascending=False,inplace=True)
feature_importances['cumsum']=feature_importances['Importance'].cumsum()
feature_importances
```

	Coefficient	Importance	cumsum
apache_3j_bodysystem_Metabolic	1.0	1.5	1.5
gcs_motor_apache_2.0	0.6	0.9	2.4
bin_d1_lactate_min_100_percentile	0.6	0.9	3.2
apache_3j_bodysystem_Neurological	0.5	0.8	4.0
gcs_motor_apache_6.0	0.5	0.8	4.8
...	...	...	...
bin_d1_lactate_max_60_percentile	0.0	0.0	100.0
bin_d1_h1_min_arterial_ph_(0.0, 0.6]	0.0	0.0	100.0
bin_d1_h1_min_glucose_(-58.0, -21.0]	0.0	0.0	100.0
bin_heart_rate_apache_20_percentile	0.0	0.0	100.0
bin_d1_max_min_sysbp_(66.0, 191.0]	0.0	0.0	100.0

681 rows × 3 columns

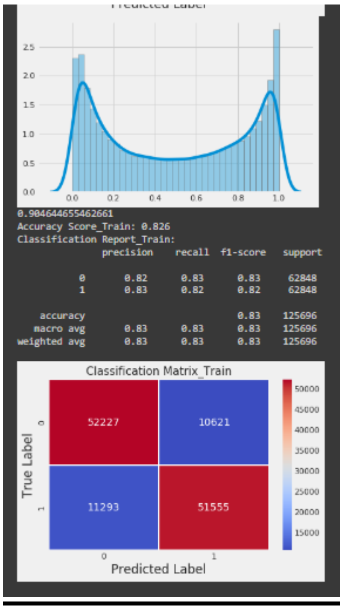
With Smote:



```
CPU times: user 11 µs, sys: 2 µs, total: 13 µs
Wall time: 17.4 µs
```

```
ROC_AUC:  0.864026912654471
Brier_Score:  0.15909049743887502
Accuracy Score_Test: 0.778
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.97      0.78      0.87     20950
           1       0.25      0.79      0.38      1979

    accuracy                           0.78     22929
   macro avg       0.61      0.78      0.62     22929
weighted avg       0.91      0.78      0.82     22929
```

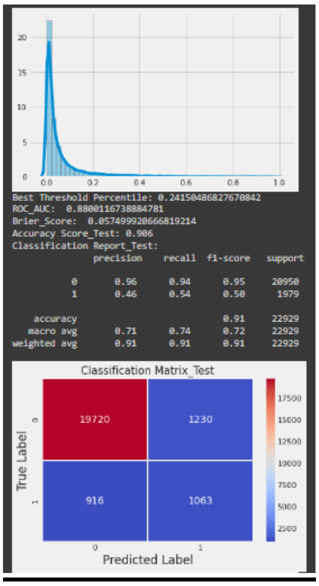
Classification Matrix_Test



```
0.904644655462661
Accuracy Score_Train: 0.826
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.82      0.83      0.83     62848
           1       0.83      0.82      0.82     62848

    accuracy                           0.83    125696
   macro avg       0.83      0.83      0.83    125696
weighted avg       0.83      0.83      0.83    125696
```

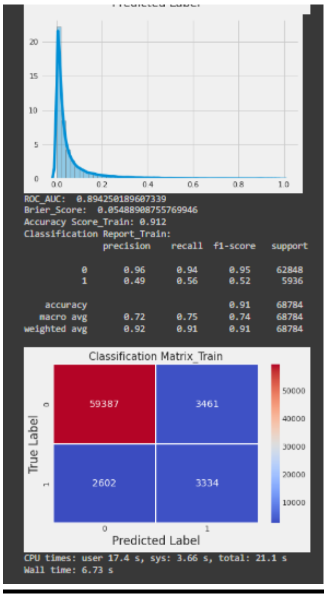
Classification Matrix_Train

Adjusted Threshold – 90%



```
Best Threshold Percentile: 0.24150486827670842
ROC_AUC:  0.8800116738884781
Brier_Score:  0.057499920666819214
Accuracy Score_Test: 0.906
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95     20950
           1       0.46      0.54      0.50      1979

    accuracy                           0.91     22929
   macro avg       0.71      0.74      0.72     22929
weighted avg       0.91      0.91      0.91     22929
```

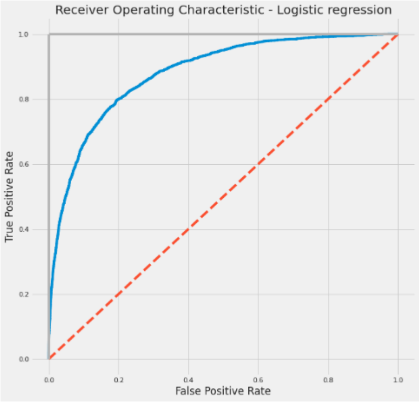
Classification Matrix_Test



```
ROC_AUC:  0.894250189607339
Brier_Score:  0.05488908755769946
Accuracy Score_Train: 0.912
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95     62848
           1       0.49      0.56      0.52      5936

    accuracy                           0.91     68784
   macro avg       0.72      0.75      0.74     68784
weighted avg       0.92      0.91      0.91     68784
```

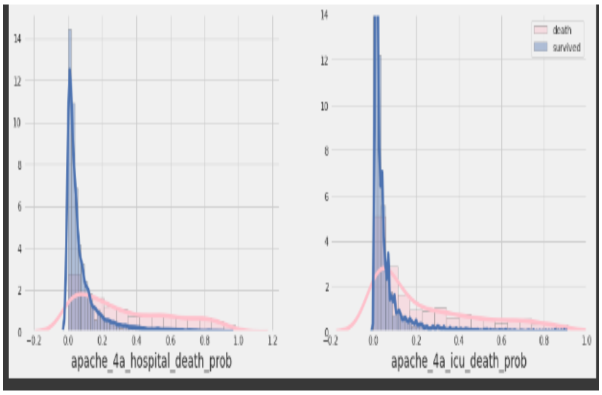
Classification Matrix_Train

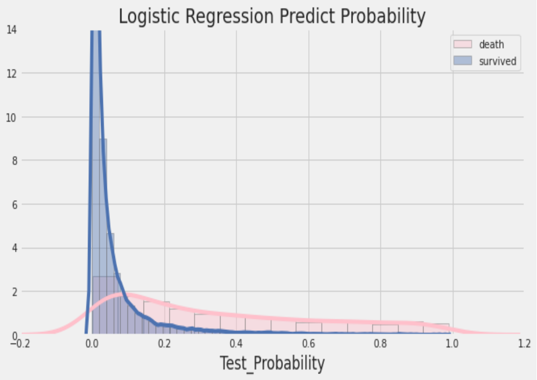
```
CPU times: user 17.4 s, sys: 3.66 s, total: 21.1 s
Wall time: 6.73 s
```



Receiver Operating Characteristic - Logistic regression

Comparison with APACHE -IV System



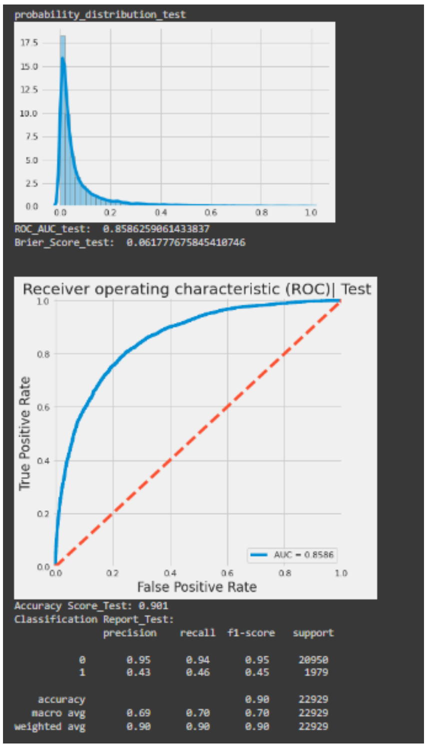


B. Logistic Regression - Imputing

i) Without smote:

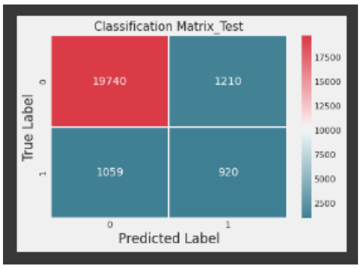
Classification report _Test

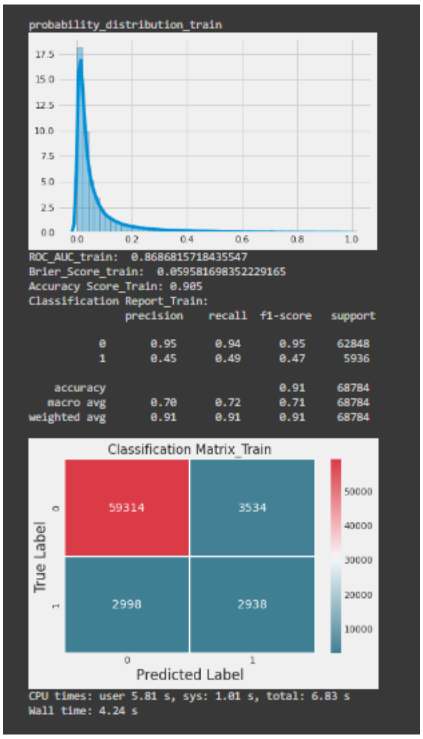
Classification Report _Train


probability_distribution_test
Receiver operating characteristic (ROC)| Test
True Positive Rate
False Positive Rate
AUC = 0.8586

```
ROC_AUC_test:  0.8586259061433837
Brier_Score_test:  0.061777675845410746
```

```
Accuracy Score_Test: 0.901
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.95      0.94      0.95     20950
           1       0.43      0.46      0.45      1979

    accuracy                           0.90     22929
   macro avg       0.69      0.70      0.70     22929
weighted avg       0.90      0.90      0.90     22929
```

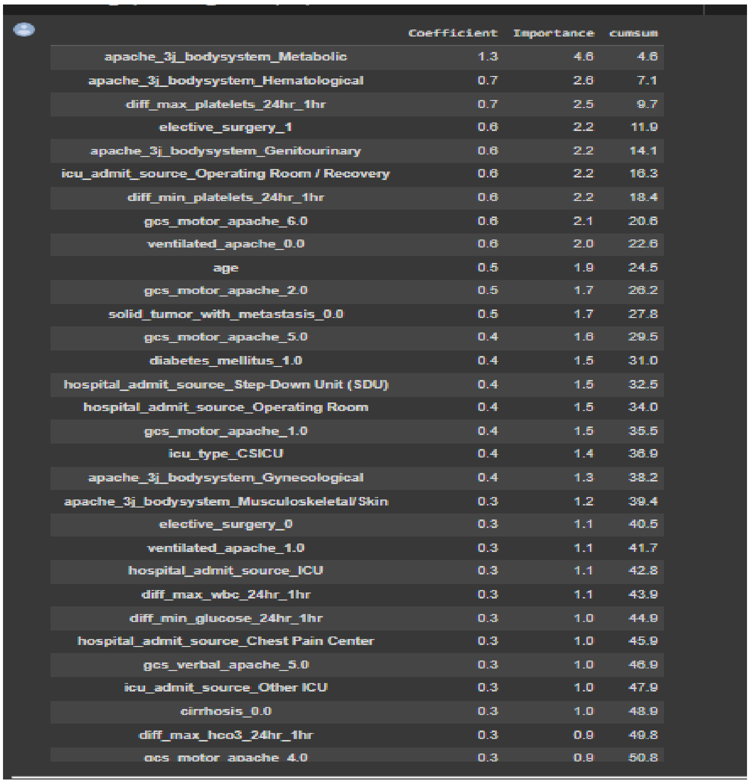




```
ROC_AUC_train:  0.8686815718435547
Brier_Score_train:  0.059581698352229165
Accuracy Score_Train: 0.905
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.95      0.94      0.95     62848
           1       0.45      0.49      0.47      5936

    accuracy                           0.91     68784
   macro avg       0.70      0.72      0.71     68784
weighted avg       0.91      0.91      0.91     68784
```

```
CPU times: user 5.81 s, sys: 1.01 s, total: 6.83 s
Wall time: 4.24 s
```

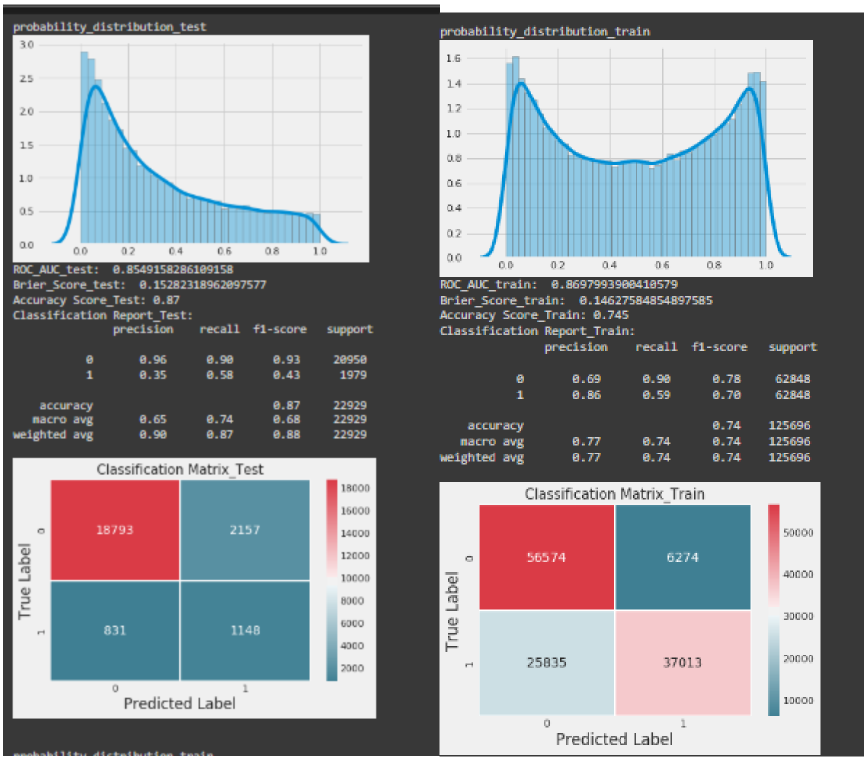
FEATURE IMPORTANCE:



	Coefficient	Importance	cumsum
apache_3j_bodysystem_Metabolic	1.3	4.6	4.6
apache_3j_bodysystem_Hematological	0.7	2.6	7.1
diff_max_platelets_24hr_1hr	0.7	2.5	9.7
elective_surgery_1	0.6	2.2	11.9
apache_3j_bodysystem_Genitourinary	0.6	2.2	14.1
icu_admit_source_Operating Room / Recovery	0.6	2.2	16.3
diff_min_platelets_24hr_1hr	0.6	2.2	18.4
gcs_motor_apache_6.0	0.6	2.1	20.6
ventilated_apache_0.0	0.6	2.0	22.6
age	0.5	1.9	24.5
gcs_motor_apache_2.0	0.5	1.7	26.2
solid_tumor_with_metastasis_0.0	0.5	1.7	27.8
gcs_motor_apache_5.0	0.4	1.6	29.5
diabetes_mellitus_1.0	0.4	1.5	31.0
hospital_admit_source_Step-Down Unit (SDU)	0.4	1.5	32.5
hospital_admit_source_Operating Room	0.4	1.5	34.0
gcs_motor_apache_1.0	0.4	1.5	35.5
icu_type_CSICU	0.4	1.4	36.9
apache_3j_bodysystem_Gynecological	0.4	1.3	38.2
apache_3j_bodysystem_Musculoskeletal/Skin	0.3	1.2	39.4
elective_surgery_0	0.3	1.1	40.5
ventilated_apache_1.0	0.3	1.1	41.7
hospital_admit_source_ICU	0.3	1.1	42.8
diff_max_wbc_24hr_1hr	0.3	1.1	43.9
diff_min_glucose_24hr_1hr	0.3	1.0	44.9
hospital_admit_source_Chest Pain Center	0.3	1.0	45.9
gcs_verbal_apache_5.0	0.3	1.0	46.9
icu_admit_source_Other ICU	0.3	1.0	47.9
cirrhosis_0.0	0.3	1.0	48.9
diff_max_hco3_24hr_1hr	0.3	0.9	49.8
gcs_motor_apache_4.0	0.3	0.9	50.8

ii) With Smote:

Classification Report_Test



probability_distribution_test

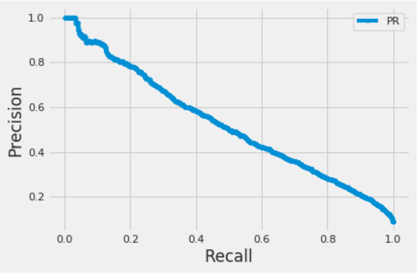
```
ROC_AUC_test:  0.8549158286109158
Brier_Score_test:  0.15282318962097577
Accuracy Score_Test: 0.87
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.96      0.90      0.93     20950
           1       0.35      0.58      0.43      1979

    accuracy                           0.87     22929
   macro avg       0.65      0.74      0.68     22929
weighted avg       0.90      0.87      0.88     22929
```

Classification Matrix_Test

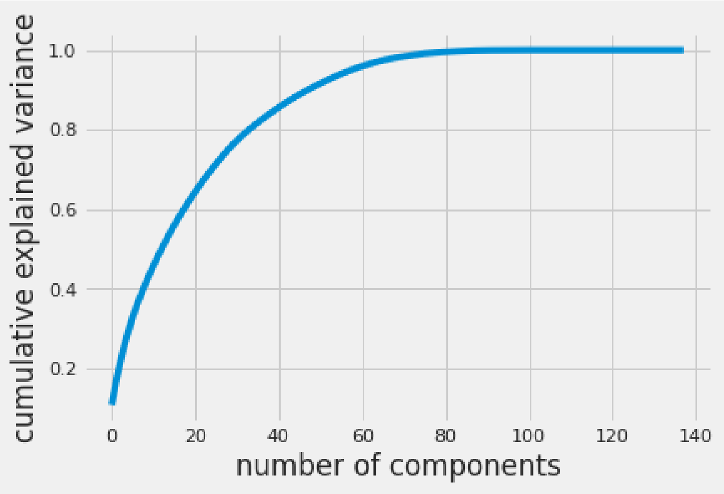
probability_distribution_train

```
ROC_AUC_train:  0.8697993900410579
Brier_Score_train:  0.14627584854897585
Accuracy Score_Train: 0.745
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.69      0.90      0.78     62848
           1       0.86      0.59      0.70     62848

    accuracy                           0.74    125696
   macro avg       0.77      0.74      0.74    125696
weighted avg       0.77      0.74      0.74    125696
```

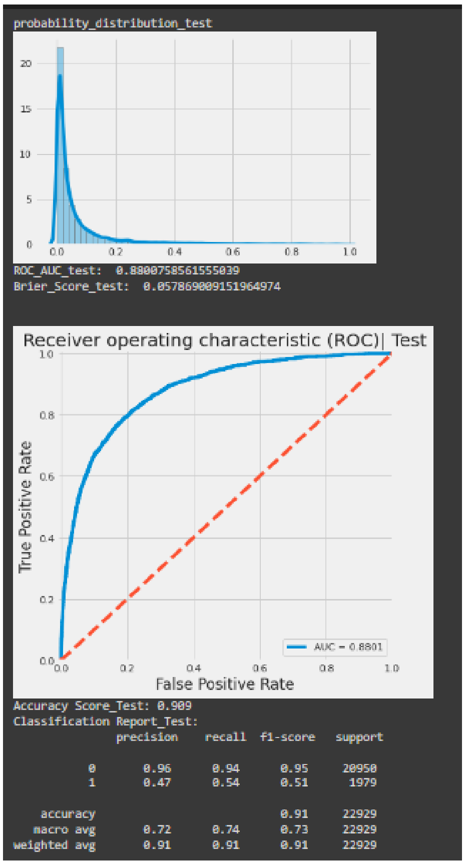
Classification Matrix_Train



iii) Logistic Regression – Principal Component Analysis

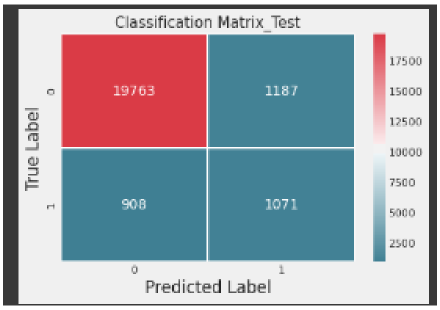


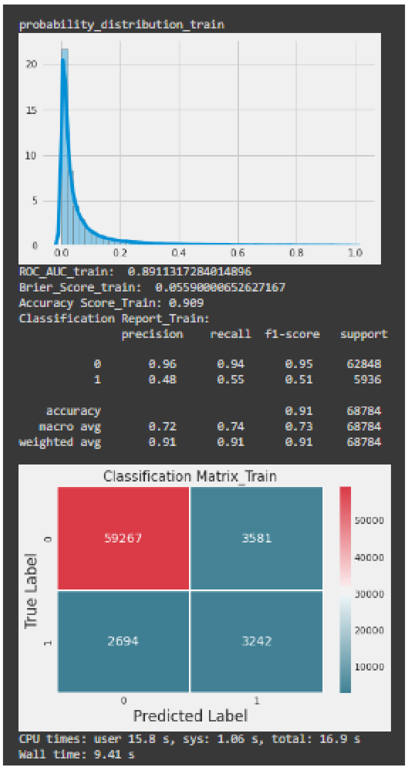
Using 70 out of 138 Principal Components


probability_distribution_test
Receiver operating characteristic (ROC)| Test
True Positive Rate
False Positive Rate
AUC = 0.8801

```
ROC_AUC_test:  0.8800758561555039
Brier_Score_test:  0.057869009151964974
```

```
Accuracy Score_Test: 0.909
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95     20950
           1       0.47      0.54      0.51      1979

    accuracy                           0.91     22929
   macro avg       0.72      0.74      0.73     22929
weighted avg       0.91      0.91      0.91     22929
```

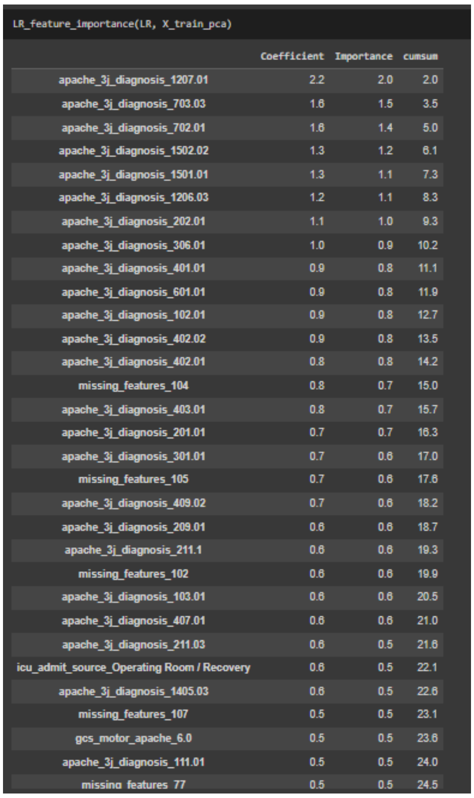




```
ROC_AUC_train:  0.8911317284014896
Brier_Score_train:  0.05590000652627167
Accuracy Score_Train: 0.909
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95     62848
           1       0.48      0.55      0.51      5936

    accuracy                           0.91     68784
   macro avg       0.72      0.74      0.73     68784
weighted avg       0.91      0.91      0.91     68784
```

```
CPU times: user 15.8 s, sys: 1.06 s, total: 16.9 s
Wall time: 9.41 s
```

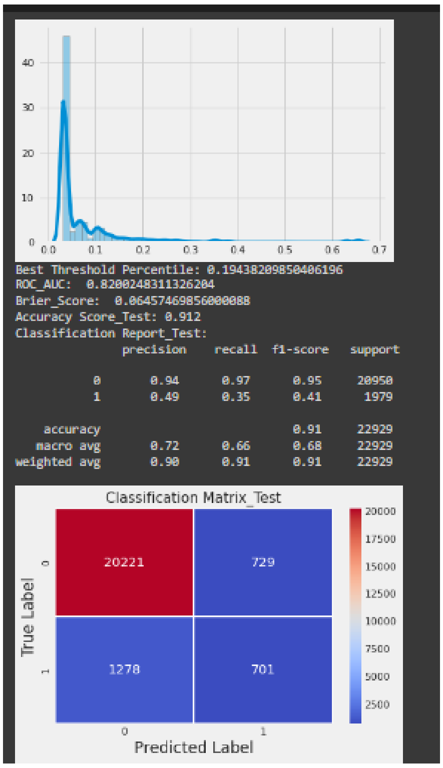
FEATURE IMPORTANCE:



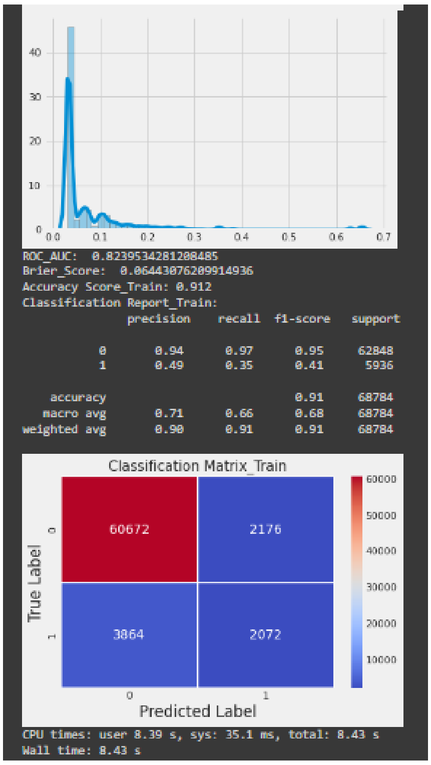
```
LR_feature_importance(LR, X_train_pca)
```

	Coefficient	Importance	cumsum
apache_3j_diagnosis_1207.01	2.2	2.0	2.0
apache_3j_diagnosis_703.03	1.6	1.5	3.5
apache_3j_diagnosis_702.01	1.6	1.4	5.0
apache_3j_diagnosis_1502.02	1.3	1.2	6.1
apache_3j_diagnosis_1501.01	1.3	1.1	7.3
apache_3j_diagnosis_1206.03	1.2	1.1	8.3
apache_3j_diagnosis_202.01	1.1	1.0	9.3
apache_3j_diagnosis_306.01	1.0	0.9	10.2
apache_3j_diagnosis_401.01	0.9	0.8	11.1
apache_3j_diagnosis_601.01	0.9	0.8	11.9
apache_3j_diagnosis_102.01	0.9	0.8	12.7
apache_3j_diagnosis_402.02	0.9	0.8	13.5
apache_3j_diagnosis_402.01	0.8	0.8	14.2
missing_features_104	0.8	0.7	15.0
apache_3j_diagnosis_403.01	0.8	0.7	15.7
apache_3j_diagnosis_201.01	0.7	0.7	16.3
apache_3j_diagnosis_301.01	0.7	0.6	17.0
missing_features_105	0.7	0.6	17.6
apache_3j_diagnosis_409.02	0.7	0.6	18.2
apache_3j_diagnosis_209.01	0.6	0.6	18.7
apache_3j_diagnosis_211.1	0.6	0.6	19.3
missing_features_102	0.6	0.6	19.9
apache_3j_diagnosis_103.01	0.6	0.6	20.5
apache_3j_diagnosis_407.01	0.6	0.6	21.0
apache_3j_diagnosis_211.03	0.6	0.5	21.6
icu_admit_source_Operating Room / Recovery	0.6	0.5	22.1
apache_3j_diagnosis_1405.03	0.6	0.5	22.6
missing_features_107	0.5	0.5	23.1
gcs_motor_apache_6.0	0.5	0.5	23.6
apache_3j_diagnosis_111.01	0.5	0.5	24.0
missing_features_77	0.5	0.5	24.5

II. Random Forest – Binning



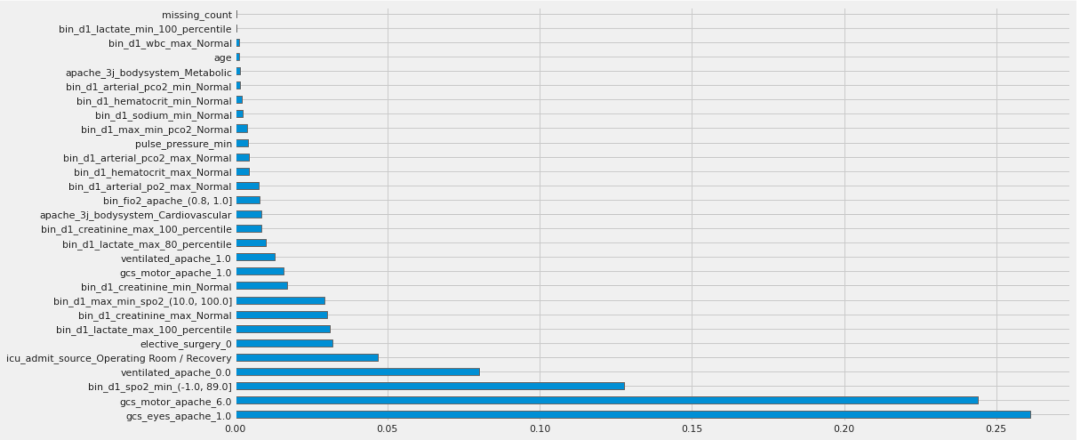
```
Best Threshold Percentile: 0.19438209850406196
ROC_AUC:  0.8200248311326204
Brier_Score:  0.0645746985600008
Accuracy Score_Test: 0.912
Classification Report_Test:
              precision    recall  f1-score   support

           0       0.94      0.97      0.95     20950
           1       0.49      0.35      0.41      1979

    accuracy                           0.91     22929
   macro avg       0.72      0.66      0.68     22929
weighted avg       0.90      0.91      0.91     22929
```



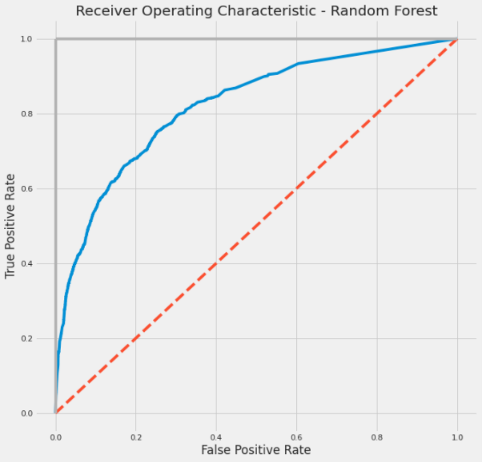
```
ROC_AUC:  0.8239534281208485
Brier_Score:  0.06443076209914936
Accuracy Score_Train: 0.912
Classification Report_Train:
              precision    recall  f1-score   support

           0       0.94      0.97      0.95     62848
           1       0.49      0.35      0.41      5936

    accuracy                           0.91     68784
   macro avg       0.71      0.66      0.68     68784
weighted avg       0.90      0.91      0.91     68784
```

```
CPU times: user 8.39 s, sys: 35.1 ms, total: 8.43 s
Wall time: 8.43 s
```

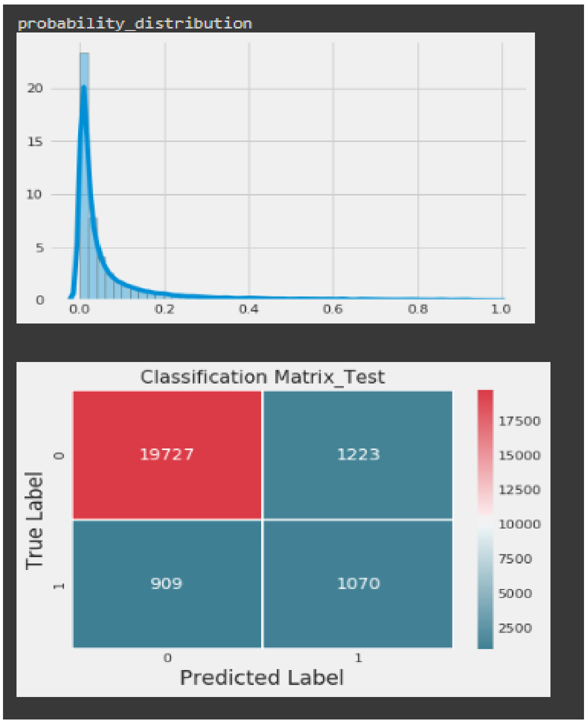
Feature Importance:

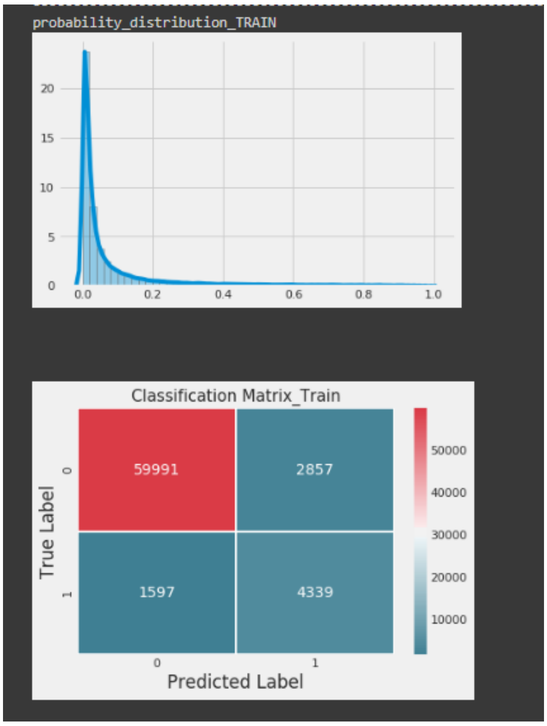


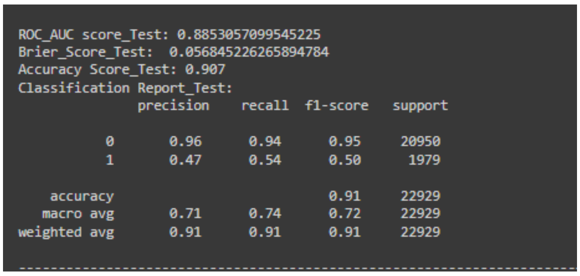
ROC-AUC Curve:



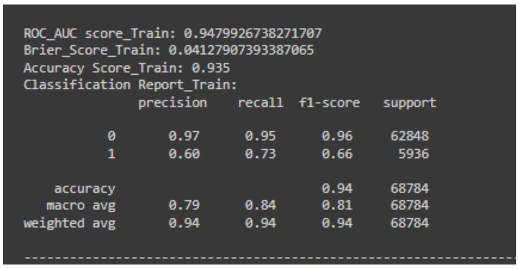
III. Light Gradient Boosting – Imputing:



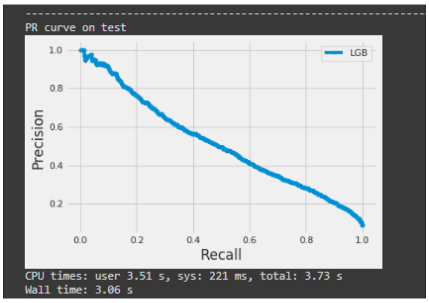




```
ROC_AUC score_Test: 0.8853057099545225
Brier_Score_Test:  0.056845226265894784
Accuracy Score_Test: 0.907
Classification Report_Test:
               precision    recall  f1-score   support

           0       0.96      0.94      0.95     20950
           1       0.47      0.54      0.50      1979

    accuracy                           0.91     22929
   macro avg       0.71      0.74      0.72     22929
weighted avg       0.91      0.91      0.91     22929

------------------------------------------------------------------------
```

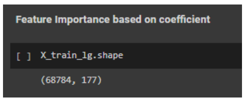


```
ROC_AUC score_Train: 0.9479926738271707
Brier_Score_Train: 0.04127907393387065
Accuracy Score_Train: 0.935
Classification Report_Train:
               precision    recall  f1-score   support

           0       0.97      0.95      0.96     62848
           1       0.60      0.73      0.66      5936

    accuracy                           0.94     68784
   macro avg       0.79      0.84      0.81     68784
weighted avg       0.94      0.94      0.94     68784

------------------------------------------------------------------------
```



```
CPU times: user 3.51 s, sys: 221 ms, total: 3.73 s
Wall time: 3.06 s
```

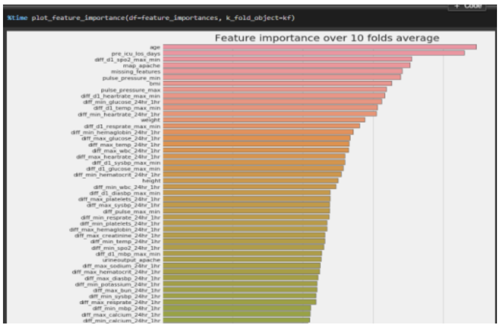
FEATURE IMPORTANCE:



Feature Importance based on coefficient

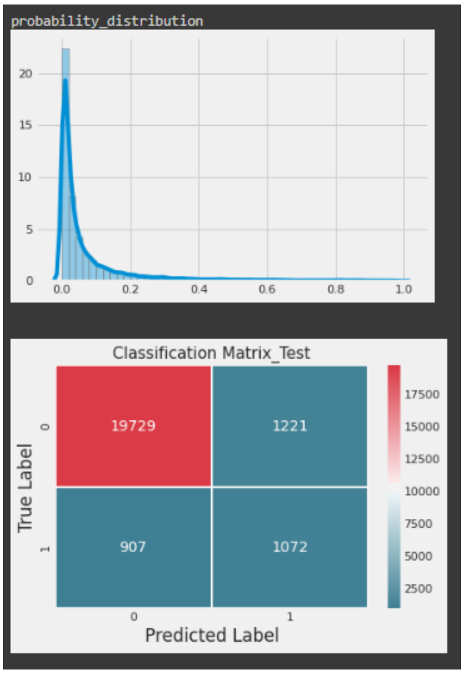
```
[ ] X_train_lg.shape

    (68784, 177)
```

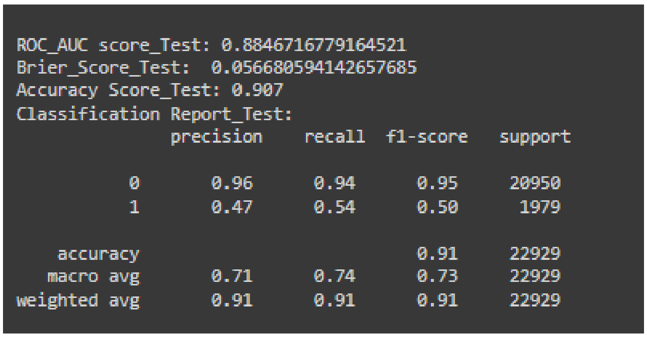
177 Features



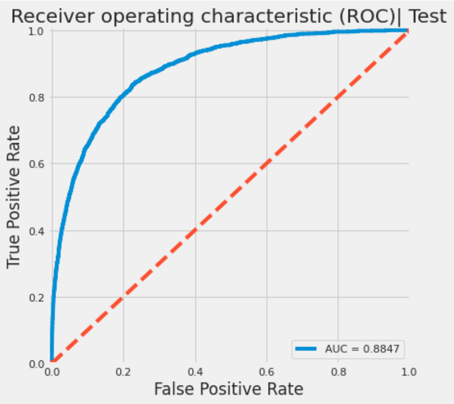
```
%time plot_feature_importance(df=feature_importances, k_fold_object=kf)
```

IV. CATBOOST – IMPUTING with Shapely Explanator for FEATURE IMPORTANCE

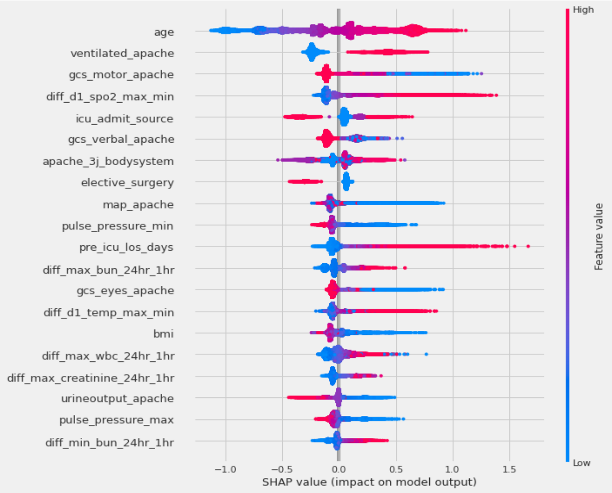




```
ROC_AUC score_Test: 0.8846716779164521
Brier_Score_Test:  0.056680594142657685
Accuracy Score_Test: 0.907
Classification Report_Test:
               precision    recall  f1-score   support

           0       0.96      0.94      0.95     20950
           1       0.47      0.54      0.50      1979

    accuracy                           0.91     22929
   macro avg       0.71      0.74      0.73     22929
weighted avg       0.91      0.91      0.91     22929
```



SHAPLEY EXPLANATOR

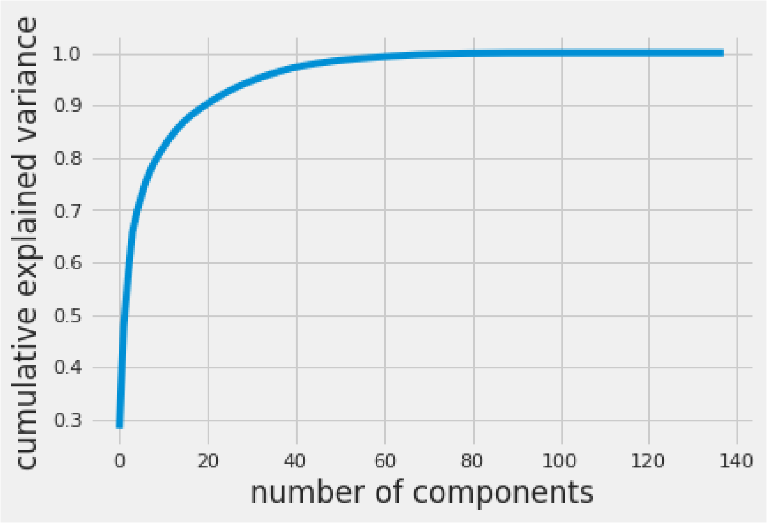


V. NEURAL NETWORK WITH PCA

Why using PCA for Neural Network?

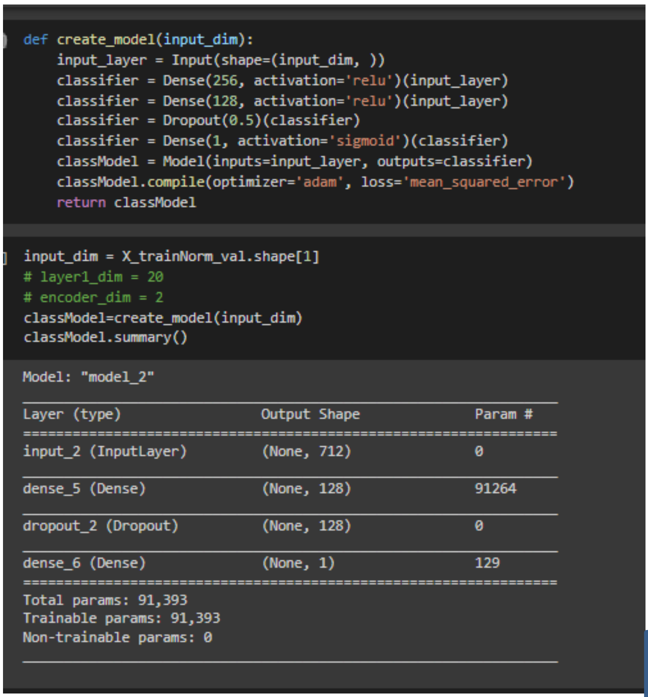
- Reduces computation complexity by reducing the size of the network, amount of data needed to train
- Reduce overfitting
- However, discriminative information that distinguishes the class might be in low variance components.

PCA-



Using 70 out of 138 Principal Components.

Neural Newtork Model:



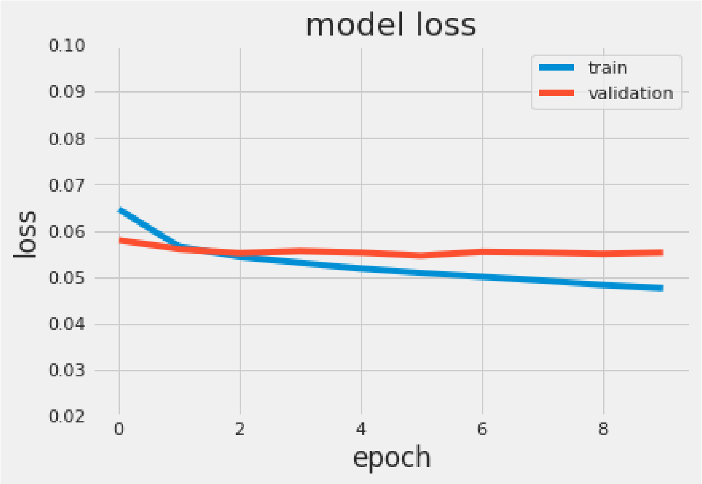
```
def create_model(input_dim):
    input_layer = Input(shape=(input_dim, ))
    classifier = Dense(256, activation='relu')(input_layer)
    classifier = Dense(128, activation='relu')(input_layer)
    classifier = Dropout(0.5)(classifier)
    classifier = Dense(1, activation='sigmoid')(classifier)
    classModel = Model(inputs=input_layer, outputs=classifier)
    classModel.compile(optimizer='adam', loss='mean_squared_error')
    return classModel
```

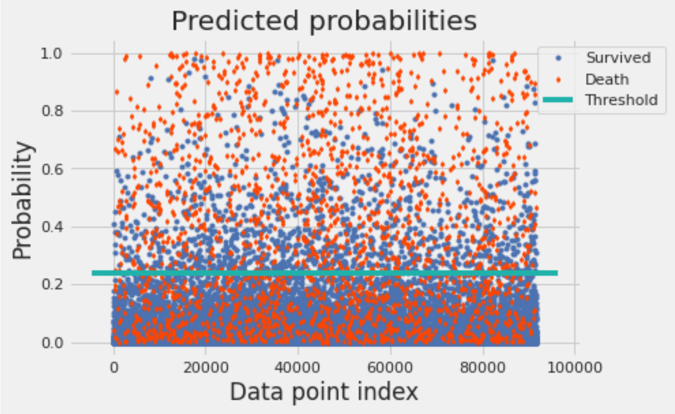
```
input_dim = X_trainNorm_val.shape[1]
# layer1_dim = 20
# encoder_dim = 2
classModel=create_model(input_dim)
classModel.summary()
```

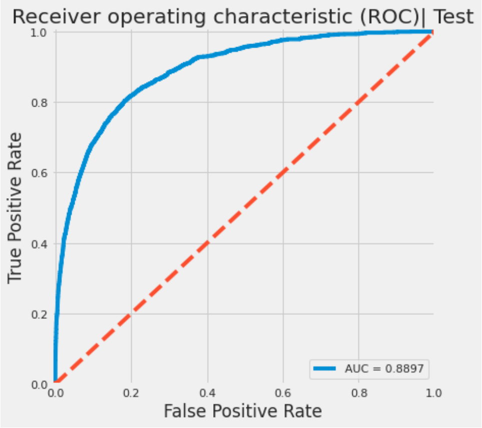
```
Model: "model_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 712)               0
_________________________________________________________________
dense_5 (Dense)              (None, 128)               91264
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 129
=================================================================
Total params: 91,393
Trainable params: 91,393
Non-trainable params: 0
_________________________________________________________________
```

91,393 Parameters are being used in this model.

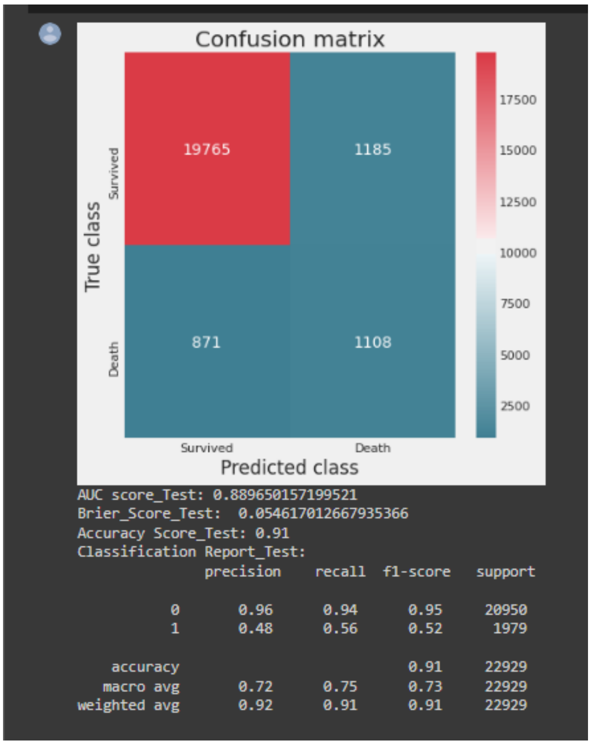
Train Vs Validation Loss





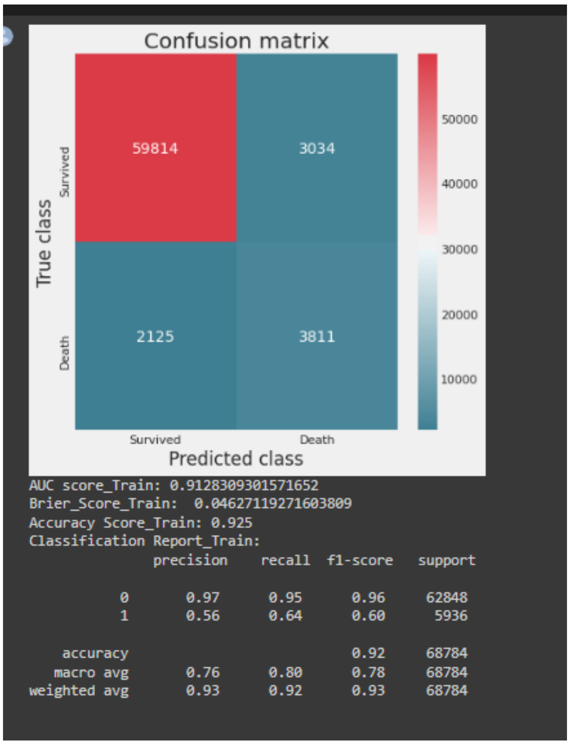


Classification Report on Test:



```
AUC score_Test: 0.889650157199521
Brier_Score_Test:  0.054617012667935366
Accuracy Score_Test: 0.91
Classification Report_Test:
               precision    recall  f1-score   support

           0       0.96      0.94      0.95     20950
           1       0.48      0.56      0.52      1979

    accuracy                           0.91     22929
   macro avg       0.72      0.75      0.73     22929
weighted avg       0.92      0.91      0.91     22929
```

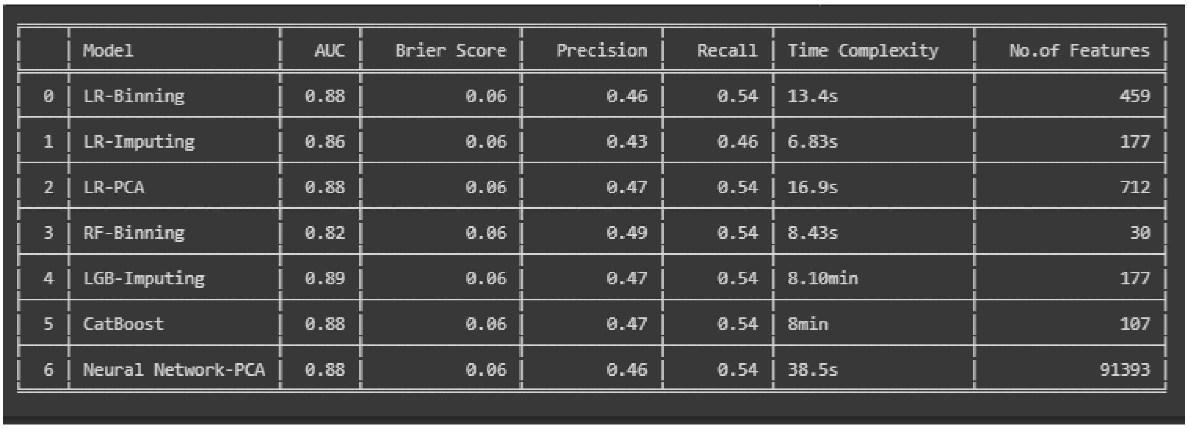
Classification Report on Train:



```
AUC score_Train: 0.9128309301571652
Brier_Score_Train:  0.04627119271603809
Accuracy Score_Train: 0.925
Classification Report_Train:
               precision    recall  f1-score   support

           0       0.97      0.95      0.96     62848
           1       0.56      0.64      0.60      5936

    accuracy                           0.92     68784
   macro avg       0.76      0.80      0.78     68784
weighted avg       0.93      0.92      0.93     68784
```

V. CONCLUSION

Model Comparison and Selection



```
|   | Model              |   AUC |   Brier Score |   Precision |   Recall | Time Complexity   |   No.of Features |
|---+--------------------+-------+---------------+-------------+----------+-------------------+------------------|
| 0 | LR-Binning         |  0.88 |          0.06 |        0.46 |     0.54 | 13.4s             |              459 |
| 1 | LR-Imputing        |  0.86 |          0.06 |        0.43 |     0.46 | 6.83s             |              177 |
| 2 | LR-PCA             |  0.88 |          0.06 |        0.47 |     0.54 | 16.9s             |              712 |
| 3 | RF-Binning         |  0.82 |          0.06 |        0.49 |     0.54 | 8.43s             |               30 |
| 4 | LGB-Imputing       |  0.89 |          0.06 |        0.47 |     0.54 | 8.10min           |              177 |
| 5 | CatBoost           |  0.88 |          0.06 |        0.47 |     0.54 | 8min              |              107 |
| 6 | Neural Network-PCA |  0.88 |          0.06 |        0.46 |     0.54 | 38.5s             |            91393 |
```

- From the above Model Comparison Table, we can conclude that.
  - Neural Network with Principal Component Analysis has best performance in terms of AUC, Brier Score, Precision, Recall.
  - But In order to predict mortality rate considering AUC, Brier Score, Precision, Recall is not just sufficient. We need to consider time complexity and see which is the most generalized model.
  - Such a model with less time complexity and more generalized compared to existing APACHE IV scoring prediction system will be considered to be the best Intensive Critical care mortality prediction model for our project.
  - Hence for the Smart-End version of Intensive Critical Care Mortality Prediction, we should build Logistic Regression – Binning Model since it has less time complexity and it is more generalized compared to other models.
  - Hence the best model applicable in Hospitals would be Logistic Regression-Binning Model.

- In this project we have developed a smart-end version for Intensive Critical Care Mortality Prediction System using various Machine Learning Algorithms considering all the dependent features and existing APACHE scoring system features and came up with an effective model compared to existing APACHE-IV prediction model which is turned out to Logistic Regression – Binning Model. This Model negotiates the non-vital features of existing mortality rate prediction system which is APACHE IV in order to achieve better accuracy in predictions. Hence the better the accurate predictions the system predicts the better the scope of delivering effective treatment services to the patient and save their lives

Future Scope:

In this project we have build a samrt end version for Intensive Critical Care Mortality Prediction using Various Machine Learning models and selected the best model which has better time complexity and more generalized along with better performance in terms of AUC, Brier-Score, Precision & Recall. This Prediction System has higher performance accuracy compared to existing APACHE-IV System.The accurate and effective mortality risk prediction system could help doctors analyze the severity in the patients priorly, identify earlier interventions and and take necessary steps to avail potentially better outcomes. It would be more better if doctors are able to provide personalized treatment. The future scope of this project is to develop a recommendation system that could recommend doctors a suitable diagnosis and treatments based on the reports, existing chronic diseases and lab results of the patient admitted in Hospital or ICU.

References:

[1]Suresh, K. , Severn, C. & Ghosh D.(2022) 'Survival prediction models: an introduction to discrete-time modeling', 'BMC Medical Research Methodology'

[2]Thorsen-Meyer, HC., Placido, D., Kaas-Hansen, B.S. et al. (2022),' Discrete-time survival analysis in the critically ill: a deep learning approach using heterogeneous data',' npj Digital Medicine'.

[3]Choi, M.H., Kim, D., Choi, E.J. et al,(2022),' Mortality prediction of patients in intensive care units using machine learning algorithms based on electronic health records',' Scientific Reports'.

[4]Chia Alvin, Khoo May, Lim Andy. et al,(2021),' Explainable machine learning prediction of ICU mortality',' Informatics in Medicine Unlocked Published by Elsevier '.

[5]Semagn Mekonnen Abate, Sofia Assen, Mengistu Yinges, Bivash Basu,(2021),' Survival and predictors of mortality among patients admitted to the intensive care units in southern Ethiopia: A multi-center cohort study',' Science Direct'.

[6]Spooner, A., Chen, E., Sowmya, A. et al,(2021),' A comparison of machine learning methods for survival analysis of high-dimensional clinical data for dementia prediction',' Scientific Reports'.

[7]Feng, S., Dubin, J.A,(2021),' Identifying early-measured variables associated with APACHE IVa providing incorrect in-hospital mortality predictions for critical care patients',' Scientific Reports'.

[8]Feng, Shuo & Dubin, Joel,(2020),' APACHE IV is an accurate measure of predicting in-hospital mortality of ICU patients, but there are covariates associated with its occasional failure - a multicentre analysis of intensive care units',' Research Square'.

[9]Annelaura B Nielsen, Hans-Christian Thorsen-Meyer, et al,(2019),' Survival prediction in intensive-care units based on aggregation of long-term disease history and acute physiology: a retrospective study of the Danish National Patient Registry and electronic patient records',' Lancet Digital Health'.

[10]Venkataraman R, Gopichandran V, et al.,(2019),' Mortality prediction using Acute Physiology and Chronic Health Evaluation II and Acute Physiology and Chronic Health Evaluation IV scoring systems: Is there a difference?',' Indian Journal of Critical Care Medicine'.

[11]Baran Balkan, Patrick Esay, Vignesh Subbian.(2018),' Evaluating ICU Clinical Severity Scoring Systems and Machine Learning Applications: APACHE IV/IVa Case Study',' International Conference of the IEEE Engineering in Medicine and Biology Society'

[12] Jae Woo Choi1, Young Sun Park2, Young Seok Lee3, et al,(2017), 'The Ability of the Acute Physiology and Chronic Health Evaluation (APACHE) IV Score to Predict Mortality in a Single Tertiary Hospital',' Korean J Crit Care Med'

[13] Aura T. Ylimartimo, Marjo Koskela, Sanna Lahtinen, Timo Kaakinen, et al.(2022), 'Outcomes in patients requiring intensive care unit (ICU) admission after emergency laparotomy: A retrospective study', 'International Journal of anesthesiology, intensive care, pain & critical emergency medicine'

[14] Colantuoni, E., Koneru, M., Akhlaghi, N. et al.(2021), 'Heterogeneity in design and analysis of ICU delirium randomized trials: a systematic review', 'SpringerLink'.

[15] Nader Markazi-Moghadda m, Mohammad Fathi, Azra Ramezankhani.(2020), 'Risk prediction models for intensive care unit readmission: A systematic review of methodology and applicability', 'Australian Critical Care'.

[16] Iwase, S., Nakada, Ta., Shimada, T. et al.(2022), 'Prediction algorithm for ICU mortality and length of stay using machine learning', 'Scientific Reports'.

[17] Hirano, Y., Kondo, Y., Hifumi, T. et al.(2021), 'Machine learning-based mortality prediction model for heat-related illness', 'Scientific Reports'.

[18] Cetin Kaymak, Irfan Sencan,Seval Izdes, et al.(2018), 'Mortality of adult intensive care units in Turkey using the APACHE II and SOFA systems (outcome assessment in Turkish intensive care units)', 'Archieves of Medical Science'.

[19] Aya Awad, Mohamed Bader-El-Den, et al.(2019), 'Predicting hospital mortality for intensive care unit patients: Time-series analysis', 'Health Informatics Journal'.