Prediction of the likelihood of a patient requiring admission to the ICU

**Project Overview:**

Predicting the likelihood of a patient requiring admission to the Intensive Care Unit (ICU) is a solution to the problem of identifying the patient beforehand so that resource allocation and bed management can be done effectively. It is the critical aspect of healthcare management, especially in hospitals and emergency departments.

Predicting the likelihood of a patient requiring admission to the Intensive Care Unit (ICU) is done based on historical hospital-wide electronic health record (EHR) data. It involves leveraging the vast amount of information contained within records to identify patterns and trends associated with ICU admissions. This prediction involves assessing various factors related to the patient's condition, medical history, and laboratory results to determine the severity of illness and the need for intensive care interventions.

**Objective:**

The aim of this project is to enhance resource allocation and optimize bed management by identifying patients at higher risk of ICU admission upon hospital admission.

**Data Source:**

We have used MIMIC-IV Version 2.2 dataset. It is sourced from two in-hospital database systems: a custom hospital wide EHR and an ICU specific clinical information system.

**Data description:**

The hosp module stores information regarding patient transfers, billed events, medication prescription, medication administration, laboratory values, microbiology measurements, and provider orders. The subject\_id column is present in all tables and allows linkage to patient demographics in the patients table. The hadm\_id column is also present in all tables and represents a single hospitalization; rows without an hadm\_id pertain to data collected outside of an inpatient encounter. Most tables may be interpreted without cross-linking to other tables. Tables which contain item\_ids are an exception, and they must be linked to a dimension table prefxed with d\_ to acquire a human interpretable description of the item\_ids. Other tables, such as emar and poe, may be linked with “detail” tables (emar\_detail and poe\_detail) which provide additional information for each row. Fig.01 give summarize view of the hosp module.

A screenshot of a medical report

Description automatically generated

**Fig 01: Summary view of Hospital dataset**

The icu module includes chartevents, d\_items, datetimeevents, icustays, inputevents, outputevents, and procedureevents. The icu module adopts a star schema, with all event tables referencing d\_items for defining itemid and icustays for defning stay\_id. Te stay\_id column is a primary key for the icustays table, and as such is unique for each row. ICU stays are defined using the administrative record of patient movement within the hospital, i.e. the icustays table is derived from the transfers table in the hosp module.

**Data Preparation:**

We have gone through all the dataset present in hospital and came to conclusion that attribute from **admission, patient and diagnosis** are useful for our problem statement. So, we merge the admission dataset with patient dataset on basis of subject\_id. By merging with this we get patient gender and age in admission data. Then we merge diagnosis of each subject\_id against each hospital admit by using hadm\_id.

After collecting all dataset at one place we started exploring each attribute to understand it importance. In admission dataset there are many attributes which is categorical like admission location, admission type, race etc so we analyze and tried to reduced the number of categories in each attribute.

A white rectangular object with a black border

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

All diagnosis code in diagnosis\_icd dataset is mapped to following diagnosis with respect to patient.

* Infectious and Parasitic Diseases
* Neoplasms
* Endocrine, Nutritional, and Metabolic Diseases
* Diseases of the Blood and Blood-forming Organs
* Mental Disorders
* Diseases of the Nervous System and Sense Organs
* Diseases of the Circulatory System
* Diseases of the Respiratory System
* Diseases of the Digestive System
* Diseases of the Genitourinary System
* Complications of Pregnancy, Childbirth, and the Puerperium
* Diseases of the Skin and Subcutaneous Tissue
* Diseases of the Musculoskeletal System and Connective Tissue
* Congenital Anomalies
* Certain Conditions Originating in the Perinatal Period
* Symptoms, Signs, and Ill-defined Conditions
* Injury and Poisoning
* External causes of injury and supplemental classification
* Codes for special purposes
* Diseases of the eye, adnexa, and mastoid process
* Factors influencing health status and contact with health services.

By going through problem statement, we have identified and added two new, feature in the dataset using the admit and discharge time of patient. These two attributes are the count of hospitalization and length of stay.

**Target variable:** As our problem statement is classification problem and target variable indicating whether that subject\_id get admitted to ICU is not directly mentioned so we add new feature in table icu\_admit. This attribute is generated using the hadm\_id. For each hospitalization in icu module corresponding hadm\_id is matched and marked true.

**Raw data Exploratory Data Analysis (EDA):** Refer to the EDA file.

**Feature Selection:** Features are chosen based on their relevance to the target variable or their potential impact on the model's performance.



**Modeling:**

Before modeling Data is split into training and testing sets using the `train\_test\_split` function from scikit-learn. This helps evaluate the model's performance on unseen data.

Several classifiers are trained and tested which are mentioned below:

* Random Forest
* Logistic Regression
* XGBoost
* Naive Bayes
* Decision Tree
* K-Nearest Neighbors

**Evaluation:**

The trained classifier is then used to make predictions on the testing data (`X\_test`). Performance metrics such as accuracy and classification report are calculated and printed for each classifier.

The accuracy score measures the proportion of correctly predicted instances. The classification report provides precision, recall, F1-score, and support for each class in the target variable. This gives insight into the model's performance for each class, particularly useful in multi-class classification problems.

**Base model results:**

Among all models trained and tested model XGboost has performed best with accuracy of 88.31 % and second-best model is random forest with accuracy of 87.93%. All other evaluation parameters like precision and recall is also calculated and is mention for XGBoost and Random Forest in the following figure.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Hyperparameter tunning:** Hyperparameter tunning is done on top performing model i.e. XGBoost and Random Forest to improve the accuracy of model. By doing hyperparameter tunning we have achieved a slight increase in performance metric of the model. Below figure shows the achieved result.

**A white paper with black text

Description automatically generated**

**Pickle and Preprocessing file**

After completing model testing Pickle file of hyper parameterized Xgboost model is created and preprocessing file is created which will help in deployment of model on stream lit.