**Prediction of severity of Patient in Emergency Department**

**Problem statement**

Predicting the severity of patients in the Emergency Department (ED) for optimizing patient outcomes, resource allocating and hospital efficiency.

**Data Source**

Dataset use isMIMIC-IV-ED. It is a large, freely available database of emergency department (ED) admissions at the Beth Israel Deaconess Medical Center between 2011 and 2019. The database contains ~425,000 ED stays. Vital signs, triage information, medication reconciliation, medication administration, and discharge diagnoses are available.

**Dataset Description**

MIMIC-IV-ED is composed of a single patient tracking table, edstays, and five data tables: diagnosis, medrecon, pyxis, triage, and vitalsign. With respect to our problem, we get all essential features in triage dataset.

The **triage table** provides information collected from the patient at the time of triage. All patients who are present to the ED are immediately triaged, a process which involves assessing their health status and ascertaining the reason for their visit. The triage table has eleven columns: subject\_id, stay\_id, temperature, heartrate, resprate, o2sat, sbp, dbp, pain, acuity, and chiefcomplaint. Vital signs collected at triage include patient temperate (temperature), heart rate (heartrate), respiratory rate (resprate), oxygen saturation (o2sat), systolic blood pressure (sbp), and diastolic blood pressure (dbp). Although vital signs can be documented as free text, the deidentification approach retained only numeric vital signs. A patient reported pain level is available in the pain column. The chiefcomplaint is a free-text field which contains the patientʼs reported reason for presenting to the ED.

**Data Preparation:**

In the dataset there are 425087 rows and 11 columns. The pain feature in dataset is object type and it contains both string and numerical value. We treat it by converting pain to integer and remove the string values. The chief complaint attribute is dropped as it contains only description in words. As the data contains null value in each column it is dropped. The reason for dropping the null value is due to sensitivity of medical data. Duplicate rows are dropped.

**EDA:** Refer to the EDA file.

**Feature selection**

As the number of features is limited in number so we have taken all the features for training except chief complaint is dropped.

features= [ 'temperature', 'heartrate', 'resprate', 'o2sat', 'sbp', 'dbp', 'pain']

**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 60.2%. All other evaluation parameters like precision and recall is also calculated and is mention for XGBoost in following figure.

A screenshot of a computer screen

Description automatically generated

**Hyperparameter Tunning**

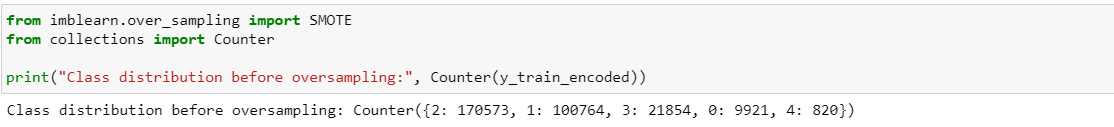
After completing the model hyperparameter tuning is done to improve the output of the model. Different parameters like learning rate, n-estimator, max depth etc. were used. After applying these parameters, we have not achieved significant improvement in the results. Figure show the result of XGboost classifier.

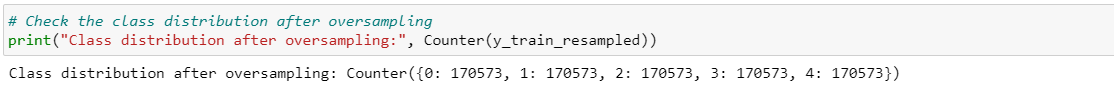
A screenshot of a computer

Description automatically generated

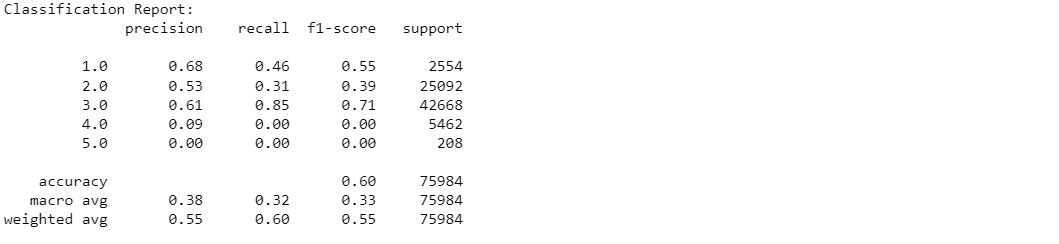
**Oversampling of Target class**

Since our target class is imbalanced so we try to balance the target class using the smote. Figure shows the distribution of target class before and after the balancing.

****

****

After applying oversampling, we found that there is not much improvement in the performance parameters.



**Ensemble Technique:** We have also tried ensemble technique by combining the output of best performing classifier, but we didn’t get significant improvement in the result. Classifier used in formed combination are: -

1. Logistic Regressor
2. XGB Classifier
3. Gradient Boosting Classifier
4. Random Forest Classifier

Following figure show the result of Logistic Regressor, XGB Classifier and Gradient Boosting Classifier.

A close-up of a white background

Description automatically generated

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