**Automated Diagnostic Assistance in Emergency Department**

* **Problem statement:**

Enhancing the accuracy and efficiency of diagnoses in the ED can lead to better patient outcomes

* **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 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 will get all essential features in triage and diagnosis table.

* 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.
* The **diagnosis table** provides coded diagnoses for the patient in the International Classification of Diseases (ICD) Ninth or Tenth revision (ICD-9 or ICD-10). These diagnoses are determined by trained coders after discharge from the emergency department and are used for billing purposes. There are six columns in the diagnosis table: subject\_id, stay\_id, seq\_num, icd\_code, icd\_version, and icd\_title. A maximum of 9 ICD codes are available for a single stay. The seq\_num column provides a pseudo-order for the ICD codes, with a value of 1 usually indicating highest relevance and a value of 9 indicating least relevance. The icd\_code provides the coded representation of the diagnosis using the ICD ontology, the icd\_version column is either 9 or 10 indicating whether the ontology used is ICD-9 or ICD-10, and the icd\_title column provides the textual description of the ICD code.
* **Raw data Exploratory Data Analysis (EDA):**

Various steps were taken to get the basic understanding of the raw data and to find the relation between the target class and other columns present. Refer to EDA file present in the GitHub repository.

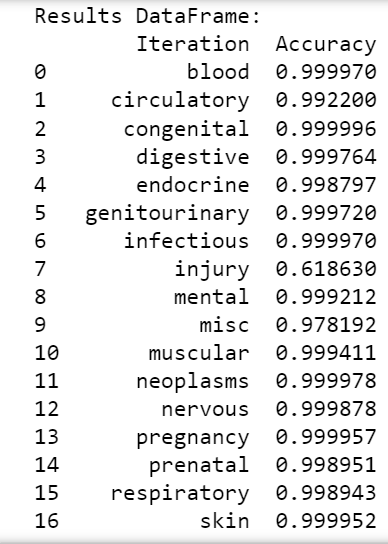
* **Dataset Preparation:**
* We performed min-max scaling which is a normalization technique to normalize the columns present in the triage table, then we removed nulls from those columns.
* In the diagnosis table first we mapped the icd-codes to the diagnosis names and then using dummy we explode 17 diagnosis into 17 new columns named by the diagnosis name.
* Then we merged both the tables and dropped columns which were not required in the process. Refer to the preprocessing file present in the GitHub repository.
* Size of the final dataset was 392620 rows and 25 columns.
* **Model-Selection:**

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. Random Forest Classifier is trained and tested in this problem statement.

* **Model-Evaluation:**

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

* **Model Working & Results:**
* For model training and testing we split the data into 70% for training and 30% for the testing. Input variables were stay\_id, temperature, heartrate, resprate, o2sat, sbp, dbp and total 17 target column named with diagnosis names were there containing 0 and 1 , so we initated a for loop in out model to iterate over each column to get best results and after the data pass through one column the model will give results for that particular disease. So at the end we got 17 different models in our case. Results of each model is presented below which contains the accuracy of each model.
* After getting 17 different pickle files of each model we put all of them in a dictionary and pass a input to the dictionary and it will again iterate over each model and which diseases having threshold greater than 70% will come as output for our model. Pickle files are uploaded in the google drive and link is also provided below.

****

* **Pickle and Preprocessing file:** After completing model testing all the Pickle file of model and preprocessing file is created which will help in deployment of model on stream lit, preprocessing file for this model is presented in the GitHub repository and for the pickle files go the drive link provided below: -

(<https://drive.google.com/file/d/1CmYIUPFFrS1KbWG5rVhPmA4ubm1VBJmi/view?usp=sharing>)