Project report on

# "Healthcare Analytics"

Submitted towards partial fulfillment of the criteria

for award of PGP-DSE by Great Lakes Institute of Management

*Submitted By*
Group No. 1 Batch: 2019

*Group Members*

**Hemanth Bugga**

**Lavina Dsouza**

**Makarand P Batchu**

**Vishal Gudla**

*Research Supervisor*
Mr.Srikar Muppidi

GREAT LAKES INSTITUTE OF MANAGEMENT
Global Mindset - Indian Roots

**Great Lakes Institute of Management**

## Abstract

Management of hyperglycemia in hospitalized patients has a significant bearing on outcome, in terms of both morbidity and mortality. However, there are few national assessments of diabetes care during hospitalization which could serve as a baseline for change. This analysis of a large clinical database (74 million unique encounters corresponding to 17 million unique patients) was undertaken to provide such an assessment and to find future directions which might lead to improvements in patient safety. Almost 70,000 inpatient diabetes encounters were identified with sufficient detail for analysis. Multivariable logistic regression was used to fit the relationship between the measurement of HbA1c and early readmission while controlling for covariates such as demographics, severity and type of the disease, and type of admission. Results show that the measurement of HbA1c was performed infrequently (18.4%) in the inpatient setting. The statistical model suggests that the relationship between the probability of readmission and the HbA1c measurement depends on the primary diagnosis. The data suggest further that the greater attention to diabetes reflected in HbA1c determination may improve patient outcomes and lower cost of inpatient care.

**Keywords –** Diabetic patients, HbA1c Result, Readmission, Solo Insulin or Combination of Drugs.

- **Techniques:** Machine Learning – Supervised Learning Classification
- **Tools:** Python, Excel, Tableau, Seaborn, Matplotlib, scikit learn.
- **Domain:** Predictive Analytics

## Certificate of completion

I hereby certify that the project titled **“Healthcare Analytics”** for case resolution was undertaken and completed under my supervision by **Hemanth Bugga, Lavina Dsouza, Makarand P Batchu, Vishal Gudla** of PGP in Data Science and Engineering (PGP-DSE).

**Date:** 13th August, 2019

**Place:** Hyderabad **Mr.Srikar Muppidi**

## Declaration

I declare that the project entitled **"Healthcare Analytics"** is a project work carried out by me under the supervision and guidance of **Mr.Srikar Muppidi,** for the award of degree PGP DSE, and this has not been previously submitted for the award of any Degree, Diploma or other similar title of any other University/ Institute.

**Date:** 13th August, 2019

**Place:** Hyderabad

**Group members:**

Hemanth Bugga
Lavina Dsouza
Makarand Batchu
Vishal Gudla

## Acknowledgement

In any work there are contributions from all quarters. Since they cannot all be mentioned, I will name the few who have contributed the most.

I would like to acknowledge my supervisor **Mr.Srikar Muppidi** for providing me the necessary guidance and valuable support throughout this research project. I value the assistance of Great Learning, Hyderabad campus. Learning from their knowledge helped me to become passionate about my research topic. I will be failing in my duty if I don’t express my gratitude for my team members, for their valuable suggestions and cheerful assistance during the implementation phase of this project.

## Table of Contents

## 1. Introduction

It is increasingly recognized that the management of hyperglycemia in the hospitalized patient has a significant bearing on outcome, in terms of both morbidity and mortality [1, 2]. This recognition has led to the development of formalized protocols in the intensive care unit (ICU) setting with rigorous glucose targets in many institutions [3]. However, the same cannot be said for most non-ICU inpatient admissions. Rather, anecdotal evidence suggests that inpatient management is arbitrary and often leads to either no treatment at all or wide fluctuations in glucose when traditional management strategies are employed. Although data are few, recent controlled trials have demonstrated that protocol-driven inpatient strategies can be both effective and safe [4, 5]. As such, implementation of protocols in the hospital setting is now recommended [6, 7]. However, there are few national assessments of diabetes care in the hospitalized patient which could serve as a baseline for change. The present analysis of a large clinical database was undertaken to examine historical patterns of diabetes care in patients with diabetes admitted to a US hospital and to inform future directions which might lead to improvements in patient safety. In particular, we examined the use of HbA1c as a marker of attention to diabetes care in a large number of individuals identified as having a diagnosis of diabetes mellitus. We hypothesize that measurement of HbA1c is associated with a reduction in readmission rates in individuals admitted to the hospital.

Databases of clinical data contain valuable but heterogeneous and difficult data in terms of missing values, incomplete or inconsistent records, and high dimensionality understood not only by number of features but also their complexity. [8]. Additionally, analyzing external data is more challenging than analysis of results of a carefully designed experiment or trial, because one has no impact on how and what type of information was collected. Nonetheless, it is important to utilize these huge amounts of data to find new information/knowledge that is possibly not available anywhere.

### 1.1 Problem statement

1. Find the efficiency of Insulin and conjunction of drugs.
2. Given the patients details, predict if the patient needs to be prescribed with Insulin or conjunction of drugs.

### 1.2 Research purpose

This study used the Health Facts database (Cerner Corporation, Kansas City, MO), a national data warehouse that collects comprehensive clinical records across hospitals throughout the United States. Health Facts is a voluntary program offered to organizations which use the Cerner Electronic Health Record System. The database contains data systematically collected from participating institutions electronic medical records and includes encounter data (emergency, outpatient, and inpatient), provider specialty, demographics (age, sex, and race), diagnoses and in-hospital procedures documented by ICD-9-CM codes, laboratory data, pharmacy data, in-hospital mortality, and hospital characteristics. All data were deidentified in compliance with the Health Insurance Portability and Accountability Act of 1996 before being provided to the investigators. Continuity of patient encounters within the same health system (EHR system) is preserved.

The Health Facts data we used was an extract representing 10 years (1999–2008) of clinical care at 130 hospitals and integrated delivery networks throughout the United States: Midwest (18 hospitals), Northeast (58), South (28), and West (16). Most of the hospitals (78) have bed size between 100 and 499, 38 hospitals have bed size less than 100, and bed size of 14 hospitals is greater than 500.

The database consists of 41 tables in a fact-dimension schema and a total of 117 features. The database includes 74,036,643 unique encounters (visits) that correspond to 17,880,231 unique patients and 2,889,571 providers. Because this data represents integrated delivery network health systems in addition to stand-alone hospitals, the data contains both inpatient and outpatient data, including emergency department, for the same group of patients. However, data from out-of-network providers is not captured.

The dataset was created in two steps. First, encounters of interest were extracted from the database with 55 attributes. This dataset is available as a Supplementary Material available online at http://dx.doi.org/10.1155/2014/781670 and is also in the process of submission to the UCI Machine Learning Repository [9] so that it is easily available to other researchers.

Second, preliminary analysis and pre-processing of the data were performed resulting in retaining only these features (attributes) and encounters that could be used in further analysis, that is, contain sufficient information. Both steps are described in the following subsections.

## 1.3 Research approach and strategy

There are two most commonly used research approaches the inductive and the deductive method. The inductive research method attempts to setup a theory by using collected data while the deductive research approach attempts to find the theory first and then test it to the observed data. I chose a deductive research approach for my study as I would move from the more general to the specific. I will present the theoretical findings on diabetic patients in the next chapter after which I will present my questionnaire in chapter four where I present my collected primary data.

When collecting data to approach the purpose of a research there are two ways in which the data can be collected. In order to acquire a general knowledge about the topic secondary data is primarily used and is one of the ways by which data can be collected. These Conway to collect data is the primary data collection. Usually when a study is conducted secondary data is not sufficient enough and needs to be completed with primary data which is collected by the researcher.

## 2. Literature review

### 2.1 Classification models

To determine what classification scheme fit our data the best, we implemented several of the most common classifiers used for supervised learning. To implement the classification, we made use of the machine learning Library using Python called Scikit- Learn. The following are the classification models we employed due to their suitability for binary classification problems:

#### 2.1.1 Logistic regression

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. Sometimes logistic regressions are difficult to interpret; the Intellects Statistics tool easily allows you to conduct the analysis, and then in plain English interprets the output. The dependent variable should be dichotomous in nature (e.g., presence vs. absent).

There should be no outliers in the data, which can be assessed by converting the continuous predictors to standardized scores, and removing values below -3.29 or greater than 3.29. There should be no high correlations (multicollinearity) among the predictors. This can be assessed by a correlation matrix among the predictors. At the centre of the logistic regression analysis is the task estimating the log odds of an event. Mathematically, logistic regression estimates a multiple linear regression function defined as:

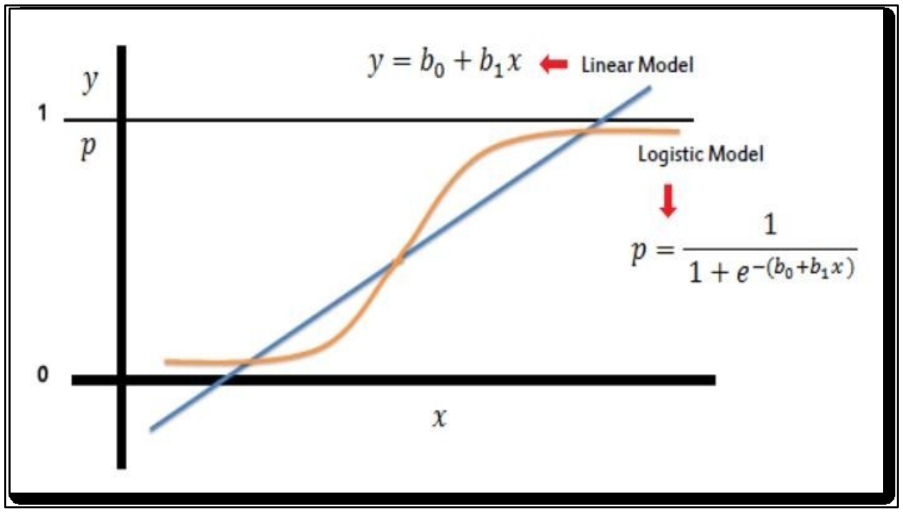


**Fig 2.1: Sigmoid Activation Function**

### 2.1.2 Decision tree

Decision Trees are trees that classify instances by sorting them based on feature values. Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. Instances are classified starting at the root node and sorted based on their feature values. Decision tree learning, used in data mining and machine learning, uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. More descriptive names for such tree models are classification trees or regression trees. Decision tree classifiers usually employ post-pruning techniques that evaluate the performance of decision trees, as they are pruned by using a validation set. Any node can be removed and assigned the most common class of the training instances that are sorted to it.

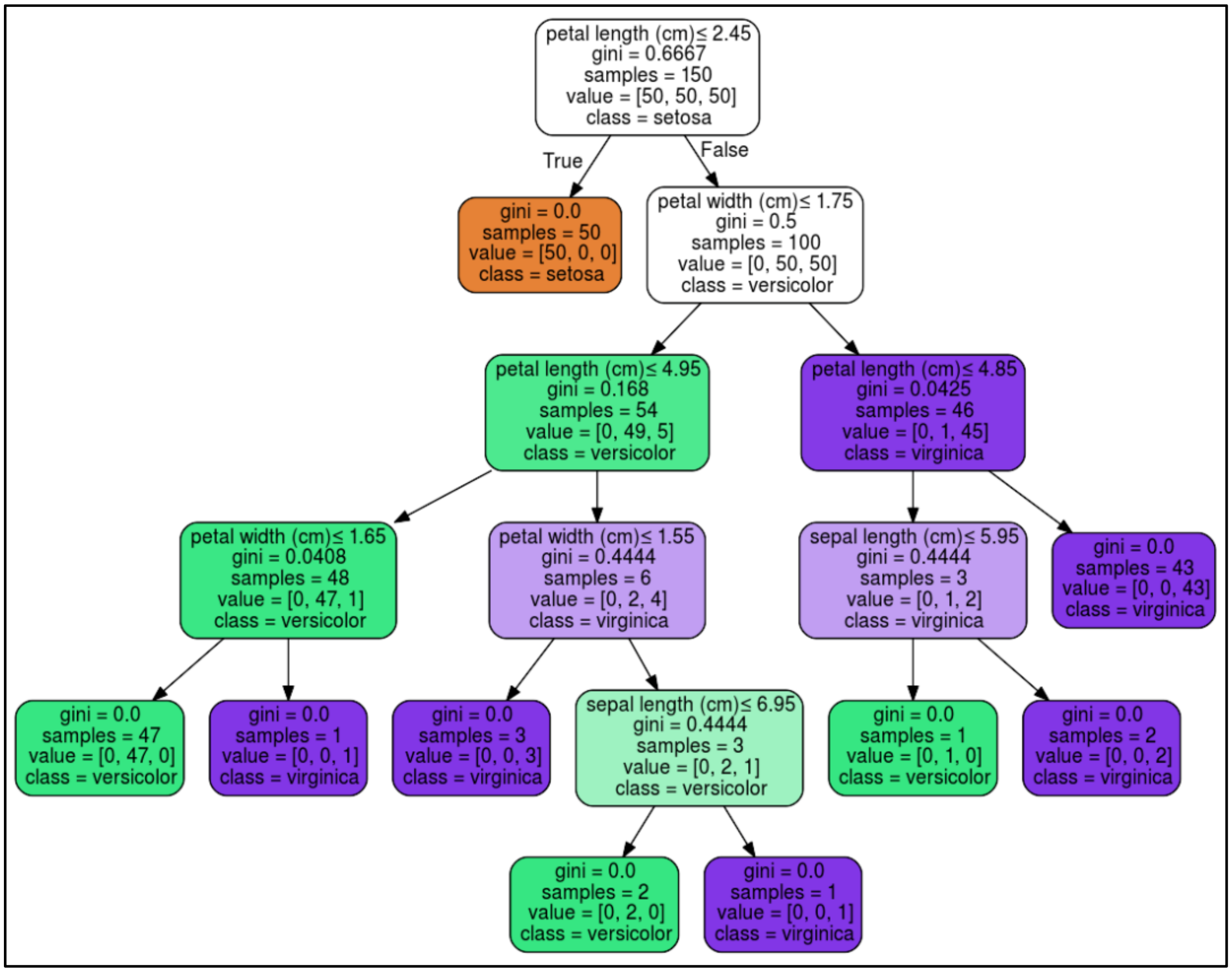


**Fig 2.2: Classification in a basic decision tree proceeds from top to bottom.**

### 2.1.3 Random Forest

Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because its simplicity and the fact that it can be used for both classification and regression tasks. In this post, you are going to learn, how the random forest algorithm works and several other important things about it.

Random Forest is a supervised learning algorithm. Like you can already see from its name, it creates a forest and makes it somehow random. The forest it builds, is an ensemble of Decision Trees, most of the time trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result. To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Overall, Random Forest is a (mostly) fast, simple and flexible tool, although it has its limitations.

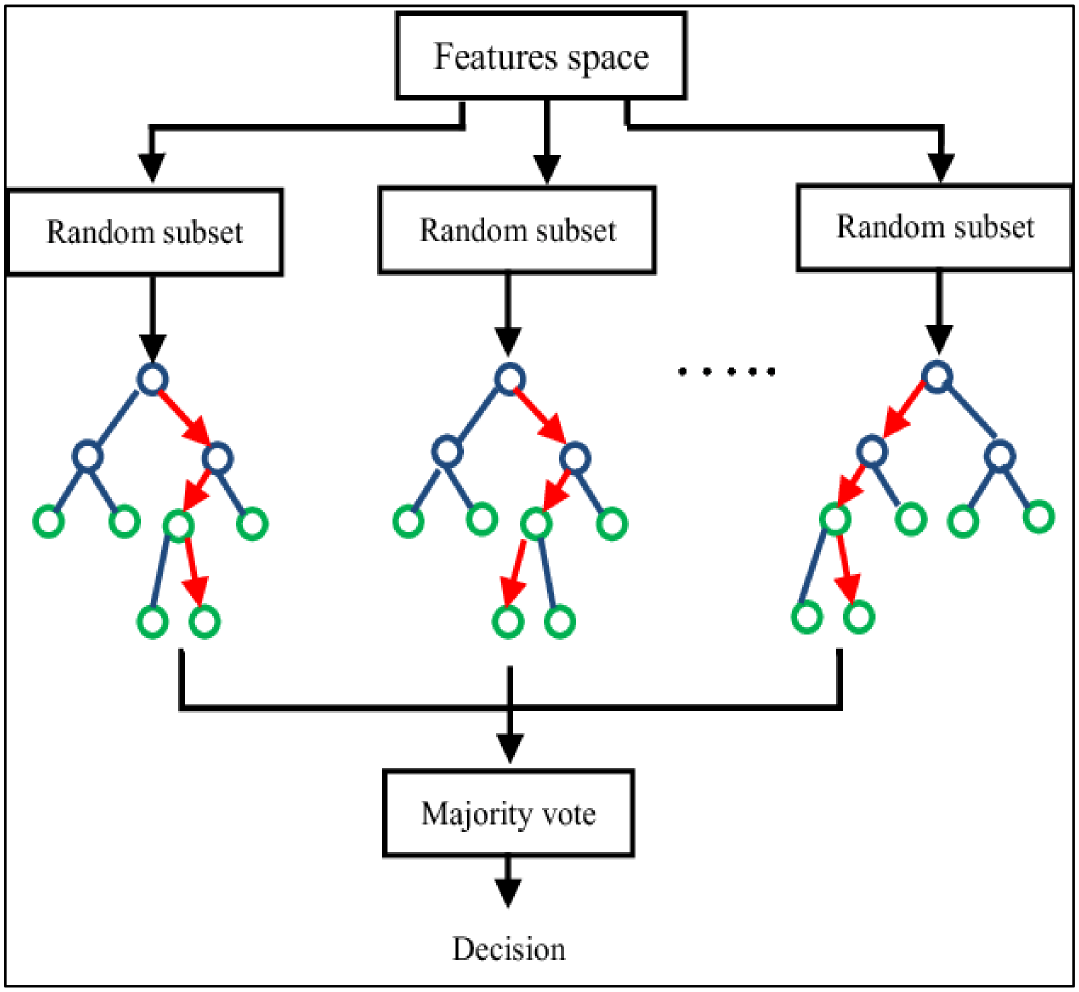


**Fig 2.3: Example of random forest tree**

### 2.1.4 Naïve Bayes Classifier

Naive Bayes classifier (NBC) is a parametric model that works on the principle of Bayes theorem. The algorithm is called Naive because of its assumption that all the features are independent to each other. BNC can also be trained on small datasets. As in most real-world scenarios, predictors are dependent at least upto some extent, NBC yields low performance models compared to other models. This can also be validated through our analysis in this project as NBC gave the least score. However, NBC works best in other applications such as recommendation systems, text mining etc.

### 2.1.5 K-Nearest Neighbours (KNN)

K Nearest neighbours (KNN) is a non-parametric model and is known as Lazy Learner. It does not learn or recognise patterns from the data but instead remembers the data on which it is trained. It calculates the distances from the target data point to all the data points in the training data set. Calculation of distance can be done in multiple ways viz, Manhattan, Euclidean and Minkowski. KNN, the number of nearest neighbours to consider is given as an input to the algorithm. The class of the majority data points among the considered data points is predicted as the class of target data point.

The below figure shows the working of KNN algorithm. The red dot is the target data point and yellow and purple dots represent different classes A and B respectively. When K=3 is given, target data point is assigned the majority class which is B. When K value is increased to 6, then the target data point is assigned the class B. Hence, KNN is highly dependent on the value of K.

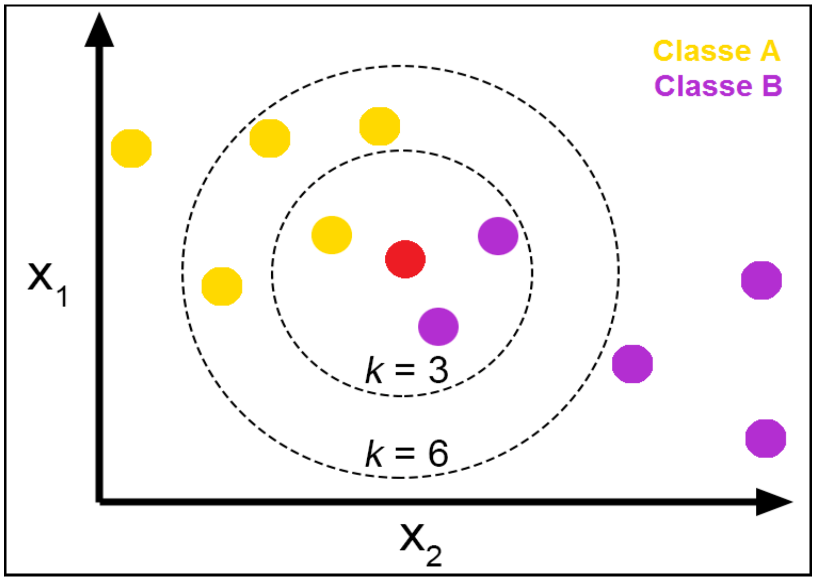


**Fig 2.4: K Nearest Neighbours (KNN)**

As the K value increases, model becomes underfit while K value is decreased, model becomes overfit. Optimal K value can be selected that gives the highest F1 score or accuracy or any other metrics. For example, in the case shown in the figure below, optimal K value is 5.

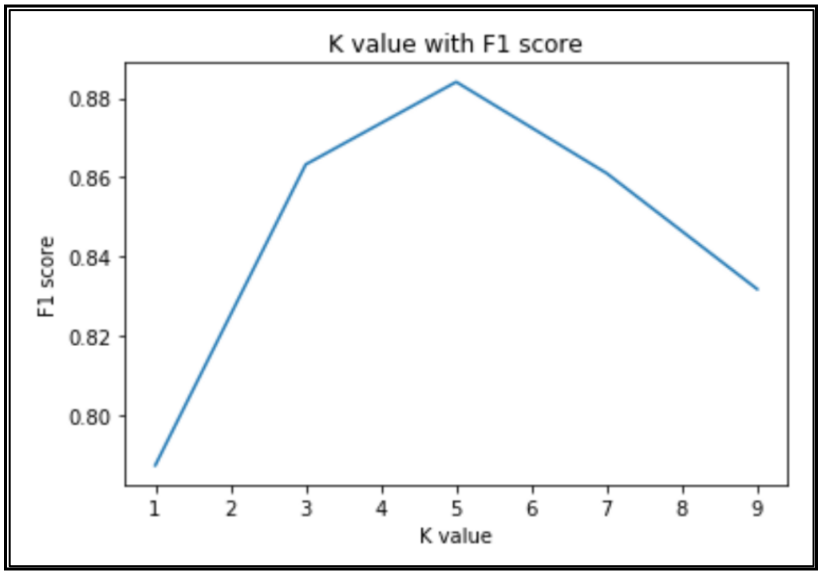


**Fig 2.5: K-value with F1-score**

### 2.1.6 Bagging (Bootstrap Aggregation) Classifier

Bootstrap Aggregation (or Bagging for short), is a simple and very powerful ensemble method. An ensemble method is a technique that combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model.

Bootstrap Aggregation is a general procedure that can be used to reduce the variance for those algorithms that have high variance. An algorithm that has high variance are decision trees, like classification and regression trees (CART). Decision trees are sensitive to the specific data on which they are trained. If the training data is changed (e.g. a tree is trained on a subset of the training data) the resulting decision tree can be quite different and in turn the predictions can be quite different.

Bagging is the application of the Bootstrap procedure to a high-variance machine learning algorithm, typically decision trees. Let's assume we have a sample dataset of 1000 instances (x) and we are using the CART algorithm. Bagging of the CART algorithm would work as follows.

1. Create many (e.g. 100) random sub-samples of our dataset with replacement.
2. Train a CART model on each sample.
3. Given a new dataset, calculate the average prediction from each model.

When bagging with decision trees, we are less concerned about individual trees overfitting the training data. For this reason and for efficiency, the individual decision trees are grown deep (e.g. few training samples at each leaf-node of the tree) and the trees are not pruned. These trees will have both high variance and low bias. These are important characterize of sub-models when combining predictions using bagging. Bagging tries to implement similar learners on small sample populations and then takes a mean of all the predictions. In generalized bagging, you can use different learners on different population. As you can expect this helps us to reduce the variance error.

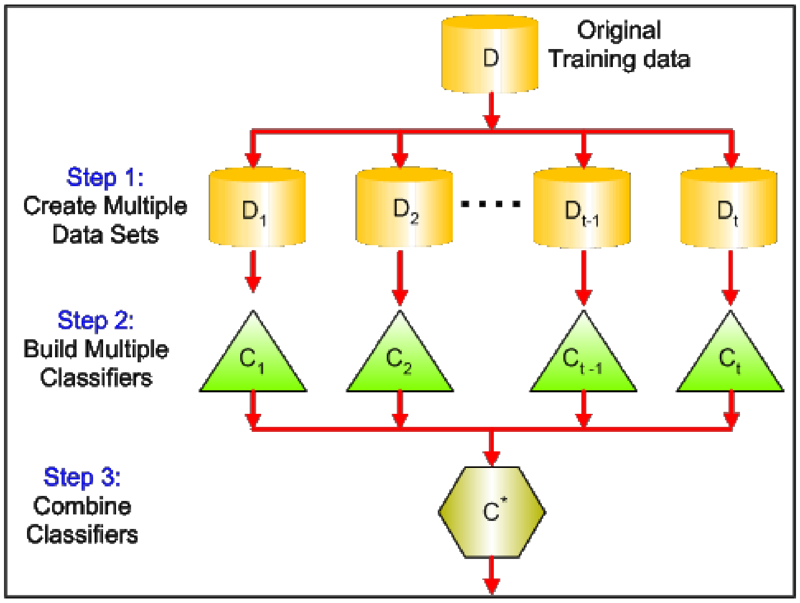

D
Original Training data
Step 1: Create Multiple Data Sets
D1
D2
Dt-1
Dt
Step 2: Build Multiple Classifiers
C1
C2
Ct-1
Ct
Step 3: Combine Classifiers
C*

**Fig 2.6: Bagging Classifier**

### 2.1.7 Voting Classifier

Voting is one of the simplest way of combining the predictions from multiple machine learning algorithms. Voting classifier isn't an actual classifier but a wrapper for set of different ones that are trained and valuated in parallel in order to exploit the different peculiarities of each algorithm.

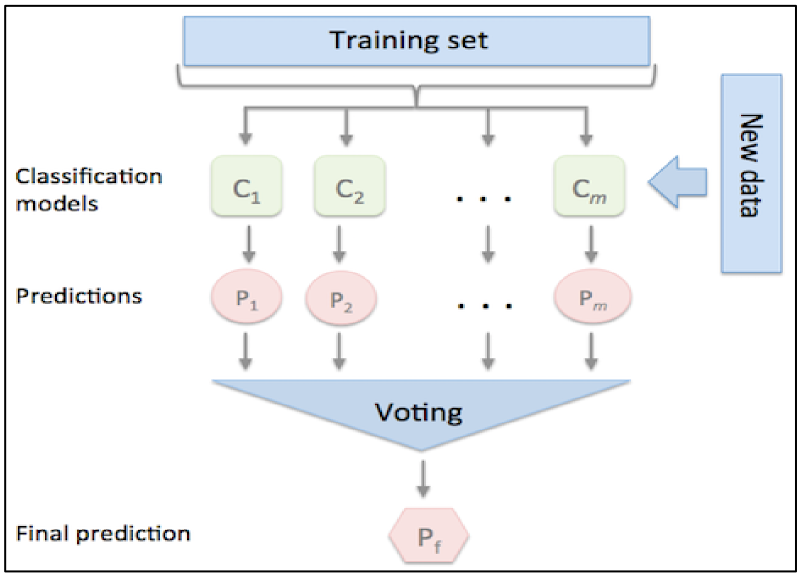


**Fig 2.7: Voting Classifier**

We can train data set using different algorithms and ensemble then to predict the final output. The final output on a prediction is taken by majority vote according to two different strategies:

**Hard voting / Majority voting**: Hard voting is the simplest case of majority voting. In this case, the class that received the highest number of votes Nc(yt) will be chosen. Here we predict the class label y^ via majority voting of each classifier.

Assuming that we combine three classifiers that classify a training sample as follows:

- classifier 1 -> class 0
- classifier 2 -> class 0
- classifier 3 -> class 1

y^=mode {0,0,1} = 0: - Via majority vote, we would we would classify the sample as "class 0".

**Soft voting**: In this case, the probability vector for each predicted class (for all classifiers) are summed up & averaged. The winning class is the one corresponding to the highest value (only recommended if the classifiers are well calibrated).

Assuming the example in the previous section was a binary classification task, our ensemble could make the following prediction:
C1(x)→[0.9,0.1]
C2(x)→[0.8,0.2]
C3(x)→[0.4,0.6]

However, assigning the weights {0.1, 0.1, 0.8} would yield a prediction y^=1

A voting classifier can be a good choice whenever a single strategy is not able to reach the desired accuracy threshold. In short voting classifier instead allows the mixing of different classifiers adopting a majority vote to decide which class must be considered as the winning one during a prediction.

## 2.2 Resampling Techniques

### 2.2.1 Random under sampling

Random Under sampling aims to balance class distribution by randomly eliminating majority class examples. This is done until the majority and minority class instances are balanced out.

- **Advantages**

It can help improve run time and storage problems by reducing the number of training data samples when the training data set is huge.

- **Disadvantages**

1. It can discard potentially useful information which could be important for building rule classifiers.
2. The sample chosen by random under sampling may be a biased sample. And it will not be an accurate representative of the population. Thereby, resulting in inaccurate results with the actual test data set.

### 2.2.2 Random over sampling

Over-Sampling increases the number of instances in the minority class by randomly replicating them in order to present a higher representation of the minority class in the sample.

- **Advantages**

1. Unlike under sampling this method leads to no information loss.
2. Outperforms under sampling

- **Disadvantages**

It increases the likelihood of overfitting since it replicates the minority class events.

### 2.2.3 Synthetic Minority Over sampling Technique (SMOTE)

This technique is followed to avoid overfitting which occurs when exact replicas of minority instances are added to the main dataset. A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset. The new dataset is used as a sample to train the classification models.

- **Advantages**

1. Mitigates the problem of overfitting caused by random oversampling as synthetic examples are generated rather than replication of instances.
2. No loss of useful information.

- **Disadvantages**

1. While generating synthetic examples SMOTE does not take into consideration neighboring examples from other classes. This can result in increase in overlapping of classes and can introduce additional noise.
2. SMOTE is not very effective for high dimensional data.

# 3 Dataset information

## 3.1 Dataset link

https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008

## 3.2 Dataset information

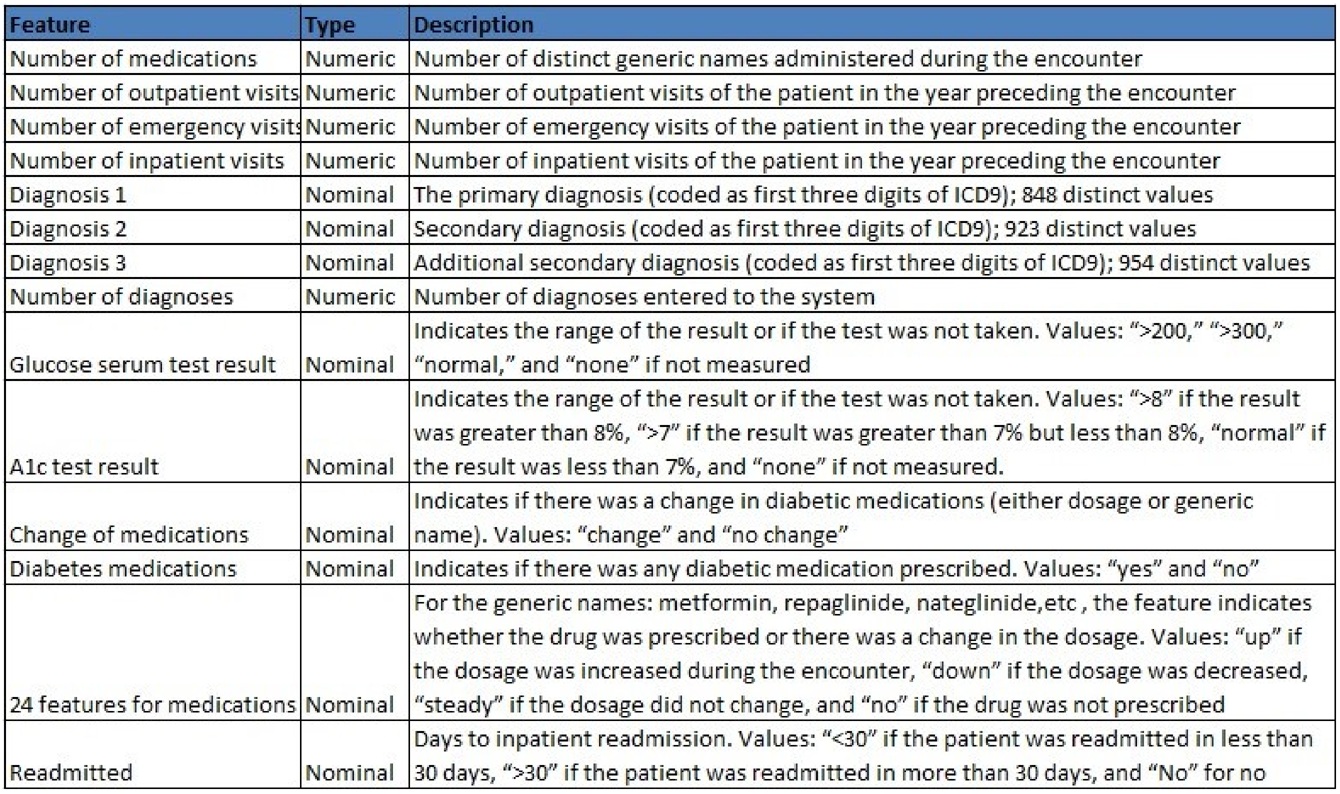
The dataset represents 10 years (1999-2008) of clinical care at 130 US hospitals and integrated delivery networks. It includes over 50 features representing patient and hospital outcomes. Information was extracted from the database for encounters that satisfied the following criteria.

(1) It is an inpatient encounter (a hospital admission).
(2) It is a diabetic encounter, that is, one during which any kind of diabetes was entered to the system as a diagnosis.
(3) The length of stay was at least 1 day and at most 14 days.
(4) Laboratory tests were performed during the encounter.
(5) Medications were administered during the encounter.

The data contains such attributes as patient number, race, gender, age, admission type, time in hospital, medical specialty of admitting physician, number of lab test performed, HbA1c test result, diagnosis, number of medications, diabetic medications, number of outpatient, inpatient, and emergency visits in the year before the hospitalization, etc.

## 3.3 Attribute information

| Feature | Type | Description |
|---|---|---|
| Encounter ID | Numeric | Unique identifier of an encounter |
| Patient number | Numeric | Unique identifier of a patient |
| Race | Nominal | Values: Caucasian, Asian, African American, Hispanic, and other |
| Gender | Nominal | Values: male, female, and unknown/invalid |
| Age | Nominal | Grouped in 10-year intervals: 0, 10), 10, 20), …, 90, 100) |
| Weight | Numeric | Weight in pounds. |
| Admission type | Nominal | Integer identifier corresponding to 9 distinct values, for example, emergency, urgent, elective, newborn, and not available |
| Discharge disposition | Nominal | Integer identifier corresponding to 29 distinct values, for example, discharged to home, expired, and not available |
| Admission source | Nominal | Integer identifier corresponding to 21 distinct values, for example, physician referral, emergency room, and transfer from a hospital |
| Time in hospital | Numeric | Integer number of days between admission and discharge |
| Payer code | Nominal | Integer identifier corresponding to 23 distinct values, for example, Blue Cross/Blue Shield, Medicare, and self-pay |
| Medical specialty | Nominal | Integer identifier of a specialty of the admitting physician, corresponding to 84 distinct values, for example, cardiology, internal medicine, family/general practice, and |
| Number of lab procedures | Numeric | Number of lab tests performed during the encounter |
| Number of procedures | Numeric | Number of procedures (other than lab tests) performed during the encounter |



| Feature | Type | Description |
|---|---|---|
| Number of medications | Numeric | Number of distinct generic names administered during the encounter |
| Number of outpatient visits | Numeric | Number of outpatient visits of the patient in the year preceding the encounter |
| Number of emergency visits | Numeric | Number of emergency visits of the patient in the year preceding the encounter |
| Number of inpatient visits | Numeric | Number of inpatient visits of the patient in the year preceding the encounter |
| Diagnosis 1 | Nominal | The primary diagnosis (coded as first three digits of ICD9); 848 distinct values |
| Diagnosis 2 | Nominal | Secondary diagnosis (coded as first three digits of ICD9); 923 distinct values |
| Diagnosis 3 | Nominal | Additional secondary diagnosis (coded as first three digits of ICD9); 954 distinct values |
| Number of diagnoses | Numeric | Number of diagnoses entered to the system |
| Glucose serum test result | Nominal | Indicates the range of the result or if the test was not taken. Values: ">200," ">300," "normal," and "none" if not measured |
| A1c test result | Nominal | Indicates the range of the result or if the test was not taken. Values: ">8" if the result was greater than 8%, ">7" if the result was greater than 7% but less than 8%, "normal" if the result was less than 7%, and "none" if not measured. |
| Change of medications | Nominal | Indicates if there was a change in diabetic medications (either dosage or generic name). Values: "change" and "no change" |
| Diabetes medications | Nominal | Indicates if there was any diabetic medication prescribed. Values: "yes" and "no" |
| 24 features for medications | Nominal | For the generic names: metformin, repaglinide, nateglinide,etc , the feature indicates whether the drug was prescribed or there was a change in the dosage. Values: "up" if the dosage was increased during the encounter, "down" if the dosage was decreased, "steady" if the dosage did not change, and "no" if the drug was not prescribed |
| Readmitted | Nominal | Days to inpatient readmission. Values: "<30" if the patient was readmitted in less than 30 days, ">30" if the patient was readmitted in more than 30 days, and "No" for no |

**Fig 3.1: Attribute description**

# 4 Exploratory Data Analysis

EDA is a general approach to exploring datasets by means of simple summary statistics and graphic visualizations in order to gain a deeper understanding of the data.

## 4.1 Handling minority classes in categorical features:

There are 3 categorical features in the data which are admission_source_id, discharge_disposition_id and admission_type_id.

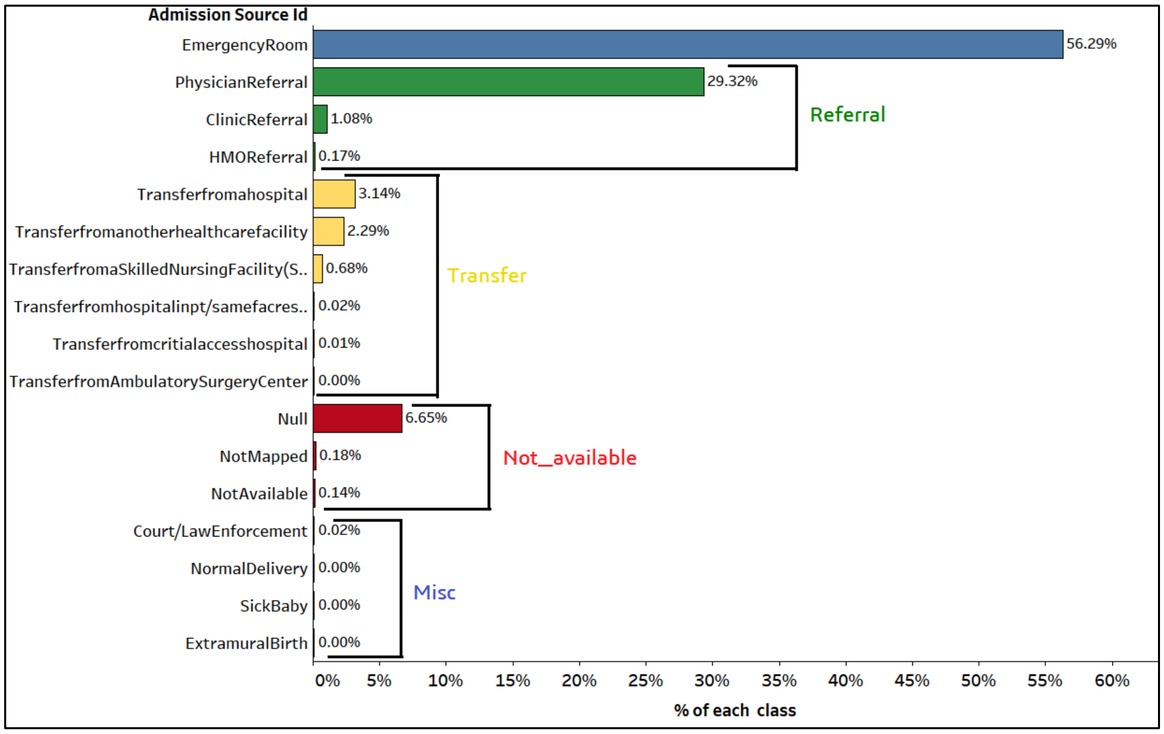


**Fig 4.1: Binning of admission-source-id**

There are 17 classes in the categorical feature admission_source_id. The above bar graph represents the percentage of number of records of that particular class in the dataset. For example, 56.29% of the records in the feature admission_source_id is related to the class named EmergencyRoom.

70% of the classes in admission_source_id (i.e. 12) contribute less than 1% of the records individually. When this feature is encoded before imparting into the model, 17 new features will be created out of which most of the features will be very sparse. That would lead to Curse of dimensionality. To avoid this scenario, number of classes are reduced by binning similar classes.

In the above plot, classes with the same color bars are binned together and given a new class name. For example, classes named PhysicalReferral, ClinicReferral and HMOReferral are binned and given new named class Referral. Similarly, other classes are binned to form new classes Transfer, Not_available and Misc (Miscellaneous).

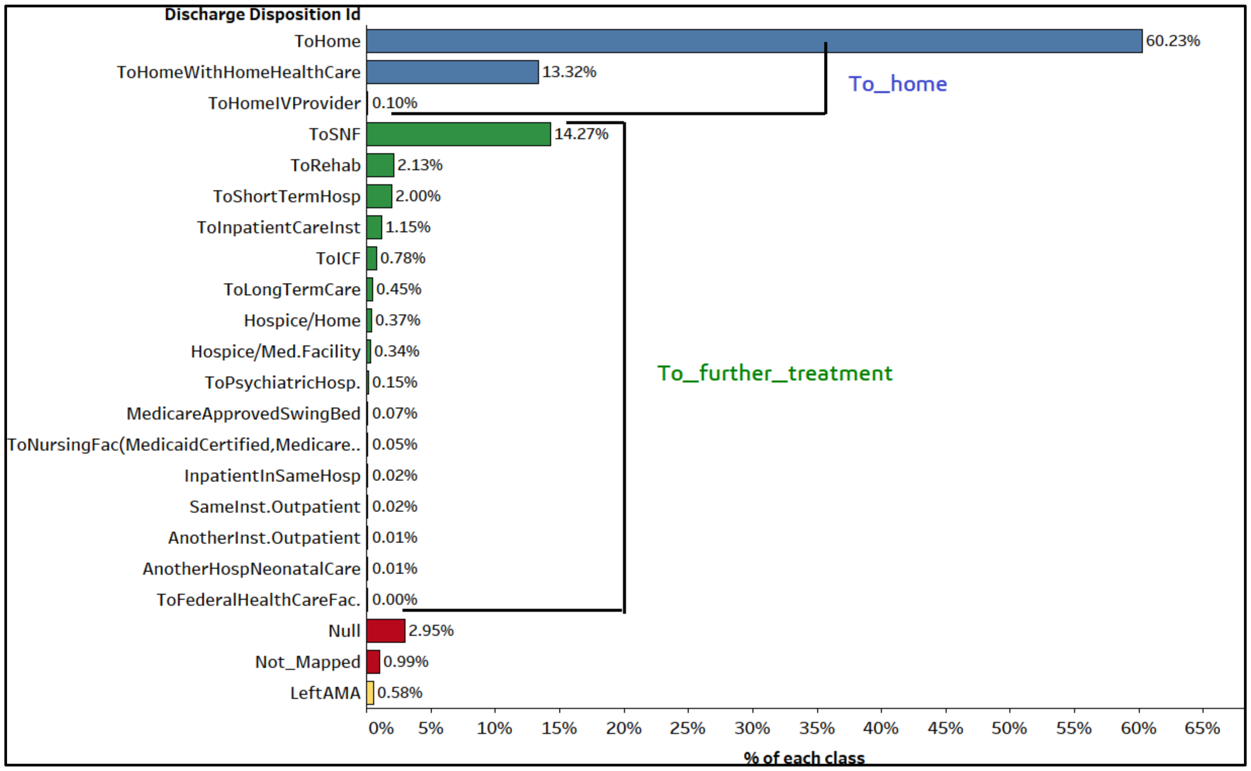


**Fig 4.2: Binning of discharge-disposition-id**

Similarly, in the categorical feature discharge_disposition_id, depending on the similarity among the classes, binnin has been done and new 4 classes are created.This can be visualised from the figure below. In the categorical feature admission_type_id,8 classes are binned to obtain 6 classes.

## 4.2 Efficiency of Insulin:

Generally, features named max_glu_serum and A1c_result would be helpful to find out if the drug has worked as expected. But, 95% and 83% of the values in the features max_glu_serum and A1c_result respectively are not available. Hence, another categorical feature named readmitted which has 3 classes >30,<30 and NO. >30 class indicates that the patient in that record has re-admitted in the hospital after 30 days. The class <30 indicates that the patient has re-admitted in the hospital within 30 days while the class NO indicates that the patient has not re-admitted.

Assumption has made that the drug has worked well for the patient in that record if he has not admitted after taking the drug. Hence, the number of records with the class NO are considered as that the drug has worked well while the records with other classes are considered as that the drug has not worked well.

Efficiency of working of drug is calculated by taking the ratio of the number of records with class NO to the number of total records who took drugs.

$$Efficiency = \frac{Number\ of\ \ records\ with\ class\ NO\ in\ readmitted}{Total\ number\ of\ records\ who\ took\ drugs}$$

$$Efficiency\ of\ Insulin = \frac{Number\ of\ records\ with\ class\ NO\ in\ readmitted}{Total\ number\ of\ records\ who\ only\ took\ Insulin}$$

To determine the efficiency of insulin exclusively, only the records with 1 in feature Insulin is considered. The total number of these records are taken in the denominator. In this filtered subset of data, number of records with class NO in readmitted feature is taken in the numerator. Thereby, the ratio of these number of records yields the efficiency of Insulin exclusively.

To find the efficiency of any single drug, data is filtered by considering the feature Total drugs and its value to be 1 that means the patient in that records has taken single drug. Total number of records of this filtered data is taken in denominator while number of records with class NO is taken in the numerator. Ratio of these numbers would determine the efficiency of any single drug. Similarly, Efficiency of other combination of drugs is also determined.

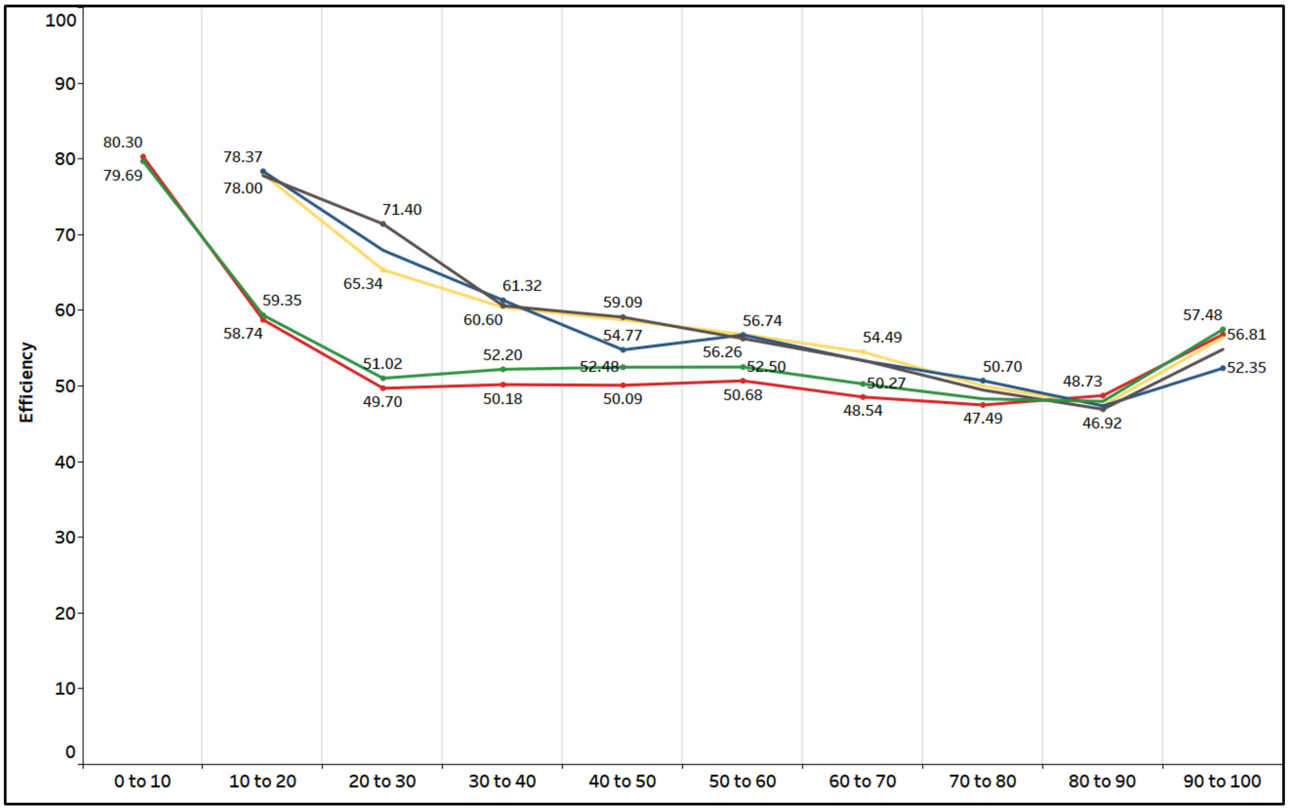
| | **Drug** | **Efficiency (%)** |
|---|---|---|
| 1 | Efficiency of Insulin exclusively | 49.6 |
| 2 | Efficiency of combination of drugs (Total Drugs > 1) | 52.8 |
| 3 | Efficiency of other drugs | 52.9 |
| 4 | Efficiency of any single drug | 60.1 |
| 5 | Efficiency of any single drug (excluding insulin) | 68.1 |

**Fig 4.3: Efficiency of different combinations of Drugs**

Effectiveness of drugs is also dependent on the age group that the patient belongs as well. Efficiency of the above drug and drug combinations is also calculated by taking the records with respect to age groups. The variation in efficiency with respect to age groups can be visualized in the plot below.

From the plot below, we can observe that drug has highest efficiency 80.3% for the patients belonging to the age group 0 to 10 years who were prescribed with Insulin exclusively. While the least efficiency of 49.62 is obtained for the patients belonging to the age group 80 to 90 years who were prescribed with a combination of drugs.

We can also observe the trend that the efficiency of all the drug combinations is decreasing as the age of the patient is increasing. This distribution seems to be relevant as drugs generally works well on younger people when compared to older people. Another observation is that the difference in the efficiency of drug combination is decreasing as the age is increasing. This can be visualized as the vertical distance (i.e. difference in efficiency) between drug combination is converging as the age is increasing and in the age group 80 to 90 years, all the drug combinations' efficiency is almost equal.



Drug combinations
- 🟩 Any single drug
- ⬛ Any single drug excluding insulin
- 🟦 Combination of Drugs (Total drugs >1)
- 🟨 Other drugs
- 🟥 Solo insulin

**Fig 4.4: Efficiency of Insulins & other drugs**

Insights are comparisons between Diabetic and Non-Diabetic encounters and can help a hospital plan its resources in a better way based on the patient's age and his/her diabetic status.

### 4.2.1 Average number of Diagnosis

- Both cases show an upward trend up to 50 years of age after which average number of diagnosis remain constant in both cases (close to 8 diagnoses on average).

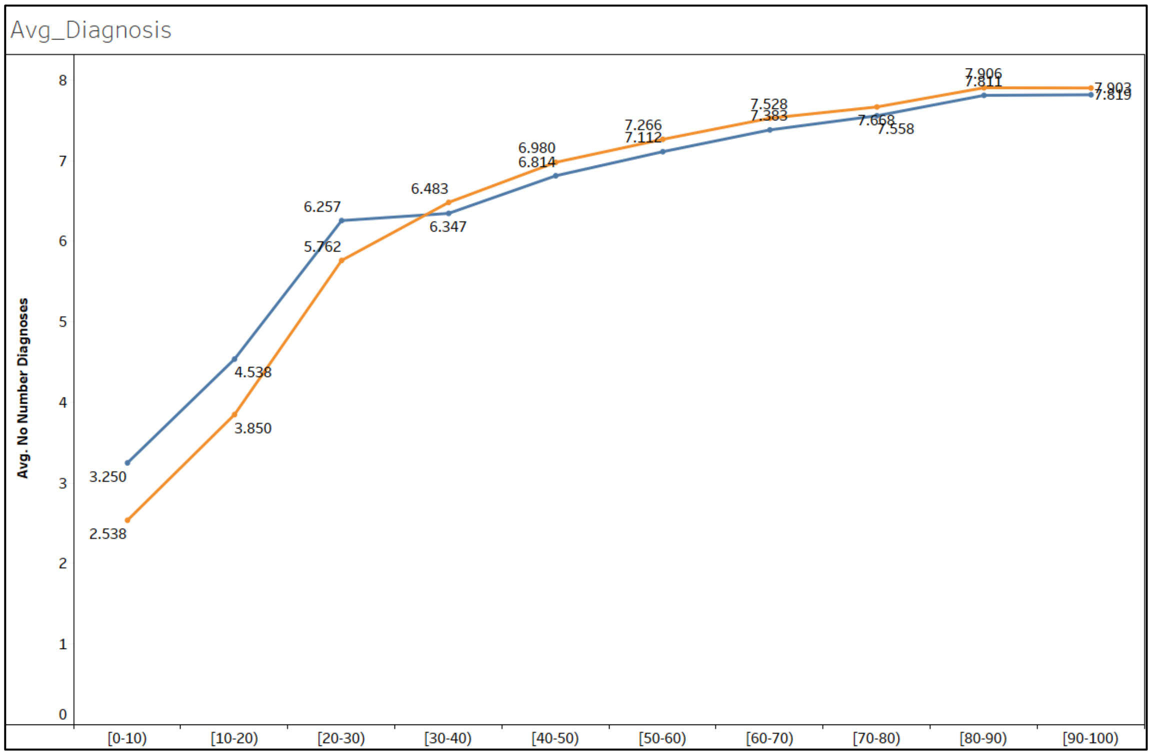


**Fig 4.5: Average number of Diagnosis**

### 4.2.2 Average number of Medications

- Highest average number of medications occur for age groups of 50-70 years in both cases.

- Highest average number of medications for diabetic patients (18 medicines on avg) is 30% more than highest average number of medications for non-diabetic patients (15 medicines on avg).

- After 70 years, the average number of medications shows a downward trend in both cases.

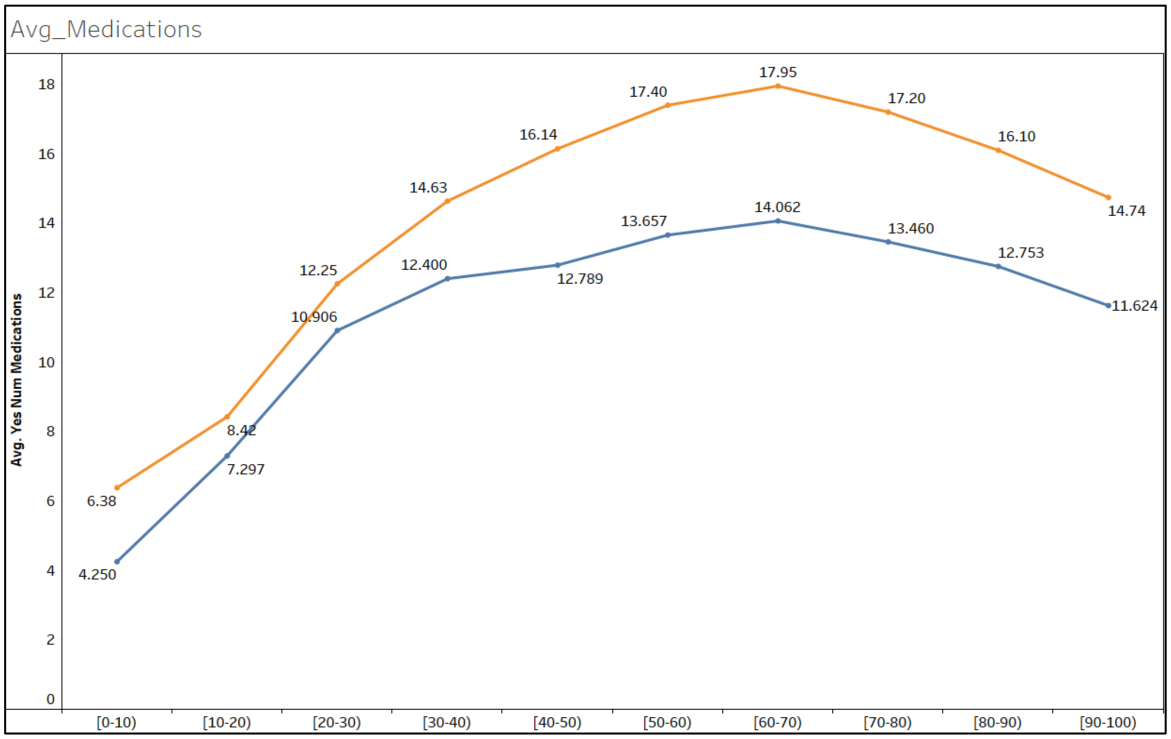


**Fig 4.6 Average number of Medications**

### 4.2.3 Average number of Emergencies

- Average number of emergencies are maximum for age group of <u>20-30 years</u> in both cases.
- From <u>30 years</u>, average number of emergencies shows a downward trend and then become constant.

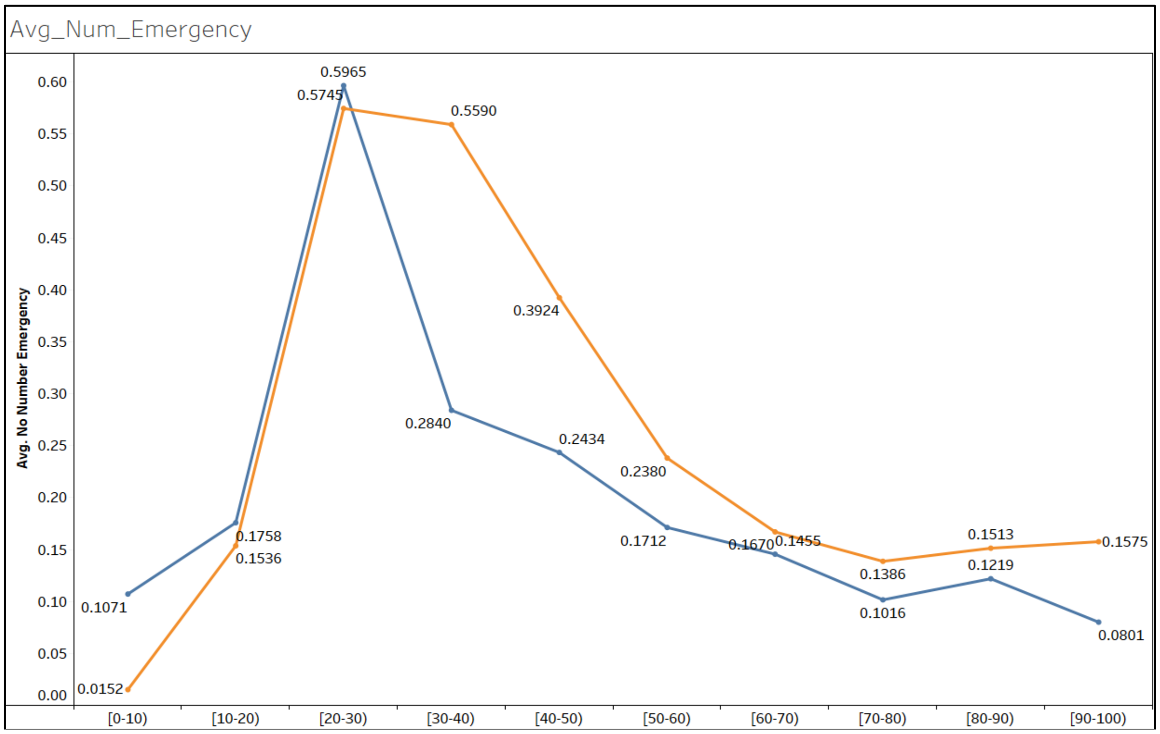


**Fig 4.7: Average number of Emergencies**

### 4.2.4 Average number of Inpatients

- Average number of inpatients are highest for age group of 20-30 years in both cases (close to 2 inpatient visits).

- Highest number of average inpatients in diabetic patients is 65% more than highest number of average inpatients in non-diabetic patients.

- From 50 years, the average number of inpatients become constant in both cases (0.6-0.8).

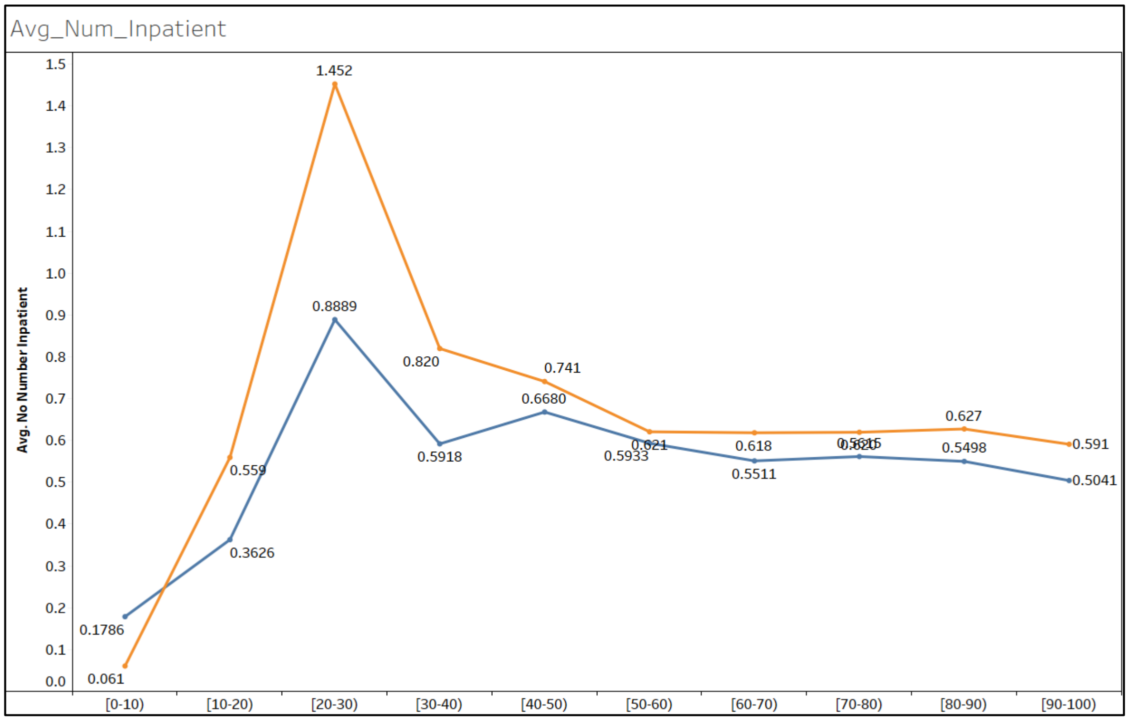


**Fig 4.8: Average number of Inpatients**

### 4.2.5 Average number of Outpatients

- For diabetic patients, highest average number of outpatients appear at age group of 80-90 years.

- For non-diabetic patients, highest average number of outpatients appear at age group of 20-30 years.

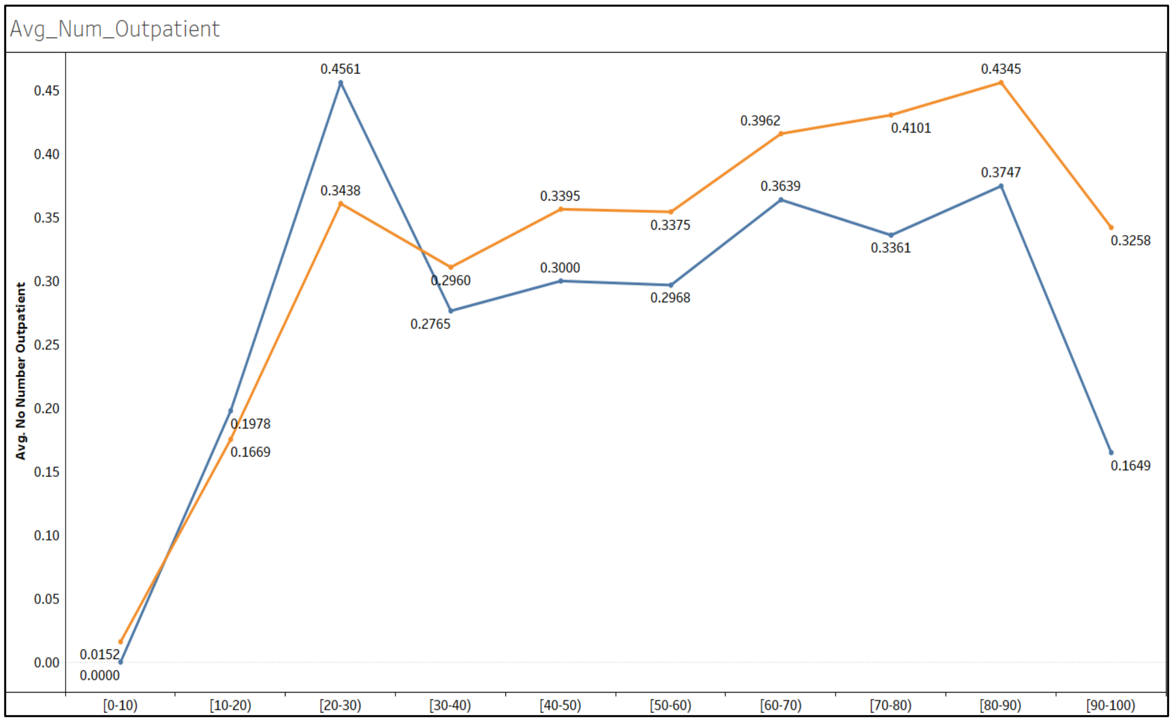


**Fig 4.9: Average number of Outpatients**

### 4.2.6 Average number of Procedures

- Highest average number of procedures appear at age group of <u>60-70 years</u> in both cases.
- <u>Beyond 70 years</u> there is sharp drop in average number of procedures.

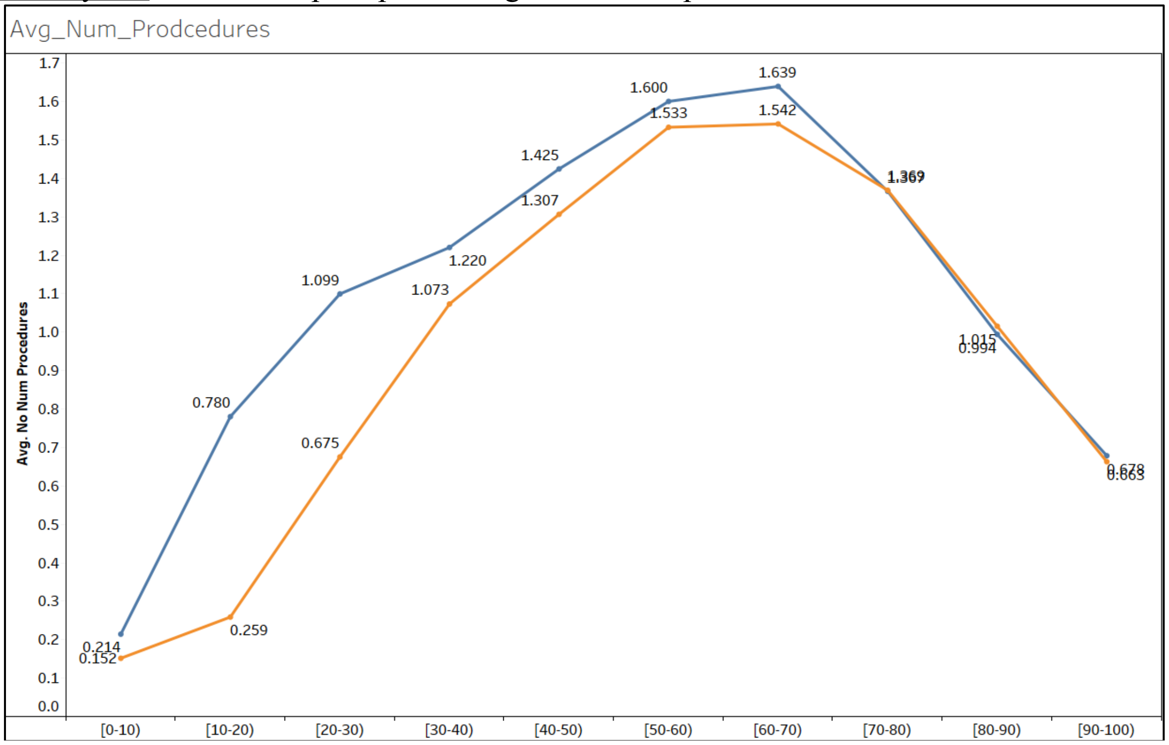


**Fig 4.10: Average number of Procedures**

## 4.3 Patient demographic dashboard

1. For ages 0-40 years, 70% people take SoloInsulin i.e. they have Type-I Diabetes. For ages above 40 years, 65% people take other drugs(other drugs Or SoloInsulin+other drugs) i.e. they have Type-II Diabetes.

2. SoloInsulin as well as Other drugs were used maximum by patients in the age group : 70-80 years.

3. For all age groups, the number of 'not readmitted' encounters are greater than 'readmitted' ones except in age group : 70-90 years.

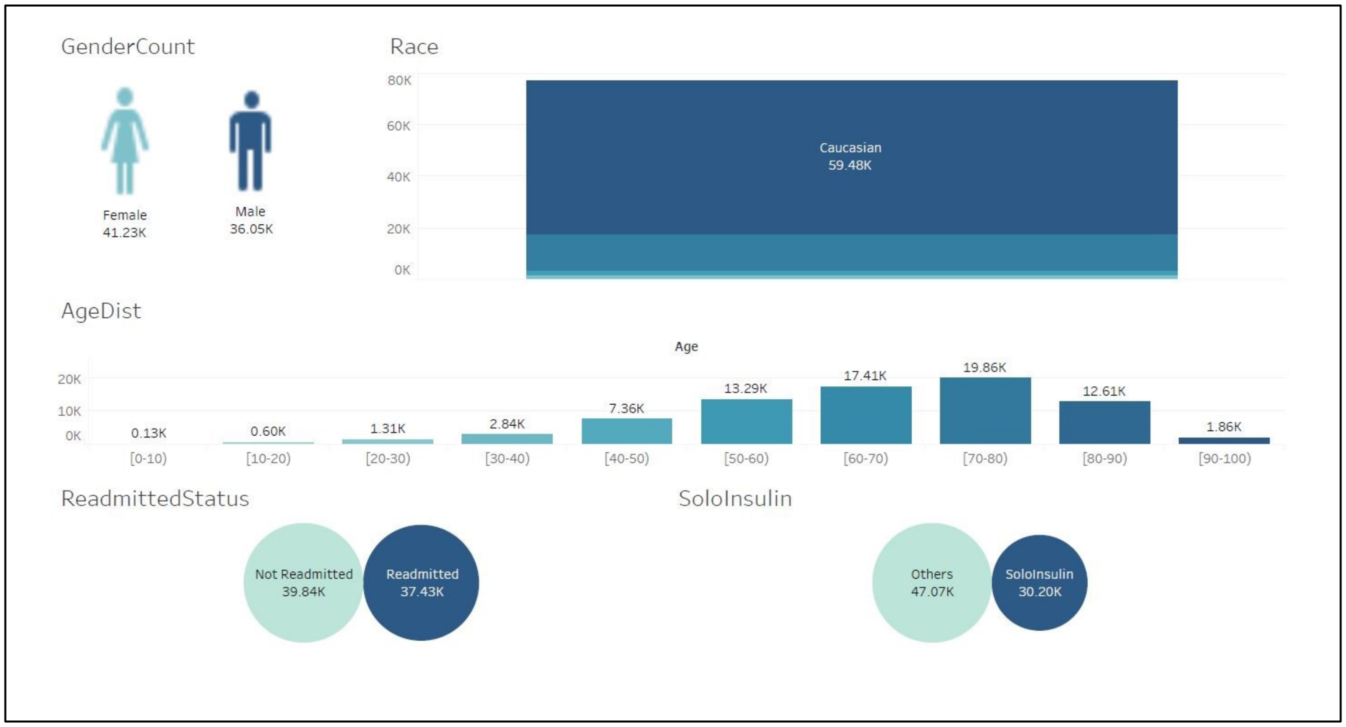


**Fig 4.11: Insights from Patient Demographic Dashboard**

## 4.4 Admission and discharge dashboard

1. In Admission Type - 'Emergency','Elective' and 'Urgent', close to 40% of encounters are taking SoloInsulin i.e. Type-I Diabetic.

2. Only for AdmissionSource : 'EmergencyRoom' number of 'Readmitted' encounters>number of 'Not Readmitted' encounters. For all other Admission Sources, it is vice versa meaning they are cured of diabetes.

3. For DischargeDispositionID as 'ToHome' and 'ToFurtherTreatment', approx. 50% of encounters were 'Readmitted' .

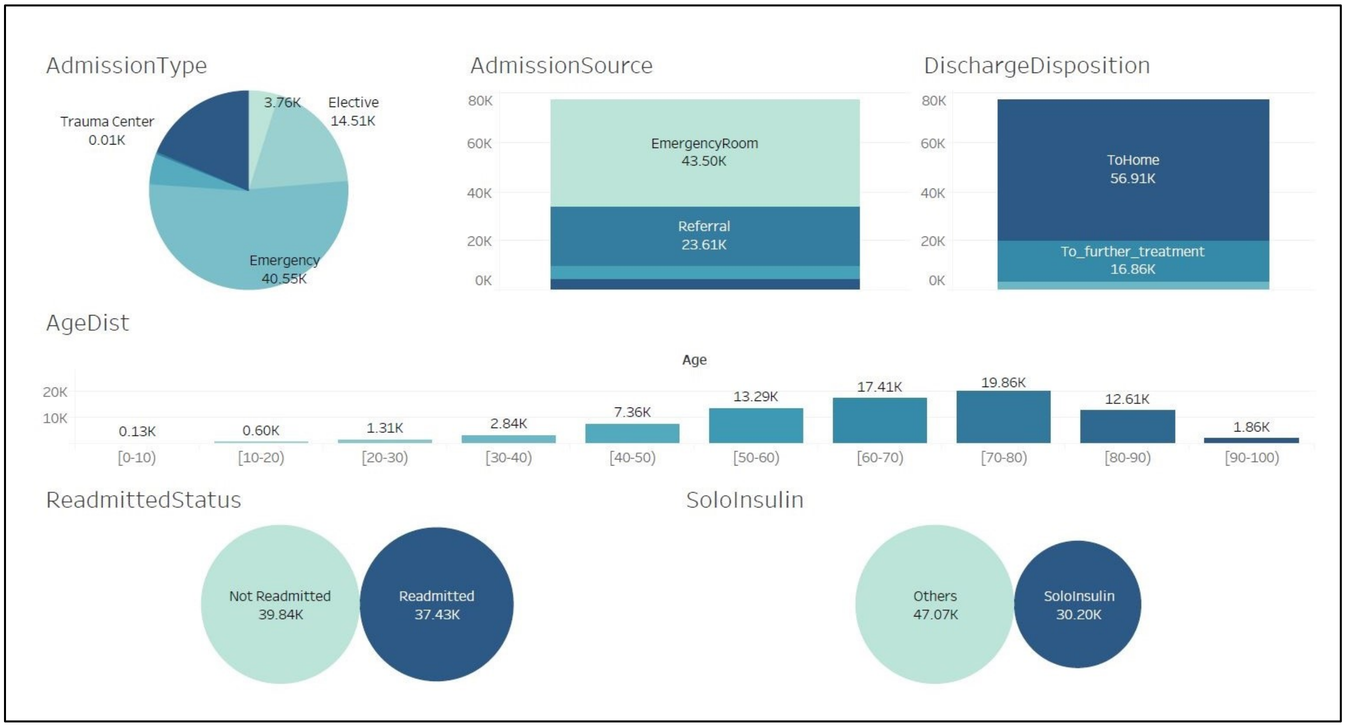


**Fig 4.12: Insights from Admission & Discharge Dashboard**

## 4.5 Handling missing values

In any real-world dataset, there are usually few null values. It doesn’t really matter whether it is regression, classification or any other kind of problem no model can handle these NULL or NaN values on its own so we need to intervene. First of all, we need to check whether we have null values in our dataset or not. We can do that using the isnull() method. There are various ways for us to handle this problem. The easiest way to solve this problem is by dropping the rows or columns that contain null values.

dropna() takes various parameters like:

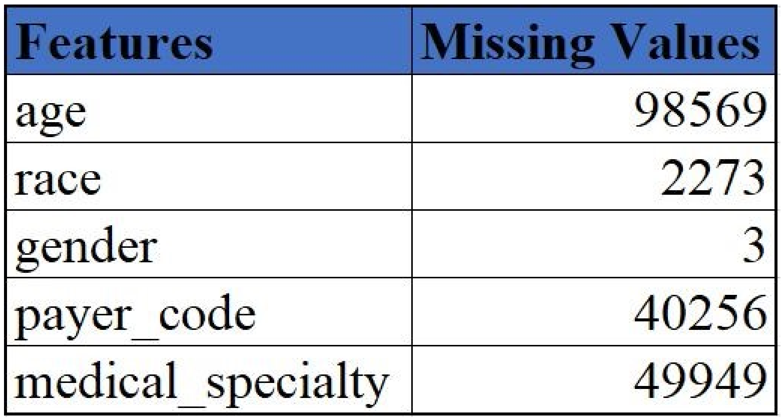
axis — We can specify axis=0 if we want to remove the rows and axis=1 if we want to remove the columns.

how — If we specify how = ‘all’ then the rows and columns will only be dropped if all the values are NaN. By default, ‘how’ is set to ‘any’.

inplace — By default no changes will be made to your data frame. So, if you want these changes to reflect onto your data frame then you need to use inplace = True.

However, it is not the best option to remove the rows and columns from our dataset as it can lead to loss of valuable information. So, if you have 300K data points then removing 2–3 rows won’t affect your dataset, whereas if you only have 100 data points and out of which 20 have NaN values for a particular field then you can’t simply drop those rows. In real life datasets it can happen quite often that you have large number of NaN values for a particular field.

Suppose we are collecting the data from a survey, then it is possible that there could be an optional field which let’s say 20% people left blank. So when we get the dataset then we need to understand that the remaining 80% data is still useful so rather than dropping these values we need to somehow substitute the missing 20% values. We can do this with the help of Imputation.Imputation is simply the process of substituting the missing values of our dataset. We can do this by defining our own customized function or we can simply perform imputation by using the Imputer class provided by sklearn.In Imputer() you can pass on the axis and strategy. strategy could be mean, median etc.



| Features | Missing Values |
|---|---|
| age | 98569 |
| race | 2273 |
| gender | 3 |
| payer_code | 40256 |
| medical_specialty | 49949 |

**Fig 4.13: No. of missing values**

- Categorical features having missing values (< 10%) were replaced with mode. (race,gender)
- When there was no clear mode, missing values were imputed with class name 'missing' wording for categorical variables.
- Features with high number of missing values (> 40%) were dropped. (weight, payer_code, medical_specialty)
- None of the numerical features had missing values.

## 4.6 Outlier treatment

Outlier is a commonly used terminology by analysts and data scientists as it needs close attention else it can result in wildly wrong estimations. Simply speaking, Outlier is an observation that appears far away and diverges from an overall pattern in a sample. Let's take an example, we do customer profiling and find out that the average annual income of customers is $0.8 million. But there are two customers having annual income of $4 and $4.2 million. These two customers annual income is much higher than rest of the population. These two observations will be seen as Outliers. Outliers can drastically change the results of the data analysis and statistical modelling. There are numerous unfavorable impacts of outliers in the data set:

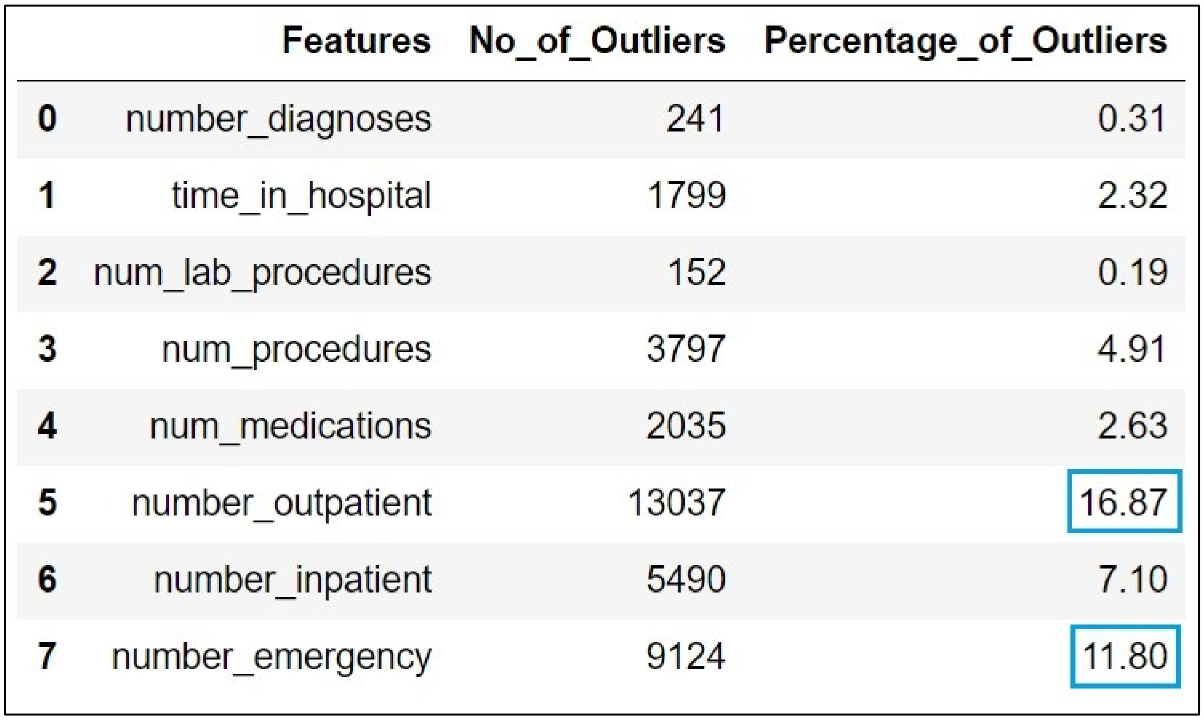
- It increases the error variance and reduces the power of statistical tests
- If the outliers are non-randomly distributed, they can decrease normality
- They can bias or influence estimates that may be of substantive interest
- They can also impact the basic assumption of Regression, ANOVA and other statistical model assumptions.

To understand the impact deeply, let's take an example to check what happens to a data set with and without outliers in the data set.

### 4.6.1 Detecting outliers

Most commonly used method to detect outliers is visualization. We use various visualization methods, like **Box-plot**, **Histogram**, **Scatter Plot** (above, we have used box plot and scatter plot for visualization). Some analysts also various thumb rules to detect outliers. Some of them are:

- Any value, which is beyond the range of -1.5 x IQR to 1.5 x IQR
- Use capping methods. Any value which out of range of 5th and 95th percentile can be considered as outlier
- Data points, three or more standard deviation away from mean are considered outlier
- Outlier detection is merely a special case of the examination of data for influential data points and it also depends on the business understanding
- Bivariate and multivariate outliers are typically measured using either an index of influence or leverage, or distance. Popular indices such as Mahalanobis' distance and Cook's *D* are frequently used to detect outliers.



| | Features | No_of_Outliers | Percentage_of_Outliers |
|---|---|---|---|
| 0 | number_diagnoses | 241 | 0.31 |
| 1 | time_in_hospital | 1799 | 2.32 |
| 2 | num_lab_procedures | 152 | 0.19 |
| 3 | num_procedures | 3797 | 4.91 |
| 4 | num_medications | 2035 | 2.63 |
| 5 | number_outpatient | 13037 | $\boxed{16.87}$ |
| 6 | number_inpatient | 5490 | 7.10 |
| 7 | number_emergency | 9124 | $\boxed{11.80}$ |

**Fig 4.14: Outliers & its percentage**

### 4.6.2 Removing Outliers

**Transforming and binning values:** Transforming variables can also eliminate outliers. Natural log of a value reduces the variation caused by extreme values. Binning is also a form of variable transformation. Decision Tree algorithm allows to deal with outliers well due to binning of variable. We can also use the process of assigning weights to different observations.

**Winsorization:** It is a way to minimize the influence of outliers in your data by either:

- Assigning the outlier, a lower weight,
- Changing the value so that it is close to other values in the set.

Steps:
1. **Analyze your data** to make sure the outlier isn't a result of measurement error or some other fixable error.
2. **Decide how much Winsorization you want.** This is specified as a total percentage of *untouched* data. For example, if you want to Winsorize the top 5% and bottom 5% of data points, this is equal to 100% – 5% – 5% = 90% Winsorization. A 80% Winsorization means that 10% is modified from each tail area (see Tips on Cut-Off Point Selection below).
3. Replace the extreme values by the maximum and/or minimum values at the threshold.

***We can see that highest number of outliers are present in number_outpatient as 16.87% and number_emergency as 11.80%***

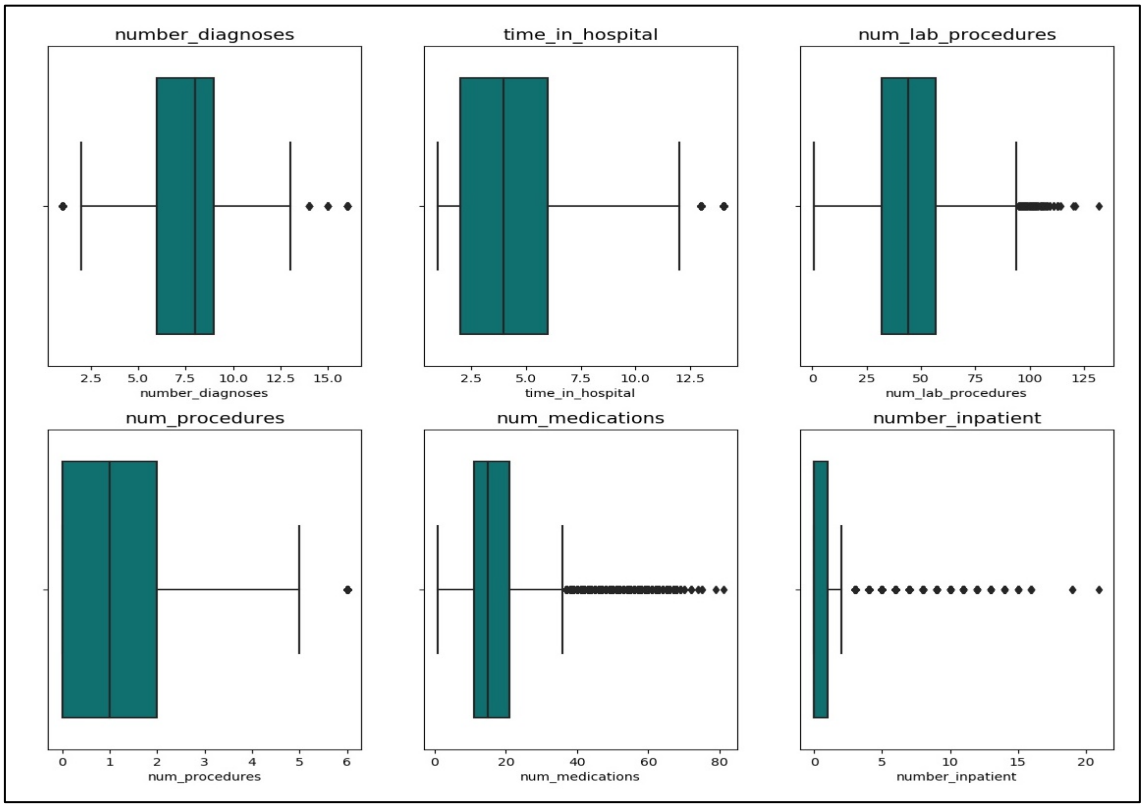


Fig 4.15: Before Outlier Treatment

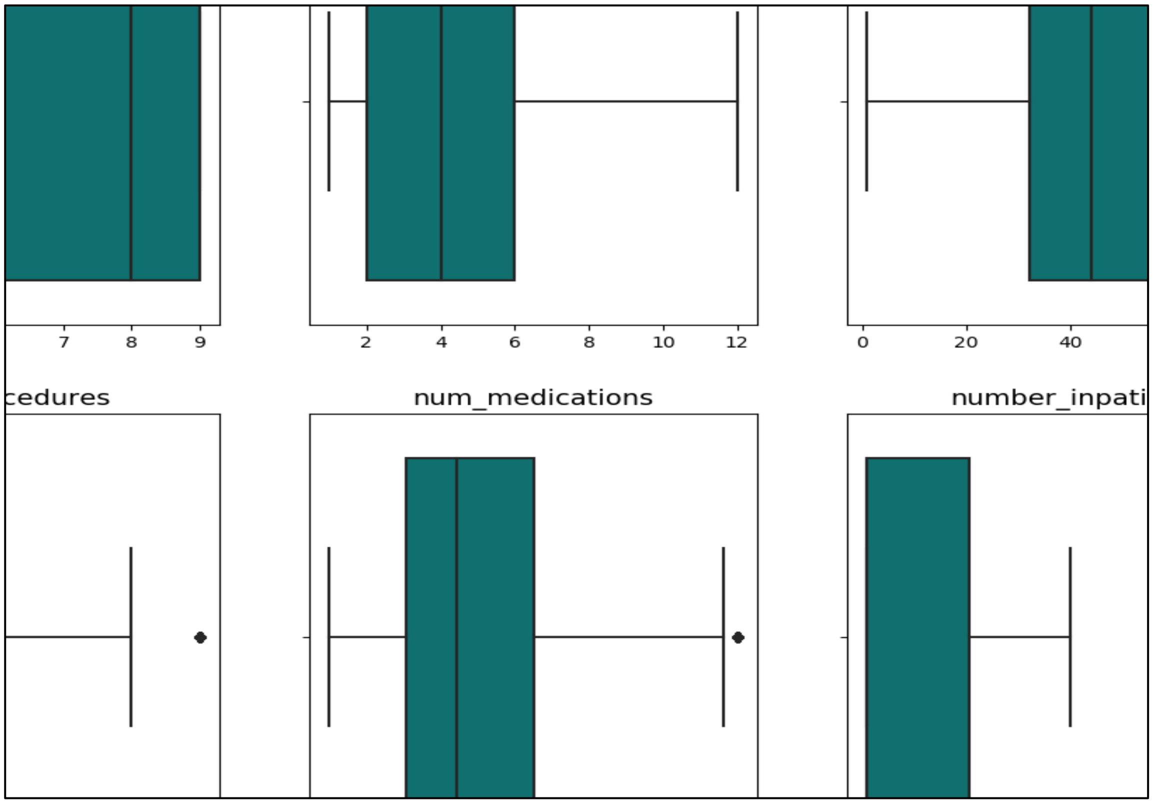


Fig 4.16: After Outlier Treatment (using Winsorizer)

## 4.7 Splitting the dataset into train set and test set

Now we need to split our dataset into two sets — a Training set and a Test set. We will train our machine learning models on our training set, i.e. our machine learning models will try to understand any correlations in our training set and then we will test the models on our test set to check how accurately it can predict. A general rule of the thumb is to allocate 80% of the dataset to training set and the remaining 20% to test set. For this task, we will import test_train_split from model_selection library of scikit.

### 4.7.1 Data imbalance

The dataset that will be used to train the model has some challenges. In particular it has been subjected to the imbalanced dataset problem, whereby not all classes have similar number of instances. To handle the imbalanced classes there were a few possible solutions and these are discussed in more detail above. One way to deal with imbalanced dataset is by down sampling which involves reducing the number of instances by tossing away some instances of classes with high number of instances to match the classes with lower number of instances. This has the advantage that it is simple to implement and has proven to be more effective than up sampling. However, this method has the major disadvantage that useful and valuable instances are thrown away upon training.

Another common solution to deal with imbalanced dataset is by up sampling which involves generating synthetic data or making multiple copies of the instances for classes with low number of instances (minority class) to match the majority class. This has the advantage that it does not throw away useful instances. However, this method has proved in the past to cause overfitting and can be computationally intensive if the number of instances and classes are very large. A common solution that changes how the model learns instead of the dataset itself is known as Cost Sensitive Learning. It works by changing the loss/error function to give a different error value to False Positive and False Negatives depending on which is more significant, and hence model will avoid the error with a higher error value. The advantage of this is that it changes the model instead of the data. However, it can be ineffective on severely imbalanced datasets and does not guarantee convergence to optimal model.

Since in our project the number of observations is less, up sampling methods are preferred over down sampling in order to avoid the loss of information. SMOTE is one of the up-sampling techniques which is followed here to avoid over fitting which occurs when exact replicas of minority instances are added to the main dataset. A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset. The new dataset is used as a sample to train the classification models.

### 4.7.2 Statistical test for categorical and numerical features

Our dataset consists of categorical as well as numerical variables. We have our target variable as categorical variable so we need to perform t test and chi-square test.

#### CHI-SQUARE TEST

A chi-square test for independence compares two Categorical variables in a contingency table to see if they are related. In a more general sense, it tests to see whether distributions of categorical variables differ from each another. In our case, we need to determine whether there is indeed a relationship between a predictor variable and any of the target variables to a significant degree. We only need to consider these features for our further analysis.

**The null hypothesis of the Chi-Square** test is that no relationship exists on the categorical variables being tested. I.e. they are independent. The p-value will tell us if our test results are significant or not. In order to perform a chi square test and get the p-value, you need two pieces of information:

- Degrees of freedom. That's just the number of categories minus 1.

- The alpha level(α). The usual alpha or significance level is 0.05 (5%), but you could also have other levels like 0.01 or 0.10.

#### T-TEST

A t-test is used to determine if there is a significant difference between the means of two groups, which may be related in certain features. would follow a normal distribution and may have unknown variances. A t-test is used as a hypothesis testing tool, which allows testing of an assumption applicable to a population.

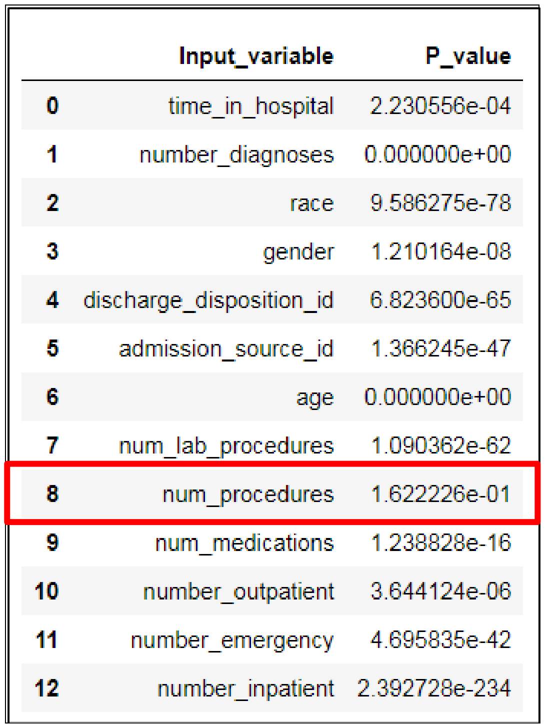
A t-test looks at the t-statistic, the t-distribution values, and the degrees of freedom to determine the probability of difference between two sets of data. To conduct a test with three or more variables, one must use an analysis of variance.

**The null hypothesis of the T-Test** is that average mean of categorical and numerical variables being tested. I.e. they are independent. The p-value will tell us if our test results are significant or not. In order to perform a T-test and get the p-value, you need two pieces of information:

- Grouping mean value by categorical variable.

- The alpha level(α). The usual alpha or significance level is 0.05 (5%), but you could also have other levels like 0.01 or 0.10.

***We reject the null hypothesis when the P-value is less than the set significance level.***

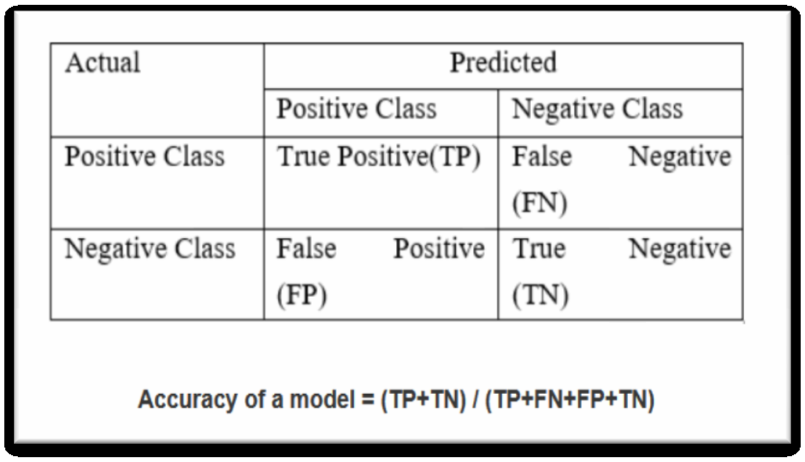


| | Input_variable | P_value |
|---|---|---|
| 0 | time_in_hospital | 2.230556e-04 |
| 1 | number_diagnoses | 0.000000e+00 |
| 2 | race | 9.586275e-78 |
| 3 | gender | 1.210164e-08 |
| 4 | discharge_disposition_id | 6.823600e-65 |
| 5 | admission_source_id | 1.366245e-47 |
| 6 | age | 0.000000e+00 |
| 7 | num_lab_procedures | 1.090362e-62 |
| 8 | num_procedures | 1.622226e-01 |
| 9 | num_medications | 1.238828e-16 |
| 10 | number_outpatient | 3.644124e-06 |
| 11 | number_emergency | 4.695835e-42 |
| 12 | number_inpatient | 2.392728e-234 |

**Fig 4.17: Results of statistics test**

## 5 Architecture

We have to take a different approach here from a normal machine learning flow because of the nature of our data. The conventional model evaluation methods do not accurately measure model performance when faced with imbalanced datasets. Standard classifier algorithms like Decision Tree and Logistic Regression have a bias towards classes which have number of instances. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of misclassification of the minority class as compared to the majority class. Evaluation of a classification algorithm performance is measured by the Confusion Matrix which contains information about the actual and the predicted class.



| Actual | Predicted | |
|---|---|---|
| | Positive Class | Negative Class |
| Positive Class | True Positive(TP) | False Negative (FN) |
| Negative Class | False Positive (FP) | True Negative (TN) |

Accuracy of a model = (TP+TN) / (TP+FN+FP+TN)

**Fig 5.1: Confusion Matrix**

However, while working in an imbalanced domain accuracy is not an appropriate measure to evaluate model performance**.** For e.g.: A classifier which achieves an accuracy of 98 % with an event rate of 2 % is not accurate, if it classifies all instances as the majority class. And eliminates the 2 % minority class observations as noise. To fully evaluate the effectiveness of our model, we must examine **precision** and **recall** as well. Unfortunately, precision and recall are often in tension. That is, improving precision typically reduces recall and vice versa.

**Precision:** What proportion of positive identifications was actually correct?

$$\text{Precision} = \frac{TP}{TP + FP}$$

**Recall:** What proportion of actual positives was identified correctly?

$$\text{Recall} = \frac{TP}{TP + FN}$$

## 5.1 Feature Engineering

Our data is a high dimensional data with majority of the columns as categorical. This will lead to creation of many more columns on performing one hot encoding. To take care of this, we had chosen to implement feature engineering.
Each of the 23 drugs have four sub-classes: 'Up', 'Down', 'Steady' and 'NO'. These features are nominal data and would have to be handled using one hot encoding which would lead to the creation of 23 x 4 = 92 extra columns. Instead, we figured out a way to extract their data into **two new columns** – '**Total_drugs**' and '**Solo_Insulin**'.

**Total_drugs** would be a numerical column which would count how many drugs amongst our 23 drugs are being administered. The values range between 1-6 drugs.
**Solo_Insulin** would be a binary categorical column which would indicate 1 if the encounter is taking only insulin and no other drug. It would indicate 0 if the encounter takes any other drug except insulin.

## 5.2 Building Base Model

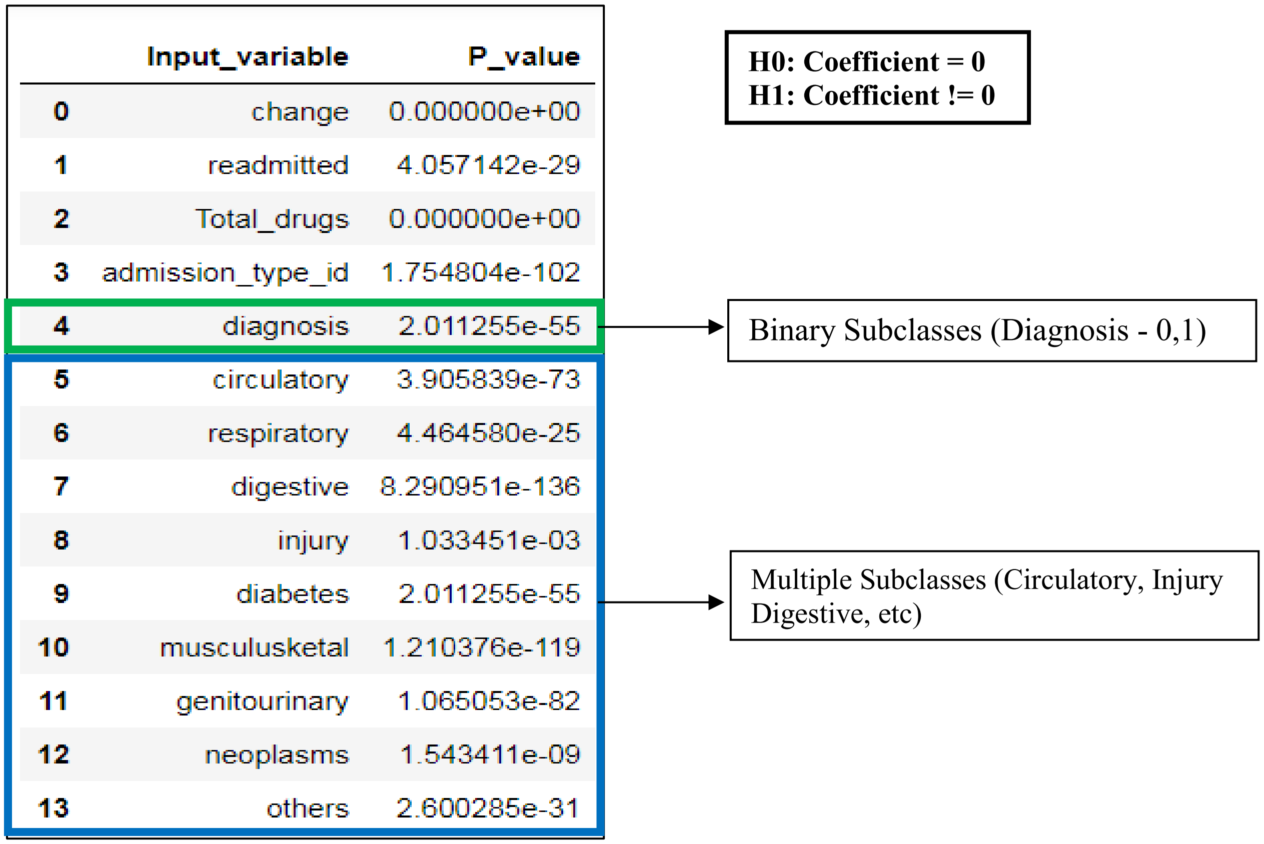
The first model that we built as our baseline model was a logistic regression. Our target variable was Solo_Insulin and the test size taken was 20%. The AUC we obtained was 50%.

## 5.3 Binary/Multi Sub-Class Approach

Predictors – diag_1, diag_2 and diag_3 are ICD9 codes which talk about the ailment that the encounter was diagnosed with in the primary diagnosis. For example, values between 250 and 251 indicate that the encounter has diabetes and values between 800-999 mean physical injury etc. The values are alpha-numeric meaning that they cannot be handled using any kind of encoding methods.

We had thought of two different approaches to tackle this problem.
Binary Sub-classes: A new feature – 'diagnosis' was created where encounter having diag_1, diag_2 or diag_3 values between 250-251 will have 'diagnosis' as 1 else it will indicate 0.
Multiple Sub-classes: A new feature – 'group_name' was created which had 9 subclasses – 'circulaory', 'respiratory', 'digestive', 'injury', 'diabetes', 'musculoskeletal', 'genitourinary', 'neoplasms' and 'others'. They would be categorized based on values of diag_1, diag_2 and diag_3. On one hot encoding we would get 9 new columns.

Due to a lack of domain knowledge, we went forward with both approaches and decided to choose the approach which yielded best results.
Below are the p-values that we obtained on conducting chi-square tests on our new features with respect to our target variable – Solo_Insulin.



|  | Input_variable | P_value |
|---|---|---|
| 0 | change | 0.000000e+00 |
| 1 | readmitted | 4.057142e-29 |
| 2 | Total_drugs | 0.000000e+00 |
| 3 | admission_type_id | 1.754804e-102 |
| 4 | diagnosis | 2.011255e-55 |
| 5 | circulatory | 3.905839e-73 |
| 6 | respiratory | 4.464580e-25 |
| 7 | digestive | 8.290951e-136 |
| 8 | injury | 1.033451e-03 |
| 9 | diabetes | 2.011255e-55 |
| 10 | musculusketal | 1.210376e-119 |
| 11 | genitourinary | 1.065053e-82 |
| 12 | neoplasms | 1.543411e-09 |
| 13 | others | 2.600285e-31 |

**H0: Coefficient = 0**
**H1: Coefficient != 0**

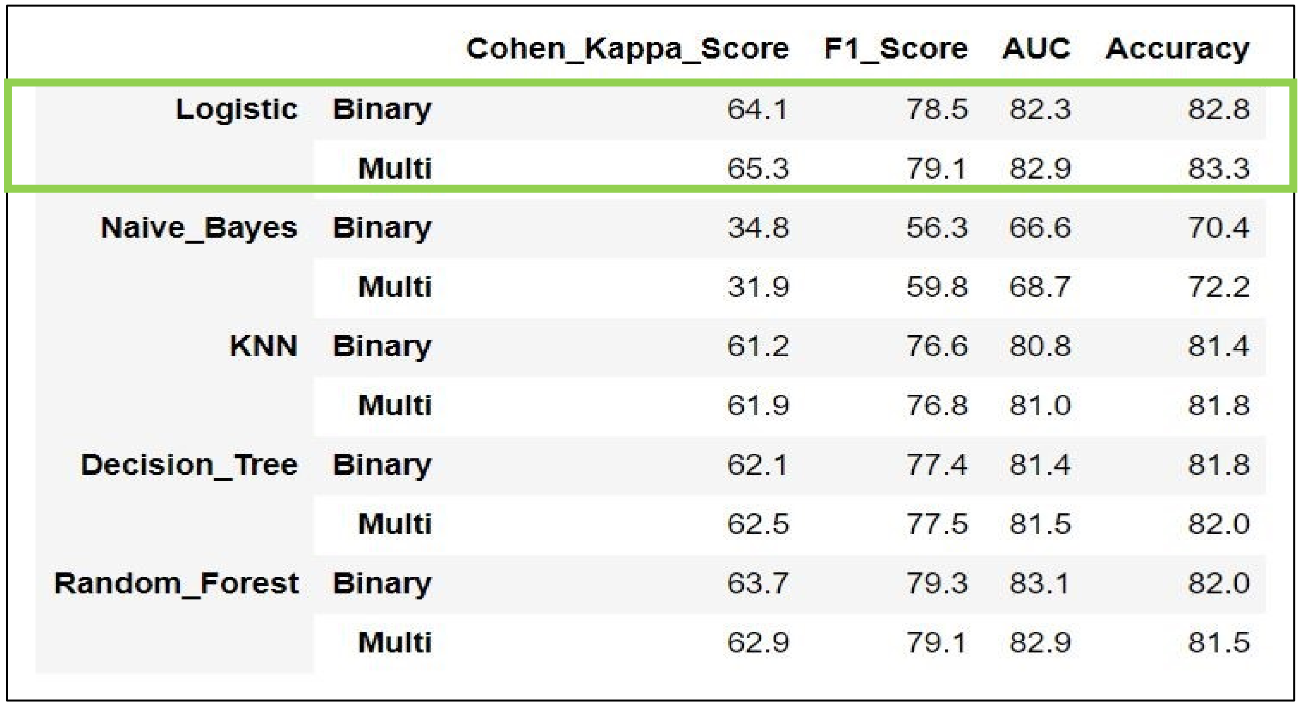
Binary Subclasses (Diagnosis - 0,1)

Multiple Subclasses (Circulatory, Injury Digestive, etc)

**Fig 5.2: Statistics results of binary & multi sub-classes**

As you can see above, all the features have p-value less than **0.05**, meaning that they can be considered in model building.

Next, we built multiple classification models and compared metrics to determine which approach would best suit our data.
The algorithms that we implemented were – Logistic Regression, Naives Bayes, K-Nearest Neighbor, Decision Tree and Random Forest. The metrics used to evaluate the algorithms were Area Under the Curve (AUC), F1-Score, Cohen’s Kappa Score and Accuracy. The below figure shows the results for binary and multiple sub-class approach for K-Fold Cross Validation with **n_folds** = **5**.



| | | Cohen_Kappa_Score | F1_Score | AUC | Accuracy |
|---|---|---|---|---|---|
| Logistic | Binary | 64.1 | 78.5 | 82.3 | 82.8 |
| | Multi | 65.3 | 79.1 | 82.9 | 83.3 |
| Naive_Bayes | Binary | 34.8 | 56.3 | 66.6 | 70.4 |
| | Multi | 31.9 | 59.8 | 68.7 | 72.2 |
| KNN | Binary | 61.2 | 76.6 | 80.8 | 81.4 |
| | Multi | 61.9 | 76.8 | 81.0 | 81.8 |
| Decision_Tree | Binary | 62.1 | 77.4 | 81.4 | 81.8 |
| | Multi | 62.5 | 77.5 | 81.5 | 82.0 |
| Random_Forest | Binary | 63.7 | 79.3 | 83.1 | 82.0 |
| | Multi | 62.9 | 79.1 | 82.9 | 81.5 |

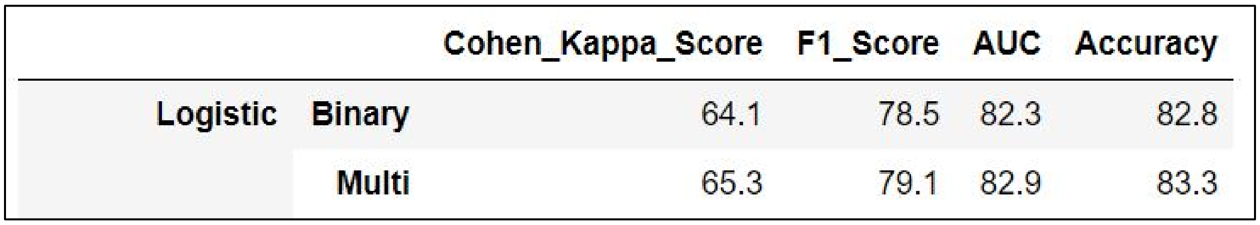
**Fig 5.3: Model's Comparison**

We got the best result for Logistic Regression for both binary and multiple sub-classes approach. Amongst the two approaches we got the best result for Multiple Sub-classes approach. Therefore, we decided to go ahead with this approach.

## 5.4 Target Sub-Class Imbalance

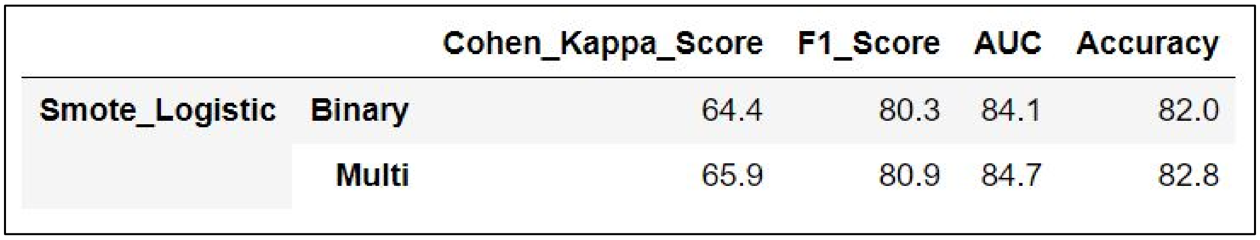
Our binary target variable – Solo_Insulin has a class imbalance. Our target variable is positive minority feature meaning that the number of 1's in the feature are less when compared to 0's in the feature. A class imbalance leads to our model not having a proper training on the minority class leading to faulty predictions. The 0's and 1's in the target variable are in 3:2 ratio.

There are different ways to tackle class imbalance but we chose to use SMOTE. We wanted to check how much improvement could be seen in the evaluation metrics after the implementation of SMOTE.



| | | Cohen_Kappa_Score | F1_Score | AUC | Accuracy |
|---|---|---|---|---|---|
| Logistic | Binary | 64.1 | 78.5 | 82.3 | 82.8 |
| | Multi | 65.3 | 79.1 | 82.9 | 83.3 |

**Fig 5.4: Metrics before SMOTE**



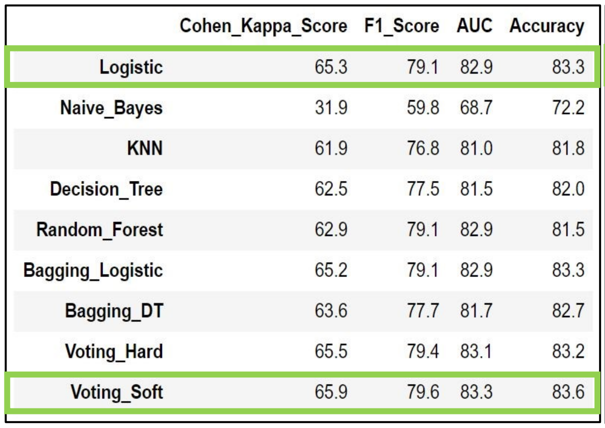
| | | Cohen_Kappa_Score | F1_Score | AUC | Accuracy |
|---|---|---|---|---|---|
| Smote_Logistic | Binary | 64.4 | 80.3 | 84.1 | 82.0 |
| | Multi | 65.9 | 80.9 | 84.7 | 82.8 |

**Fig 5.5: Metrics after SMOTE**

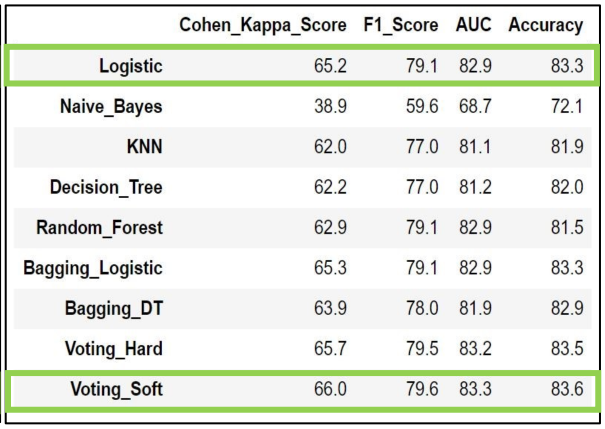
As you can see, there was only a slight improvement in all the metrics after implementing SMOTE. Therefore, in our case SMOTE was not very effective and therefore was not implemented further.

## 5.5 More Ensemble Techniques

After eliminating the binary sub-class approach and SMOTE, we went ahead to perform some more ensemble algorithms to check if there would be any significant improvement in metrics.
We made predictions using 4 simple algorithms and 5 ensemble techniques and compared the results using K-Fold Cross Validation for number of folds equal to 5 and 10. Below are the results.



| | Cohen_Kappa_Score | F1_Score | AUC | Accuracy |
|---|---|---|---|---|
| Logistic | 65.3 | 79.1 | 82.9 | 83.3 |
| Naive_Bayes | 31.9 | 59.8 | 68.7 | 72.2 |
| KNN | 61.9 | 76.8 | 81.0 | 81.8 |
| Decision_Tree | 62.5 | 77.5 | 81.5 | 82.0 |
| Random_Forest | 62.9 | 79.1 | 82.9 | 81.5 |
| Bagging_Logistic | 65.2 | 79.1 | 82.9 | 83.3 |
| Bagging_DT | 63.6 | 77.7 | 81.7 | 82.7 |
| Voting_Hard | 65.5 | 79.4 | 83.1 | 83.2 |
| Voting_Soft | 65.9 | 79.6 | 83.3 | 83.6 |



| | Cohen_Kappa_Score | F1_Score | AUC | Accuracy |
|---|---|---|---|---|
| Logistic | 65.2 | 79.1 | 82.9 | 83.3 |
| Naive_Bayes | 38.9 | 59.6 | 68.7 | 72.1 |
| KNN | 62.0 | 77.0 | 81.1 | 81.9 |
| Decision_Tree | 62.2 | 77.0 | 81.2 | 82.0 |
| Random_Forest | 62.9 | 79.1 | 82.9 | 81.5 |
| Bagging_Logistic | 65.3 | 79.1 | 82.9 | 83.3 |
| Bagging_DT | 63.9 | 78.0 | 81.9 | 82.9 |
| Voting_Hard | 65.7 | 79.5 | 83.2 | 83.5 |
| Voting_Soft | 66.0 | 79.6 | 83.3 | 83.6 |

**Fig 5.6: n_fold = 5** **Fig 5.7: n_fold = 10**

The base estimators taken into voting classifiers were – Logistic Regression, Naives_Bayes, KNN, Decision Tree, Random Forest and Logistic Regression with Bagging. In both cases, the best performing models were Logistic Regression and Soft Voting Classifier.

## 6 Conclusions

- When compared to any combination of drugs, any drug given exclusively showed the highest efficiency (i.e. 68%) in treating diabetes.
- Among all our models, Soft Voting Classifier and Logistic regression yield the best performance.
- Logistic Regression being a simpler algorithm and much more interpretable, we would recommend it for suggesting one of the two treatments to future patients.

## *Bibilography*

1. G. E. Umpierrez, S. D. Isaacs, N. Bazargan, X. You, L. M. Thaler, and A. E. Kitabchi, "Hyperglycemia: an independent marker of in-hospital mortality in patients with undiagnosed diabetes," Journal of Clinical Endocrinology and Metabolism, vol. 87, no. 3, pp. 978–982, 2002. View at Publisher · View at Google Scholar · View at Scopus
2. C. S. Levetan, M. Passaro, K. Jablonski, M. Kass, and R. E. Ratner, "Unrecognized diabetes among hospitalized patients," Diabetes Care, vol. 21, no. 2, pp. 246–249, 1998. View at Google Scholar · View at Scopus
3. S. E. Siegelaar, J. B. L. Hoekstra, and J. H. Devries, "Special considerations for the diabetic patient in the ICU; targets for treatment and risks of hypoglycaemia," Best Practice and Research: Clinical Endocrinology and Metabolism, vol. 25, no. 5, pp. 825–834, 2011. View at Publisher · View at Google Scholar · View at Scopus
4. A. G. Pittas, R. D. Siegel, and J. Lau, "Insulin therapy for critically ill hospitalized patients: a meta-analysis of randomized controlled trials," Archives of Internal Medicine, vol. 164, no. 18, pp. 2005–2011, 2004. View at Publisher · View at Google Scholar · View at Scopus
5. A. C. Tricco, N. M. Ivers, J. M. Grimshaw et al., "Effectiveness of quality improvement strategies on the management of diabetes: a systematic review and meta-analysis," The Lancet, vol. 379, no. 9833, pp. 2252–2261, 2012. View at Publisher · View at Google Scholar
6. M. C. Lansang and G. E. Umpierrez, "Management of inpatient hyperglycemia in noncritically ill patients," Diabetes Spectrum, vol. 21, no. 4, pp. 248–255, 2008. View at Publisher · View at Google Scholar· View at Scopus
7. R. Vinik and J. Clements, "Management of the hyperglycemic inpatient: tips, tools, and protocols for the clinician," Hospital Practice, vol. 39, no. 2, pp. 40–46, 2011. View at Google Scholar · View at Scopus
8. K. J. Cios and G. W. Moore, "Uniqueness of medical data mining," Artificial Intelligence in Medicine, vol. 26, no. 1-2, pp. 1–24, 2002. View at Publisher · View at Google Scholar · View at Scopus
9. A. Frank and A. Asuncion, UCI Machine Learning Repository, University of California, School of Information and Computer Science, 2010.