Group 1

PREDICTION OF PATIENT’S READMISSION PROBABILITY IN U.S. HOSPITALS.

Abstract

Hospital readmission is an indicator of the quality of care and plays a significant role in total medical costs. A large proportion of hospital costs are attributed to a small percentage of patients with chronic medical conditions who have repeated visits. The Center for Medicare & Medicaid Services (CMS) has implemented a new scheme where hospitals will be penalized depending on readmissions.

Mentored By

Shirude Shashank Prakash

Submitted By

- AnandRaj C
- Jeshurun Paul S
- Prashanth Srinivasan
- Sriram Shanmugam
- Victor Johnson W

GREAT LAKES
INSTITUTE OF MANAGEMENT
Global Mindset - Indian Roots

greatlearning

Table of Contents

Acknowledgement

The study on Prediction of patient’s readmission probability in U.S. hospitals was motived by the fact that about $566 million is being charged as penalty from 2600 hospitals by Medicare for readmission of patients under 30 days (for selective illness) from the day of discharge.

Our mentor and guide Shirude Shashank Prakash was excellent in training us and teaching us all the techniques, his support and guidance were immense. We would like to thank all our faculty members for their encouragement and guidance without which we could have not completed this project. A special thanks to Dr. Mahesh Anand S who did more than prepare us academically, his confidence and dedication in teaching not only made learning fun and easy for us, but also had raised our bars in exploring further.

We would like to thank all our peers, with whom the learning grew exponentially.

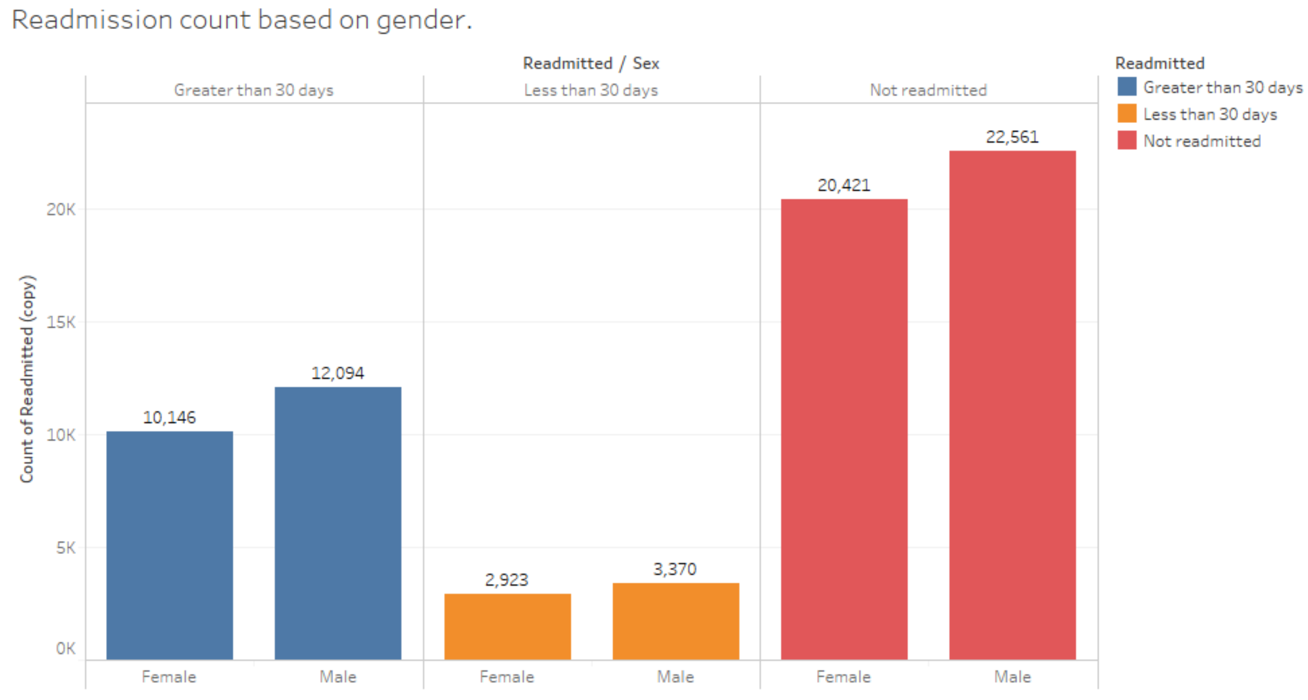
1. Introduction

"The enjoyment of the highest attainable standard of health is one of the fundamental rights of every human being without distinction of race, religion, political belief, economic or social condition"

- World Health Organization (WHO).

The right to health for all people means that everyone should have access to the health services they need, anywhere and anytime. But, Is the quality of healthcare provided, up to the standard? is the next question. The Center of Medicare & Medicaid Services (CMS) has implemented the Hospital Readmission Reduction Program (HRRP), where CMS penalizes the hospitals providing sub-standard quality of service by linking payment to the quality of hospital care.

Hospital readmission is a high-priority health care quality measure and target for cost reduction, particularly within 30 days of discharge (30-day readmission, or early readmission). Patients with diabetes may be at higher risk of readmission than those without diabetes. The burden of diabetes among hospitalized patients, however, is substantial, growing, and costly, and readmissions contribute a significant portion of this burden. Reducing readmission rates among patients with diabetes has the potential to greatly reduce health care costs while simultaneously improving care. Diabetes is also associated with an increased risk of readmission in patients hospitalized for cardiac surgery, heart failure, acute myocardial infarction, stroke, or liver disease. Considered together, these data consist details of diabetic patients which provides the effect of readmission risk. If the readmission rates increase, it might have some serious impact not only on hospital reputation but also on the health of our future generations.



Count of Readmitted (copy) for each Sex broken down by Readmitted. Colour shows details about Readmitted. The marks are labelled by count of Readmitted (copy).

1.1 Need of the Study

The basic need of this study is to reduce the readmission rate which will have a greater reduction in extra cost and focus on improving the infrastructure with high technology and to support research studies.

1.2 Objectives

Studying the Health fact database which collected comprehensive clinical records across hospitals throughout the United States.

Identify the level of significance that each variable contributes to the patient’s readmission rate. Interpreting the variables to find out the possible claims for readmission adhering for a major population in data.

To help doctors identify unique patterns within the data which in turn might help them revise certain practices in hospitals.

The goal of the analysis is to allow health centers to better anticipate and address unplanned readmissions while improving their quality of care and cost-efficiency

1.3 Scope of Study

The aim of the study is to reduce and predict the early readmission of patients which will help the hospital financially. This would be likely to reduce pressure on the system and on patients caused by unnecessary overstays and readmissions.

1.4 Data Source

1.4.1 ADA Dataset:

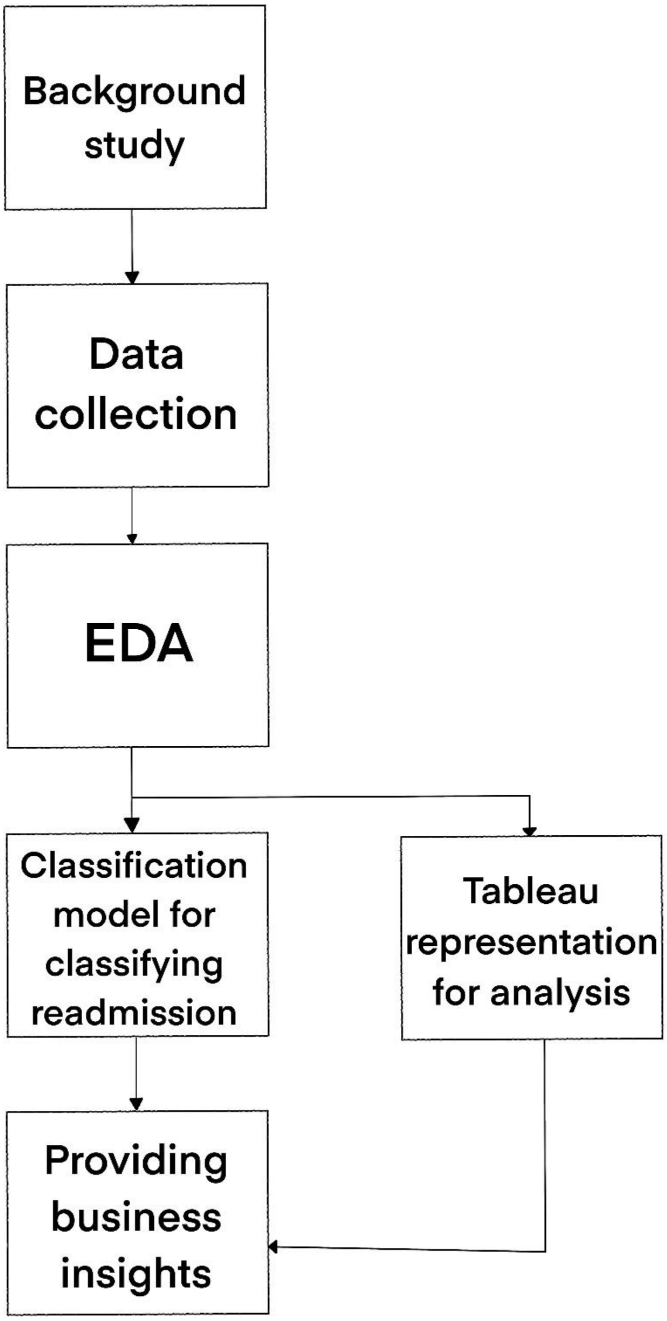
This dataset is taken from the Health Facts database (Cerner Corporation, Kansas City, MO), a national data warehouse that collects comprehensive clinical records across hospitals throughout the United States. The Health Facts data used was an extract representing 10 years (1999–2008) of clinical care 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 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.

Since the primary interest is in factors that lead to early readmission, the readmission attribute (outcome) is defined as having two values: “readmitted,” if the patient was readmitted within 30 days of discharge or “otherwise,” which covers both readmission after 30 days and no readmission at all. The values of the readmission attribute were determined by examination of all patient records in the database to determine the first inpatient visit after discharge. Note that 30 days was chosen based on criteria often used by funding agencies.

ADA Dataset - Number of (Observations, features): (101766,50)

1.5 Methodology

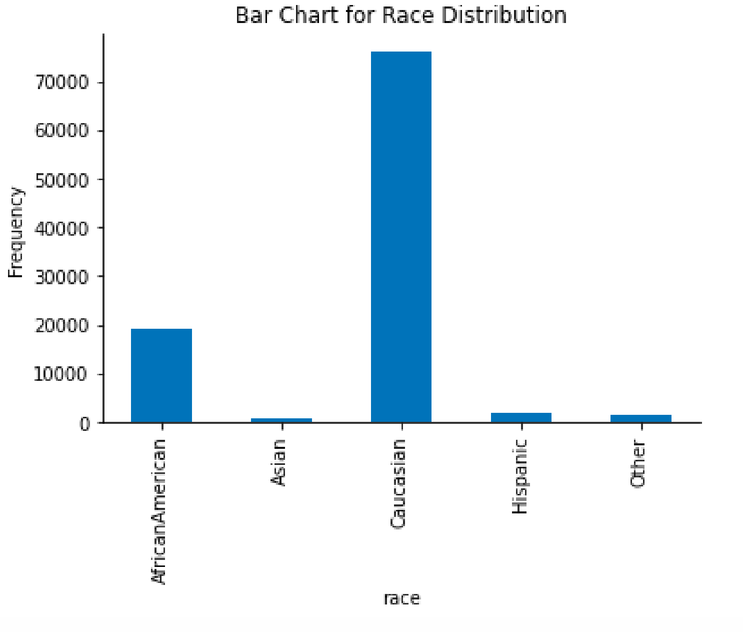

Background study
Data collection
EDA
Classification model for classifying readmission
Tableau representation for analysis
Providing business insights

2. EDA

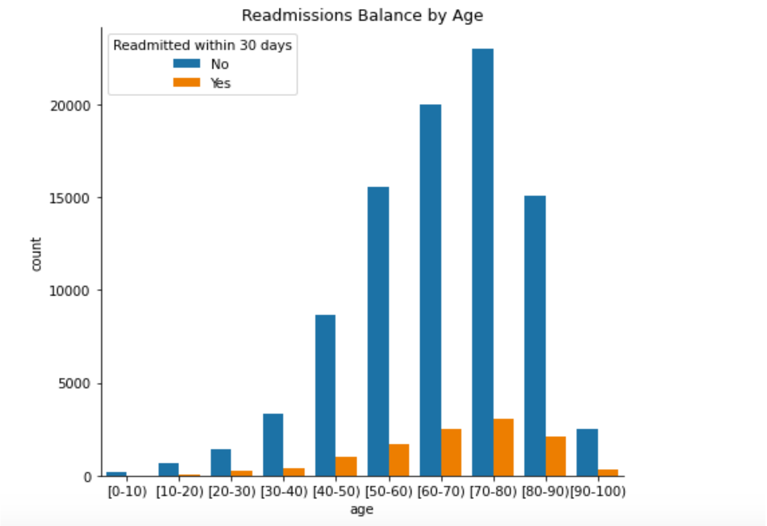
2.1 Data Insights

In this section the insights on all the features with bivariate and multivariate analysis are presented visually.

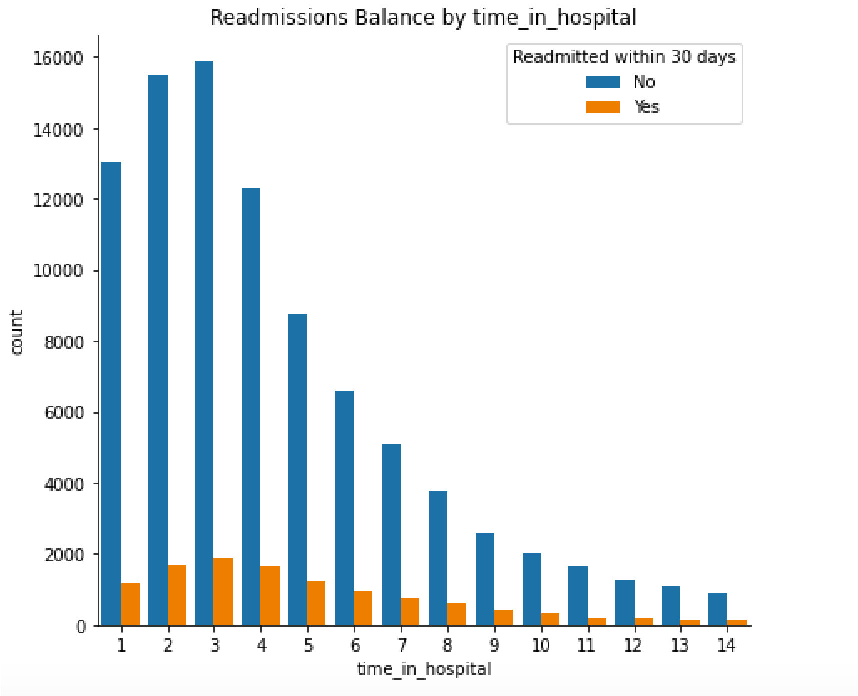
This plot shows the distribution of no of patients in each Race in the data, there are five different categories.



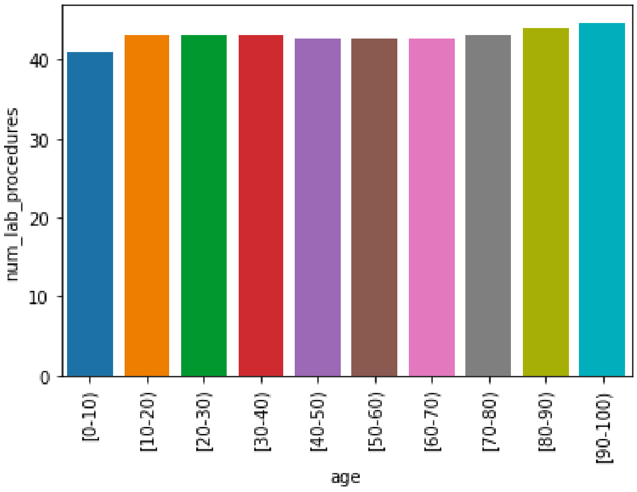
This plot shows the distribution of Age in the data, with respect to whether or not they have readmitted within 30 days.

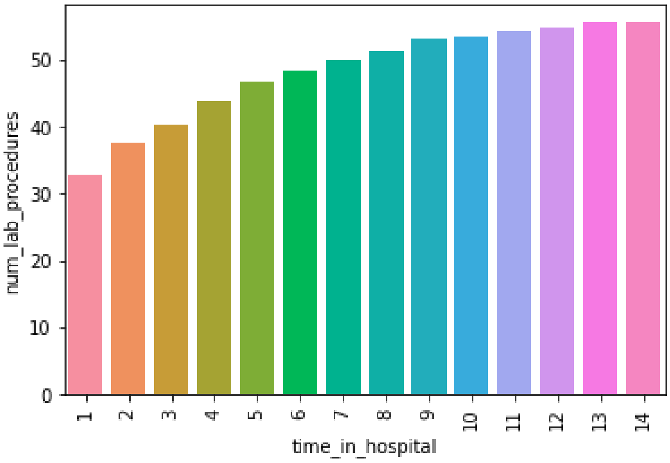


This plot shows the distribution of time in hospital in the data, with respect to whether or not they have readmitted within 30 days.

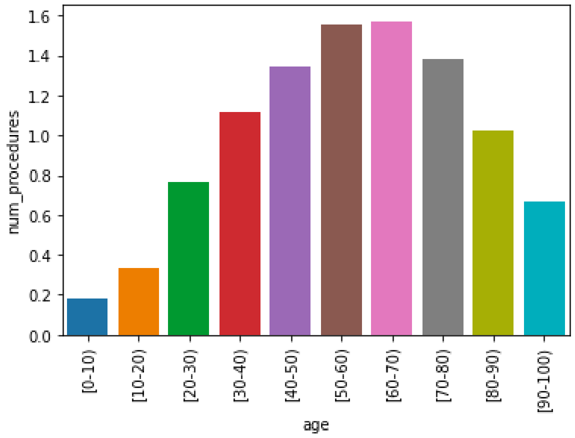


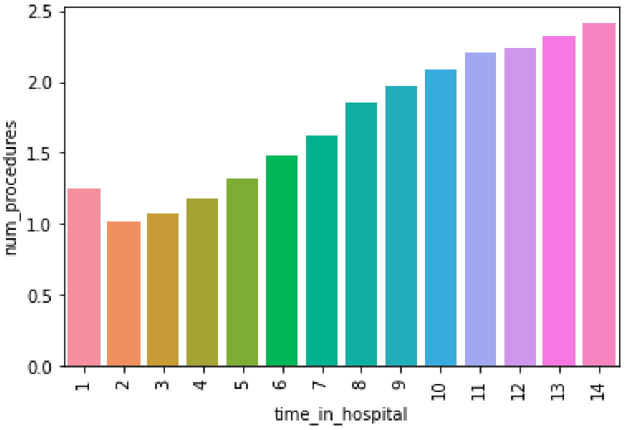
This plot shows the distribution of Age and time in hospital in the data, with respect to the number of lab procedures done on the patients.



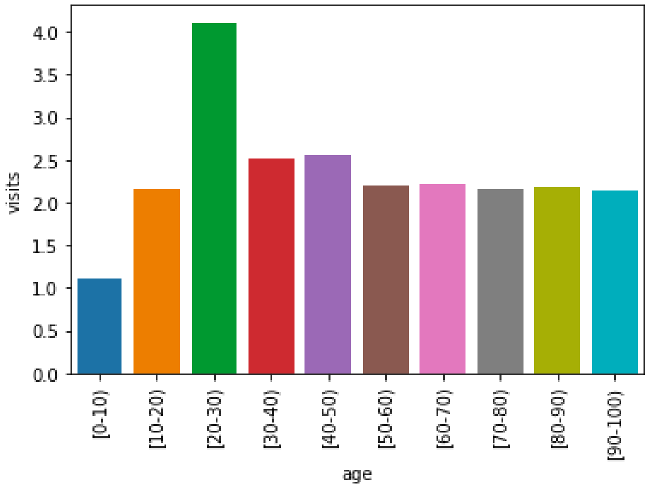


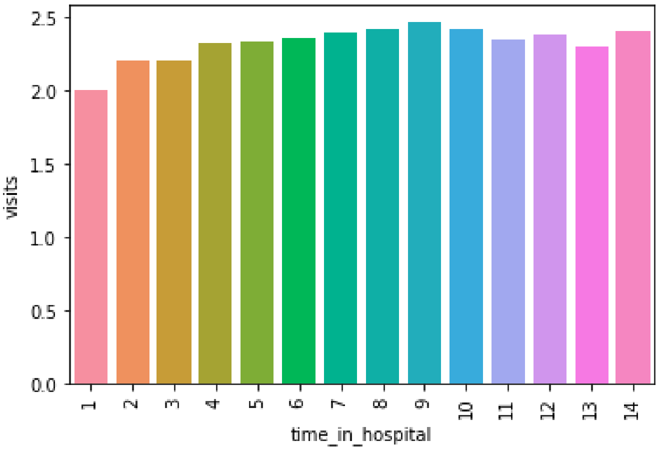
This plot shows the distribution of Age and time in hospital in the data, with respect to the number of procedures done on the patients.



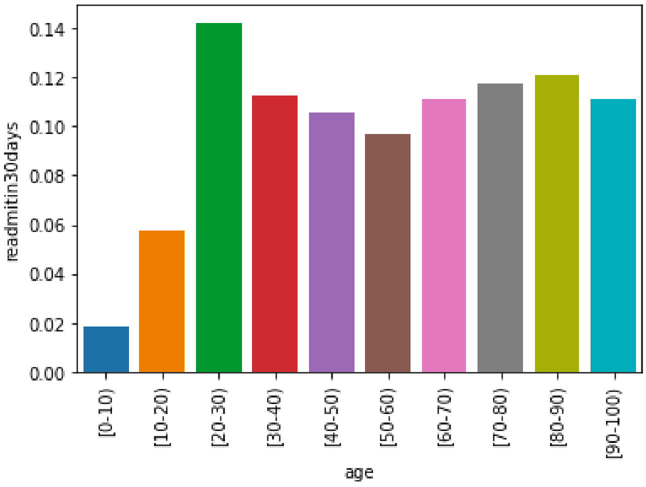


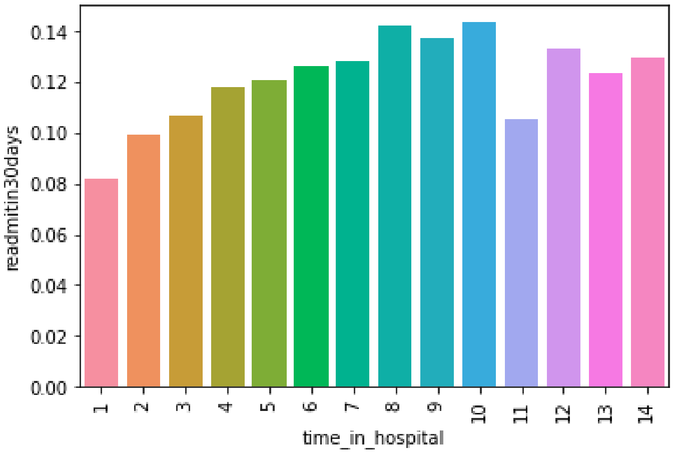
This plot shows the distribution of Age and time in hospital in the data, with respect to the number of visits to the hospital by the patients.



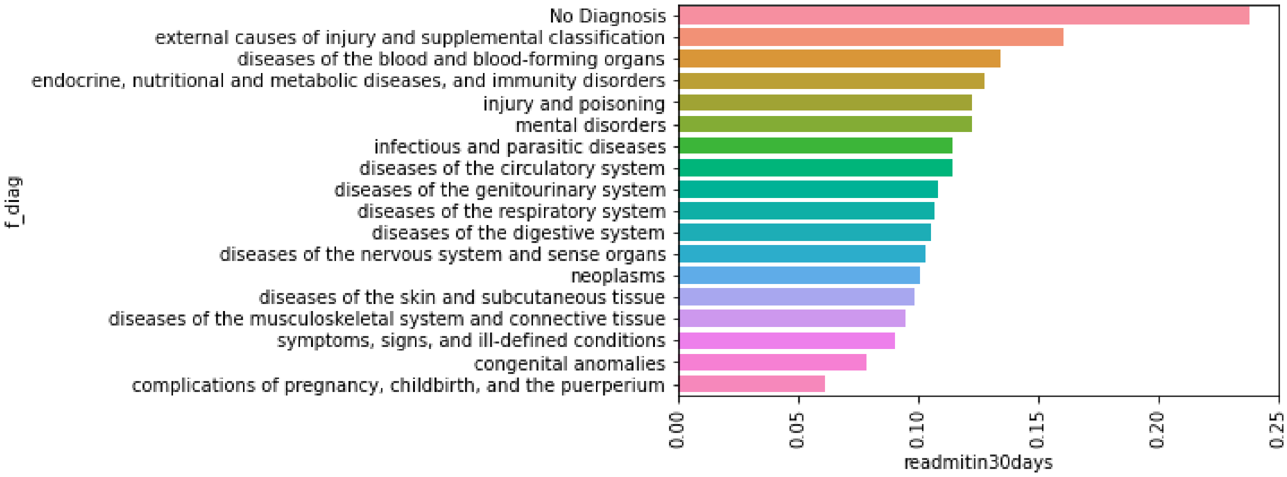


This plot shows the distribution of Age and time in hospital in the data, with respect to the patient’s readmission within 30 days.

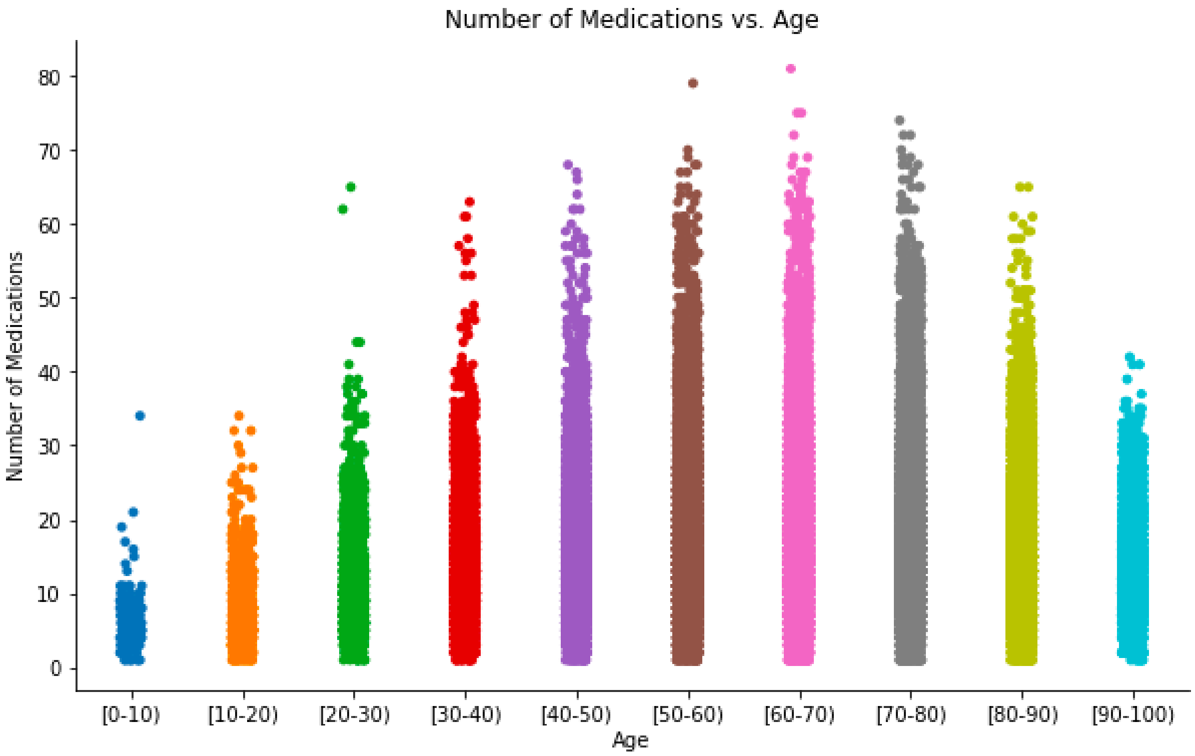




This plot shows the distribution of the diagnosis for different patients and compares it with their readmission within 30 days.

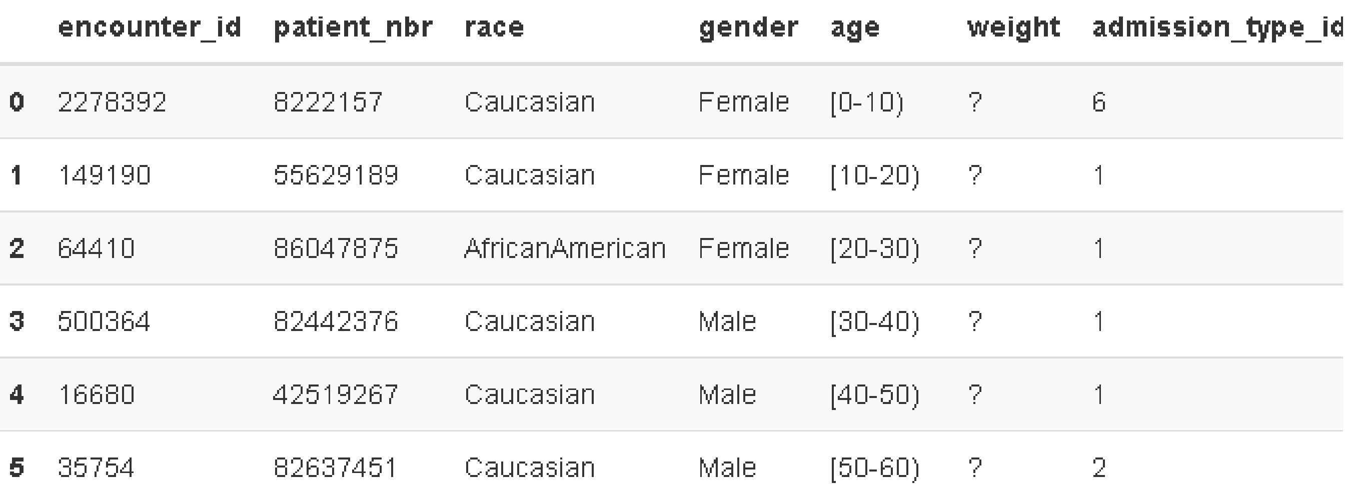


This plot shows the distribution of number of medications with patients of different age categories.



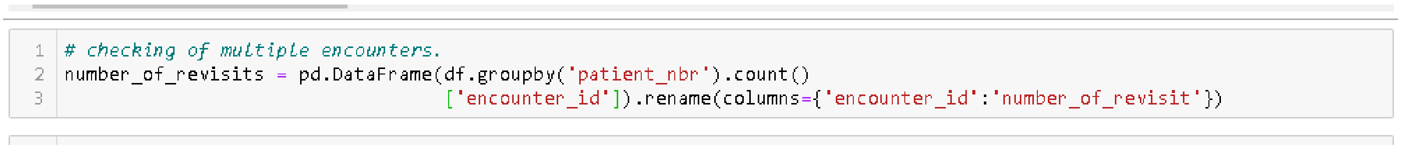
2.2 Data Cleaning

View of first 5 rows and 7 columns of the dataset



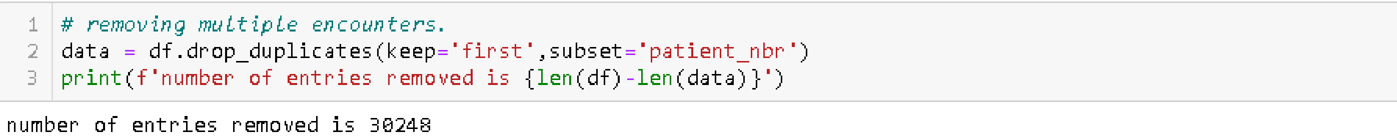
	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id
0	2278392	8222157	Caucasian	Female	[0-10)	?	6
1	149190	55629189	Caucasian	Female	[10-20)	?	1
2	64410	86047875	AfricanAmerican	Female	[20-30)	?	1
3	500364	82442376	Caucasian	Male	[30-40)	?	1
4	16680	42519267	Caucasian	Male	[40-50)	?	1
5	35754	82637451	Caucasian	Male	[50-60)	?	2

- ➢ A new feature named "number_of_revisits" is created so that distinct count of patients who visited hospitals for the first instance can be found and the duplicates can be removed, without losing the information.



```
# checking of multiple encounters.
number_of_revisits = pd.DataFrame(df.groupby('patient_nbr').count()
                                  ['encounter_id']).rename(columns={'encounter_id':'number_of_revisit'})
```

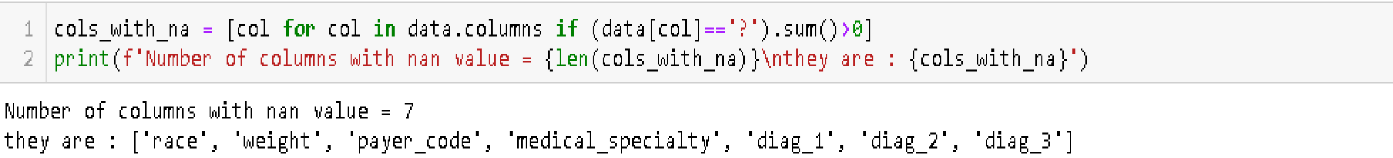
Here, the duplicates are removed using the below code,



```
# removing multiple encounters.
data = df.drop_duplicates(keep='first',subset='patient_nbr')
print(f'number of entries removed is {len(df)-len(data)}')
```

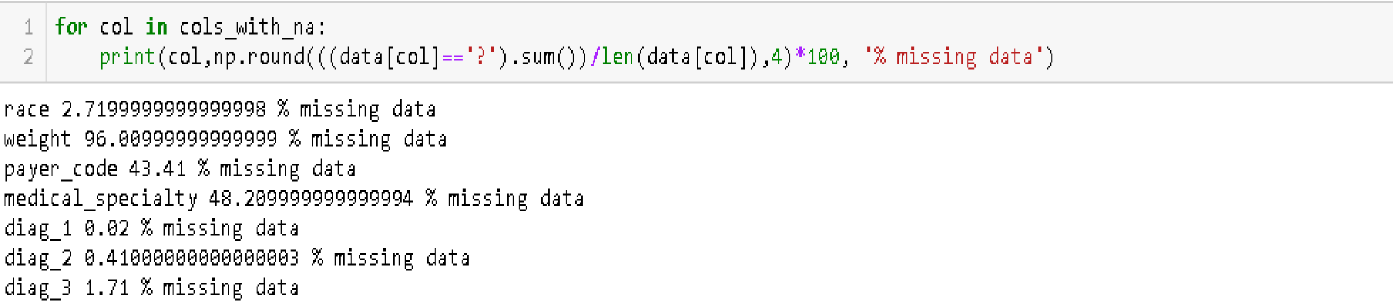
```
number of entries removed is 30248
```

With 71518 observations and 51 features, it can be proceeded to data cleaning. Missing and null values has to be checked, unfortunately this data contains "?" as null values.



```
cols_with_na = [col for col in data.columns if (data[col]=='?').sum()>0]
print(f'Number of columns with nan value = {len(cols_with_na)}\nthey are : {cols_with_na}')
```

```
Number of columns with nan value = 7
they are : ['race', 'weight', 'payer_code', 'medical_specialty', 'diag_1', 'diag_2', 'diag_3']
```

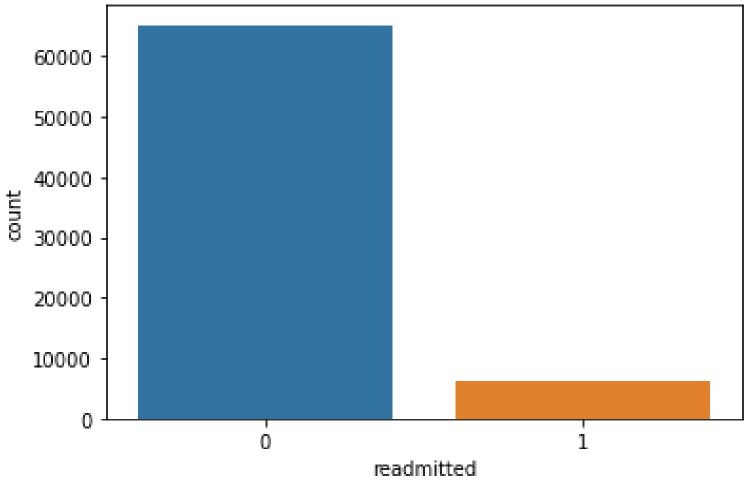


```
for col in cols_with_na:
    print(col,np.round(((data[col]=='?').sum())/len(data[col]),4)*100, '% missing data')
```

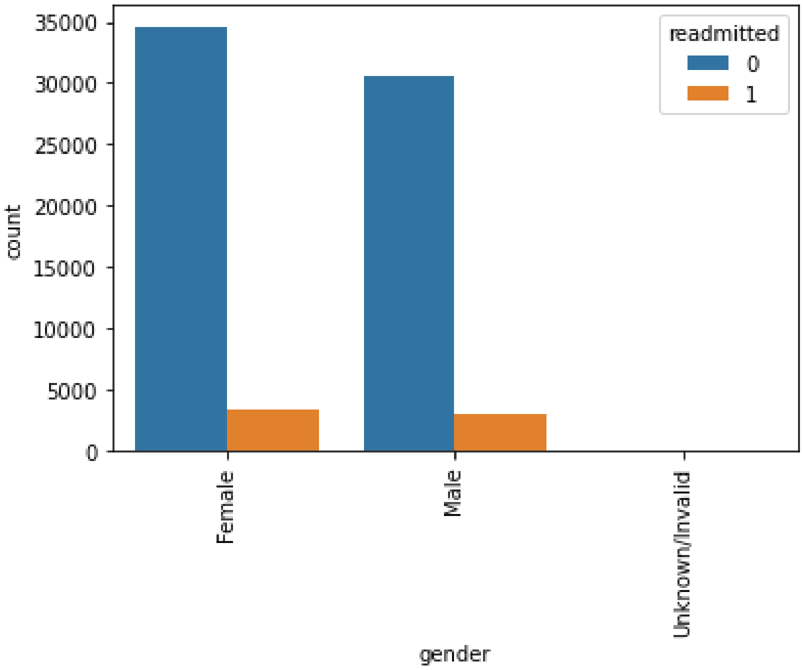
```
race 2.7199999999999998 % missing data
weight 96.00999999999999 % missing data
payer_code 43.41 % missing data
medical_specialty 48.209999999999994 % missing data
diag_1 0.02 % missing data
diag_2 0.41000000000000003 % missing data
diag_3 1.71 % missing data
```

- ➢ Feature like “weight”, “payer_code” and “medical_speciality” has some high missing vales which are not important in classifying target variable.

- ➢ The project focuses on whether the “readmission” has taken place less than 30 days from discharge or not. This grouping is such as "NO" and ">30" together are grouped as "0" and "<30" as "1".



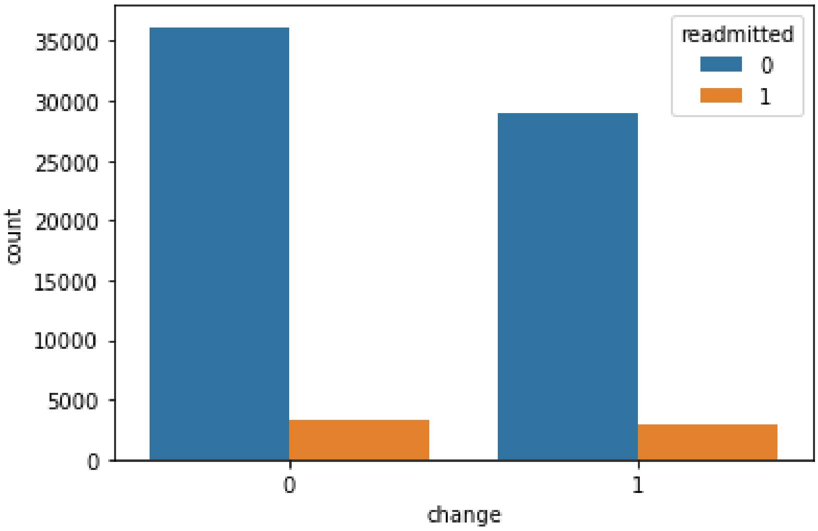
With 65225 records as 0’s and 6293 records as 1’s in readmitted feature, it can be proceeded to data cleaning.From the below Bar graph it can be can concluded that “Unknow/Invalid” data has 3 missing values which has no effect on the target variable hence it is reasonable to drop them.



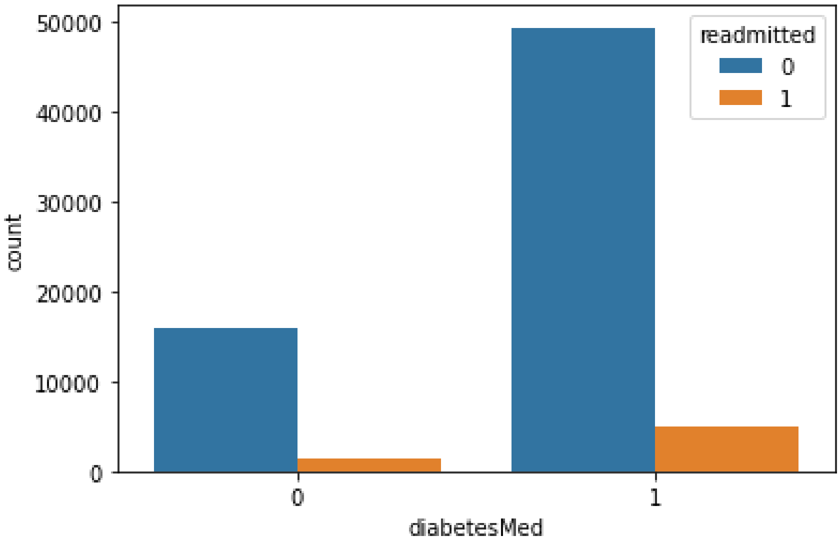
- Treating medications in the dataset. There are totally 23 different features of medications. Any medication is not going to be removed though it has huge imbalance in data, but still there are three medications like 'examide','glimepiride-pioglitazone' and 'citoglipton' which has completely 100% no data and hence has been removed.

- Other medication are treated as 0's and 1's by mapping NO to 0 and Down, Up, Steady as 1.

- Change feature having 2 unique values "Yes" and "No" are replaced with 1 and 0 respectively.



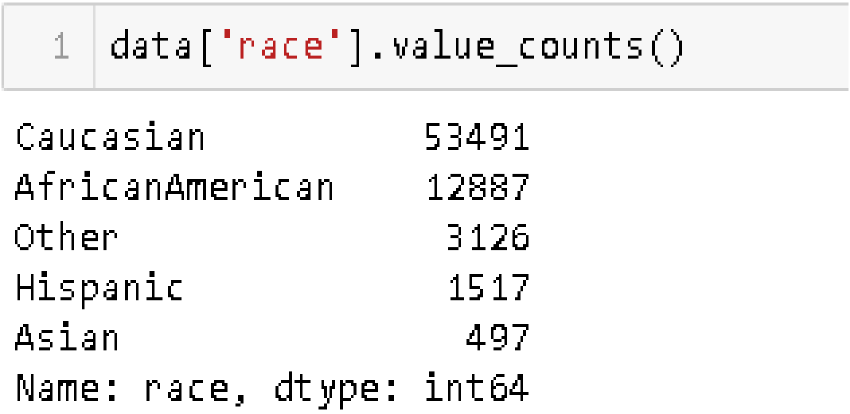
- DiabetesMed feature having 2 unique values "Yes" and "No" are replaced with 1 and 0 respectively.



2.3 Data Grouping

Here comes the challenging part in data cleaning, Data Grouping. Grouping is done to reduce the number of dummies to be created which will affect the data. Every value has been precisely grouped on each feature in a reasonable and logical way.

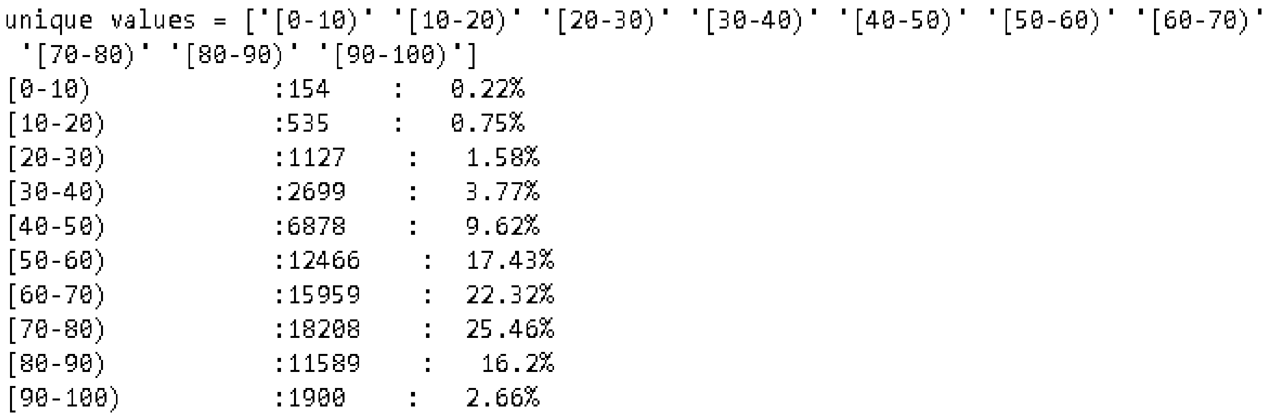
➢ Race feature has 6 unique values “Caucasian”, “AfricanAmerican”, “?”, “Other”, “Asian” and “Hispanic”. Here rows containing “?” are 1948 which is 2.72% of data which cannot be removed without proper reason, so, instead of removing or treating will null values it can replaced to “Other” category since “Other” has just 1.65% of data. Therefore, by grouping “other” and “?” dummification can be performed on the Race column. Values below show the value counts of each:



```
data['race'].value_counts()
```

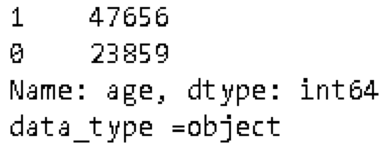
```
Caucasian          53491
AfricanAmerican    12887
Other               3126
Hispanic            1517
Asian                497
Name: race, dtype: int64
```

➢ Age feature having 10 unique values of different range as object data type. From the graph below age is grouped into 2 categories as “age below 60” and “above 60” with 0’s and 1’s respectively since it will give a partially distributed data.

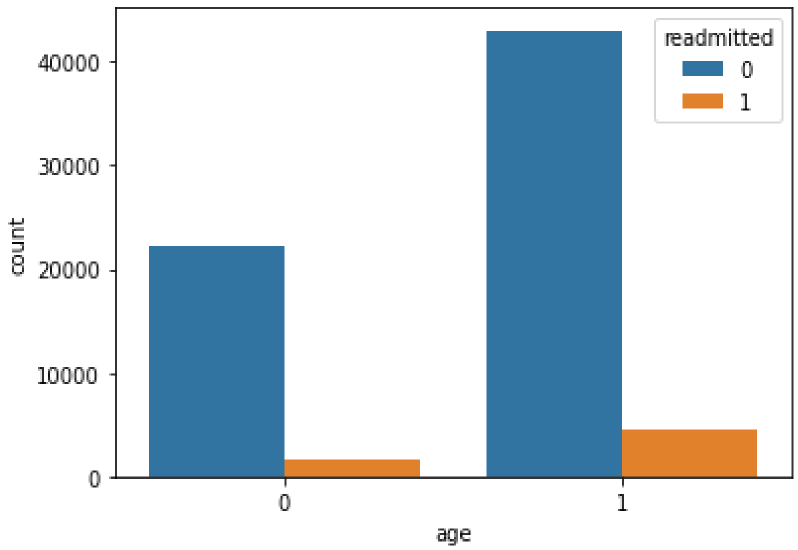


```
unique values = ['[0-10)' '[10-20)' '[20-30)' '[30-40)' '[40-50)' '[50-60)' '[60-70)'
 '[70-80)' '[80-90)' '[90-100)']
[0-10)           :154    :   0.22%
[10-20)          :535    :   0.75%
[20-30)          :1127    :   1.58%
[30-40)          :2699    :   3.77%
[40-50)          :6878    :   9.62%
[50-60)          :12466    :  17.43%
[60-70)          :15959    :  22.32%
[70-80)          :18208    :  25.46%
[80-90)          :11589    :   16.2%
[90-100)         :1900    :   2.66%
```

After grouping



```
1    47656
0    23859
Name: age, dtype: int64
data_type =object
```



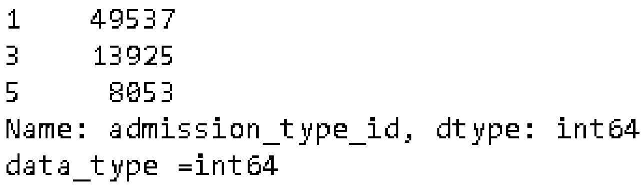
➢ Admission_type_id have 6 independent unique values which can be grouped.

1	Emergency
2	Urgent
3	Elective
4	Newborn
5	Not Aavailable
6	NULL
7	Trauma Centre
8	Not Mapped

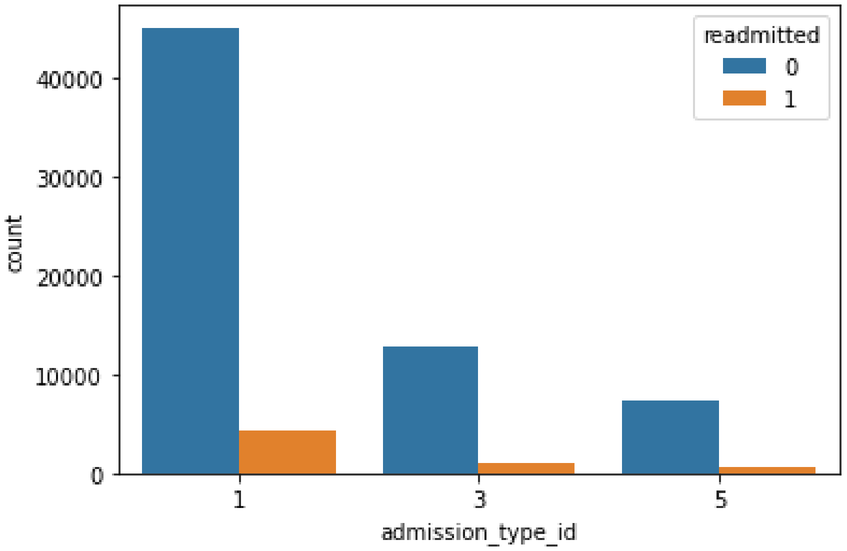
From the above data,

- “Emergency”, “Urgent” and “Trauma centre” can be replaced with 1
- “Null”, “Not mapped”, “Not available” are grouped and replaced with 5,
- Since “New-born” and “Elective” are considered as “Other” it is replaced with 3.

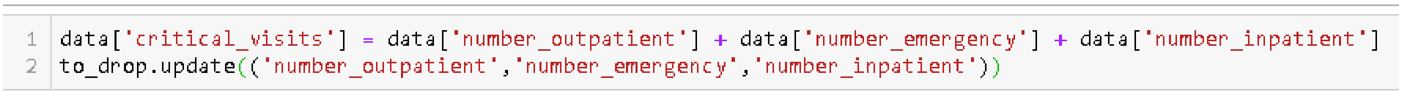
After Grouping



```
1     49537
3     13925
5      8053
Name: admission_type_id, dtype: int64
data_type =int64
```

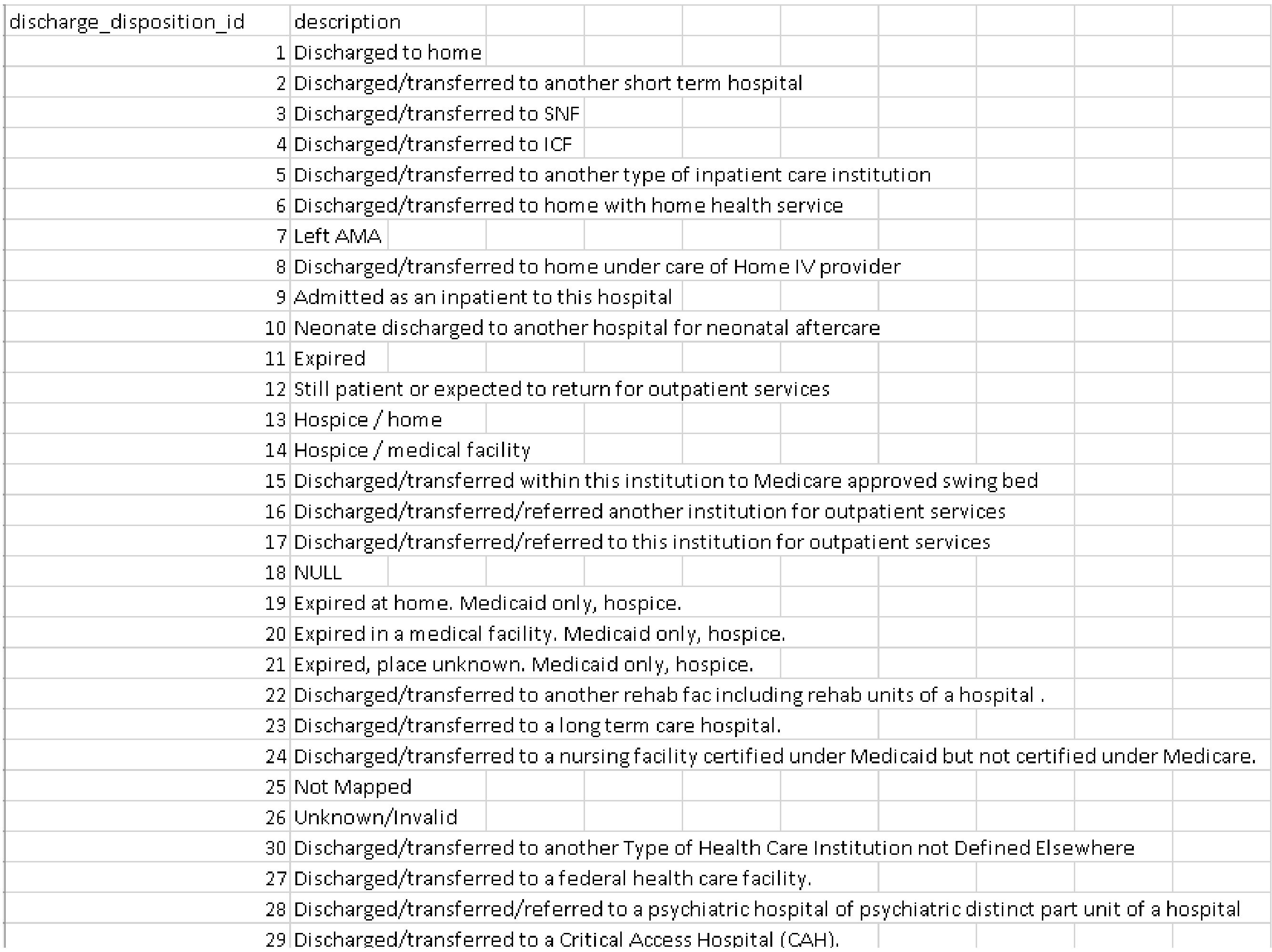


➢ Since no changes can be made to “Num_medications”, “Outpatient” and “Inpatient” feature it can be added together as single feature “Critical_Count” by following code,



```
data['critical_visits'] = data['number_outpatient'] + data['number_emergency'] + data['number_inpatient']
to_drop.update(('number_outpatient','number_emergency','number_inpatient'))
```

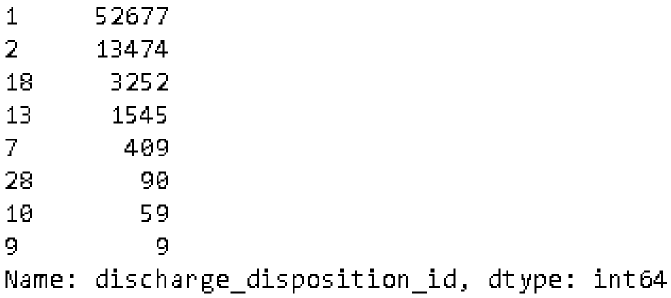
➢ Discharge_disposition_id has 26 independent values which also can be grouped as follows,



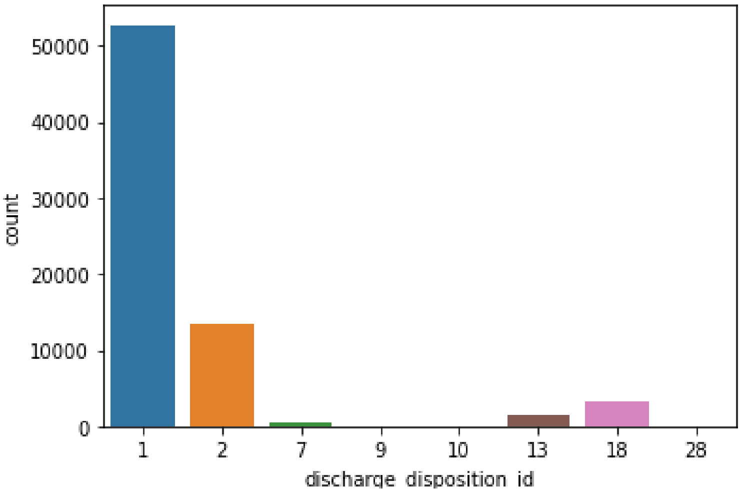
discharge_disposition_id	description
1	Discharged to home
2	Discharged/transferred to another short term hospital
3	Discharged/transferred to SNF
4	Discharged/transferred to ICF
5	Discharged/transferred to another type of inpatient care institution
6	Discharged/transferred to home with home health service
7	Left AMA
8	Discharged/transferred to home under care of Home IV provider
9	Admitted as an inpatient to this hospital
10	Neonate discharged to another hospital for neonatal aftercare
11	Expired
12	Still patient or expected to return for outpatient services
13	Hospice / home
14	Hospice / medical facility
15	Discharged/transferred within this institution to Medicare approved swing bed
16	Discharged/transferred/referred another institution for outpatient services
17	Discharged/transferred/referred to this institution for outpatient services
18	NULL
19	Expired at home. Medicaid only, hospice.
20	Expired in a medical facility. Medicaid only, hospice.
21	Expired, place unknown. Medicaid only, hospice.
22	Discharged/transferred to another rehab fac including rehab units of a hospital .
23	Discharged/transferred to a long term care hospital.
24	Discharged/transferred to a nursing facility certified under Medicaid but not certified under Medicare.
25	Not Mapped
26	Unknown/Invalid
30	Discharged/transferred to another Type of Health Care Institution not Defined Elsewhere
27	Discharged/transferred to a federal health care facility.
28	Discharged/transferred/referred to a psychiatric hospital of psychiatric distinct part unit of a hospital
29	Discharged/transferred to a Critical Access Hospital (CAH).

- 1,6,8 are taken as '1' since all represents or relates to discharge/transferred to home in some sort of way.
- 28,29 is taken as '28' since they both relates to hospital with different abilities.
- 2,3,4,5,22,23,24,30 are taken as '2' since SNF IS skilled nursing facilities, ICF is Intermediate Care Facility where others are referring to care units.
- 10,12,15,16,17 are taken as 10, where, Outpatients means “The patient who attends a hospital for treatment without staying there overnight." and many are referring to outpatients. Neonate means “A newborn child”, where, as per condition it is discharged to another hospital later so it is also considered here.
- 18,25,26 deals with null or unknows data entries.
- 7 is left as AMA.
- 9 is Admitted as an inpatient to this hospital.
- 27 is Discharged/transferred to a federal health care facility.

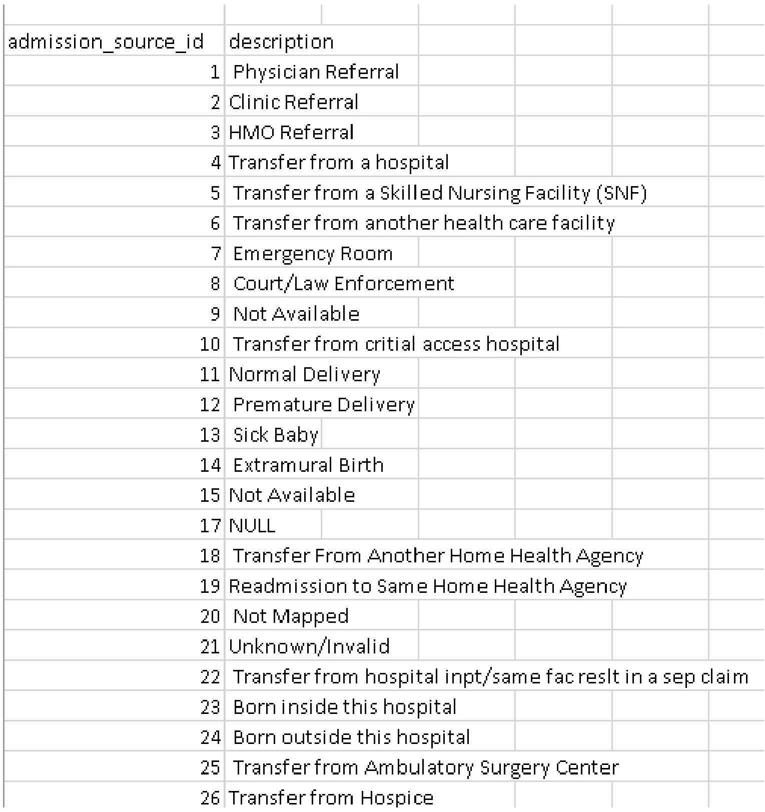
After grouping



```
1      52677
2      13474
18      3252
13      1545
7        409
28        90
10        59
9          9
Name: discharge_disposition_id, dtype: int64
```



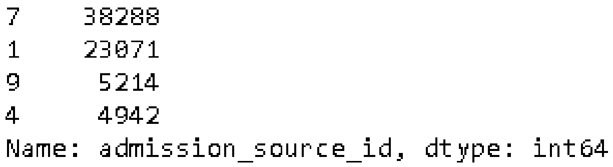
- ➢ Admission_source_id is also grouped which has 25 independent values as follows,



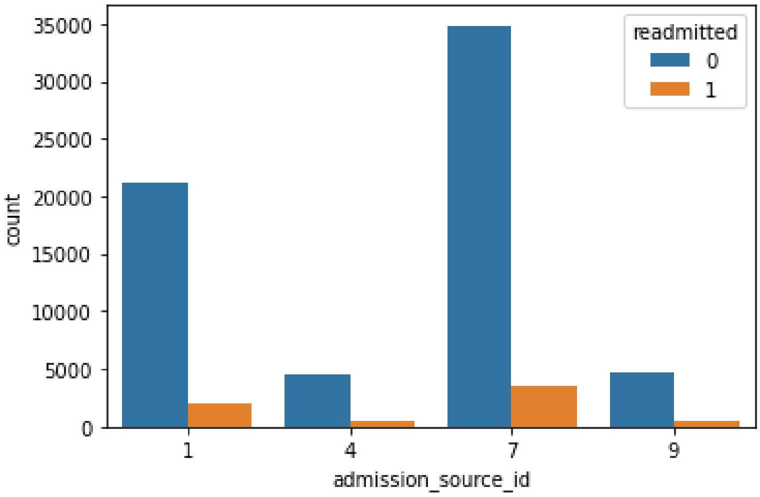
admission_source_id	description
1	Physician Referral
2	Clinic Referral
3	HMO Referral
4	Transfer from a hospital
5	Transfer from a Skilled Nursing Facility (SNF)
6	Transfer from another health care facility
7	Emergency Room
8	Court/Law Enforcement
9	Not Available
10	Transfer from critial access hospital
11	Normal Delivery
12	Premature Delivery
13	Sick Baby
14	Extramural Birth
15	Not Available
17	NULL
18	Transfer From Another Home Health Agency
19	Readmission to Same Home Health Agency
20	Not Mapped
21	Unknown/Invalid
22	Transfer from hospital inpt/same fac reslt in a sep claim
23	Born inside this hospital
24	Born outside this hospital
25	Transfer from Ambulatory Surgery Center
26	Transfer from Hospice

- 1,2,3 are referrals which are grouped as '1'.
- 9,15,17,20,21 are unknown data which are grouped as '9'.
- 12,13,14 are sick babies which are grouped as '12'.
- 4,5,6,10,22,25 are related to facility or care units which are grouped as '4'.
- 7 is left as it since it cannot grouped with any independent values.

After Grouping

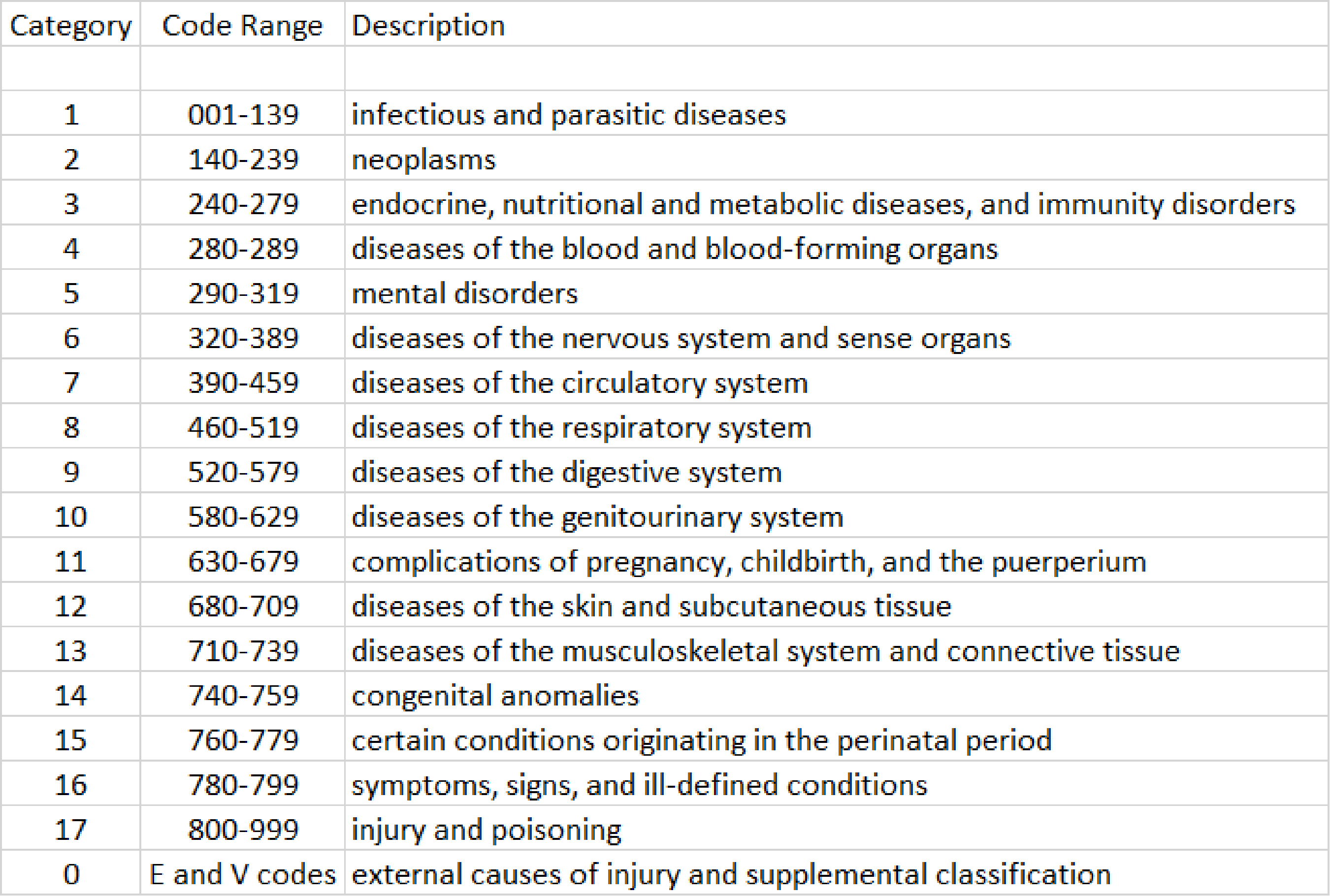


```
7    38288
1    23071
9     5214
4     4942
Name: admission_source_id, dtype: int64
```



➢ Diag1, Diag2, Diag3 has so many unique values including “?”, in order to impute them ICD9-Codes are used.
ICD9 – Codes is a list of codes for International Statistical Classification of Diseases and Related Heath Problems.

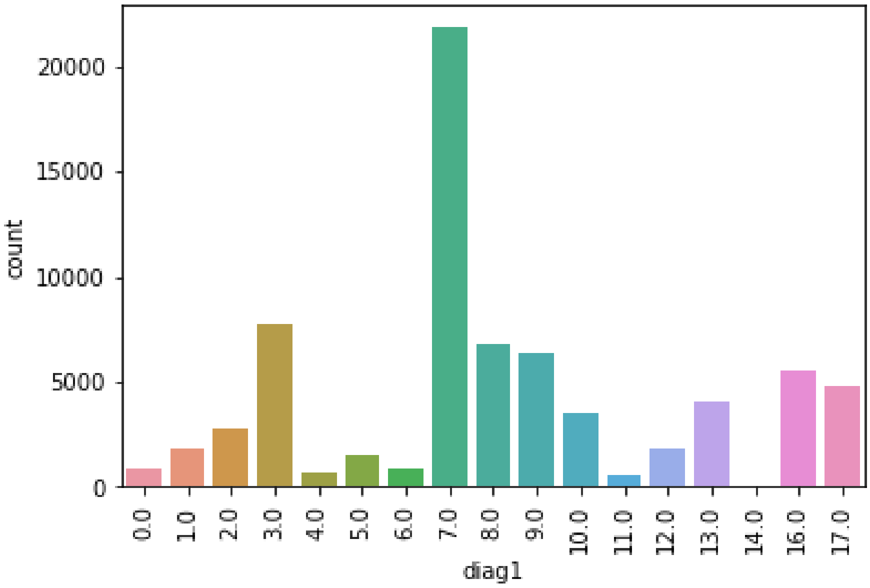
- A new Dataframe is created with diagnosis features, the code with E and V can be converted to ‘0’ and ‘?’ Can be replaced with ‘-1’.
- And also the data type has to be changed to float



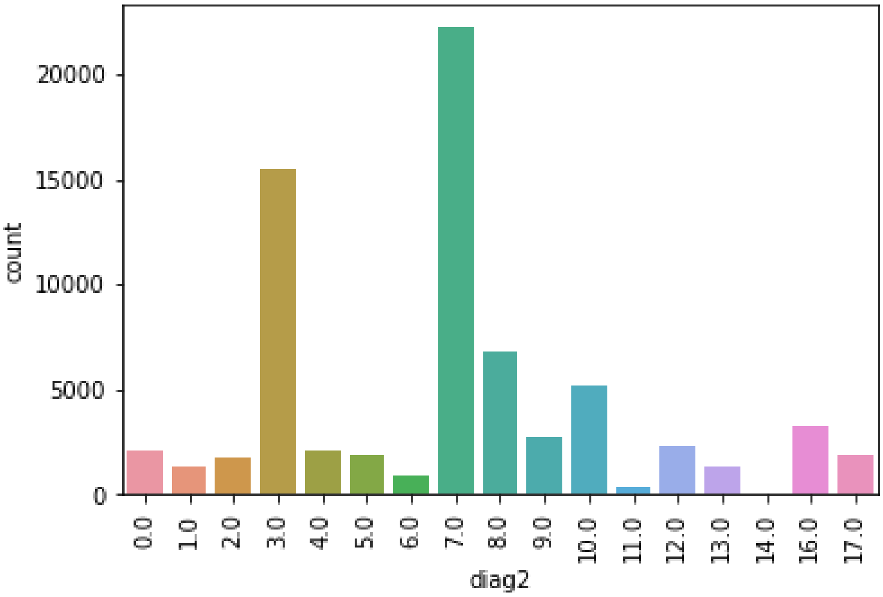
Category	Code Range	Description
1	001-139	infectious and parasitic diseases
2	140-239	neoplasms
3	240-279	endocrine, nutritional and metabolic diseases, and immunity disorders
4	280-289	diseases of the blood and blood-forming organs
5	290-319	mental disorders
6	320-389	diseases of the nervous system and sense organs
7	390-459	diseases of the circulatory system
8	460-519	diseases of the respiratory system
9	520-579	diseases of the digestive system
10	580-629	diseases of the genitourinary system
11	630-679	complications of pregnancy, childbirth, and the puerperium
12	680-709	diseases of the skin and subcutaneous tissue
13	710-739	diseases of the musculoskeletal system and connective tissue
14	740-759	congenital anomalies
15	760-779	certain conditions originating in the perinatal period
16	780-799	symptoms, signs, and ill-defined conditions
17	800-999	injury and poisoning
0	E and V codes	external causes of injury and supplemental classification

After Grouping,

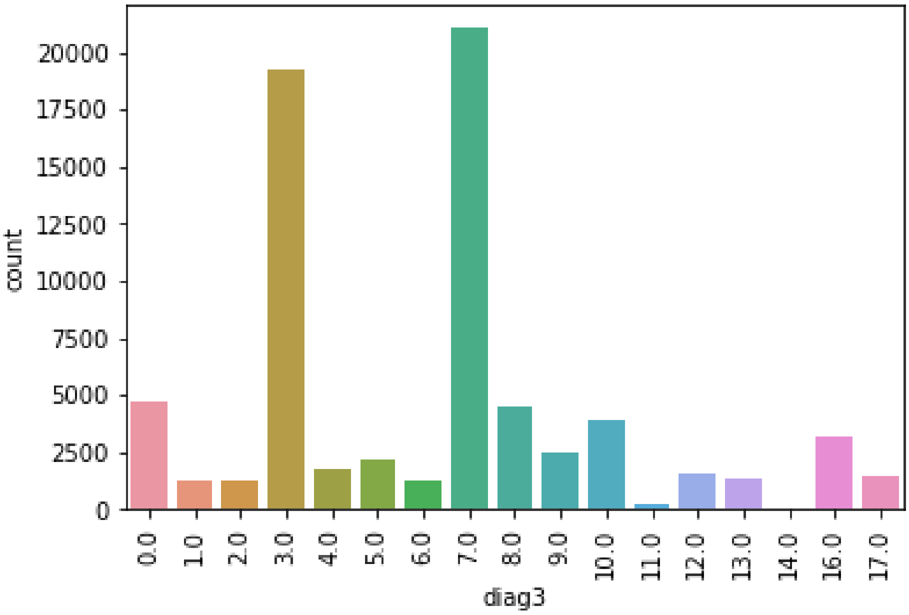
- Diag_1



- Diag_2

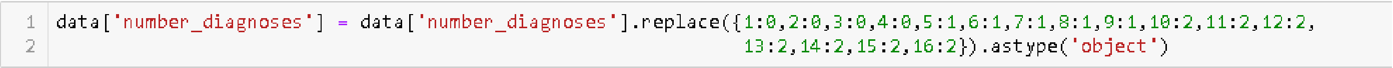

count
20000
15000
10000
5000
0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
11.0
12.0
13.0
14.0
16.0
17.0
diag2

- Diag_3

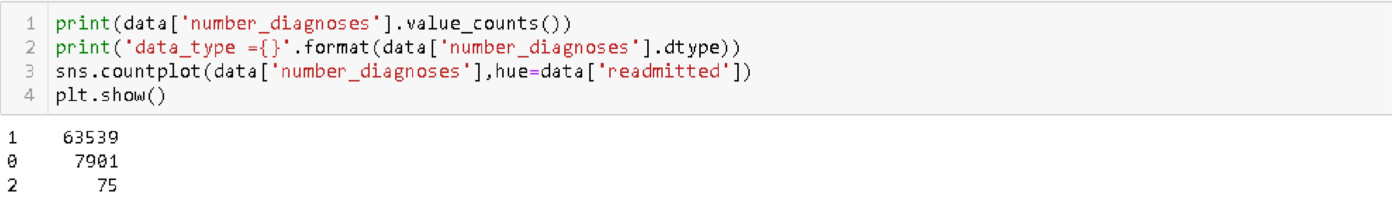

count
20000
17500
15000
12500
10000
7500
5000
2500
0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
11.0
12.0
13.0
14.0
16.0
17.0
diag3

Finally adding diag1, diag2, diag3 to dummy set.

- Number_diagnosis as 16 unique values which can be grouped as follows,
-



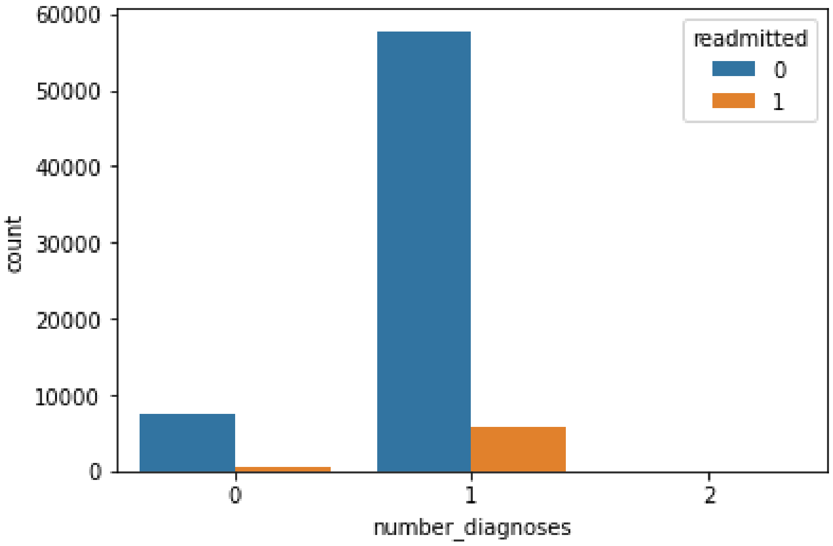
```
data['number_diagnoses'] = data['number_diagnoses'].replace({1:0,2:0,3:0,4:0,5:1,6:1,7:1,8:1,9:1,10:2,11:2,12:2,
                                                               13:2,14:2,15:2,16:2}).astype('object')
```



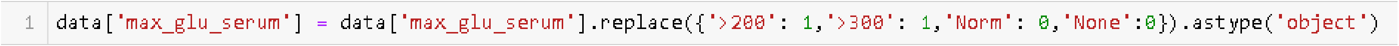
```
print(data['number_diagnoses'].value_counts())
print('data_type ={}'.format(data['number_diagnoses'].dtype))
sns.countplot(data['number_diagnoses'],hue=data['readmitted'])
plt.show()
```

```
1    63539
0     7901
2       75
```

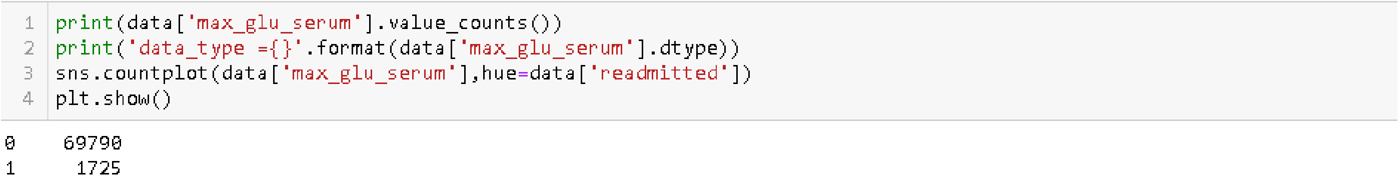
After Grouping



- Max_glu_serum having 4 independent values which can be grouped to a approximate range like replacing "Norm" with 0 and ">200",">300" with 1 and since "None" has very low value which can be grouped with "Norm" as 0 by following code,



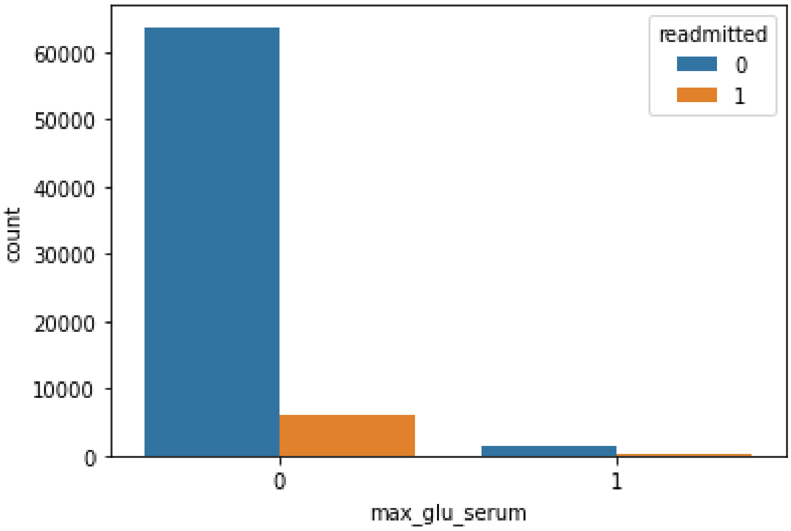
```
data['max_glu_serum'] = data['max_glu_serum'].replace({'>200': 1,'>300': 1,'Norm': 0,'None':0}).astype('object')
```



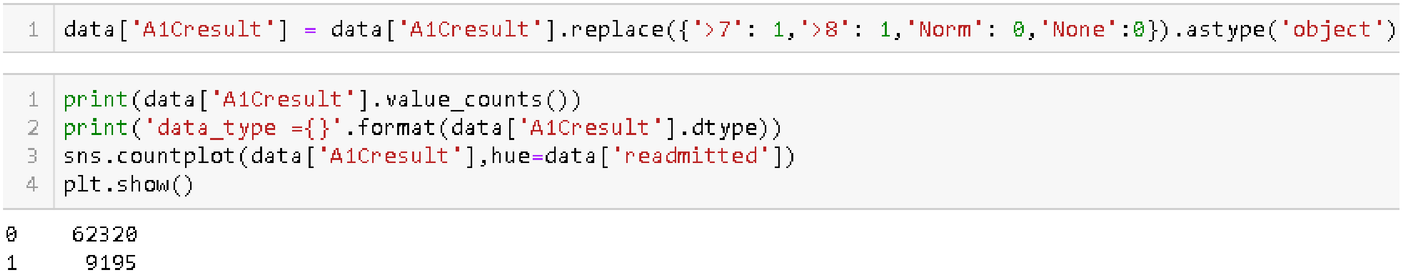
```
print(data['max_glu_serum'].value_counts())
print('data_type ={}'.format(data['max_glu_serum'].dtype))
sns.countplot(data['max_glu_serum'],hue=data['readmitted'])
plt.show()
```

```
0    69790
1     1725
```

After Grouping,



- ➢ A1Cresult feature also has 4 independent value which can be grouped as “Norm” and “Above 7” with 0, 1 respectively. Since “None” has low effect in feature it is grouped with “Norm” as 0 by following code,

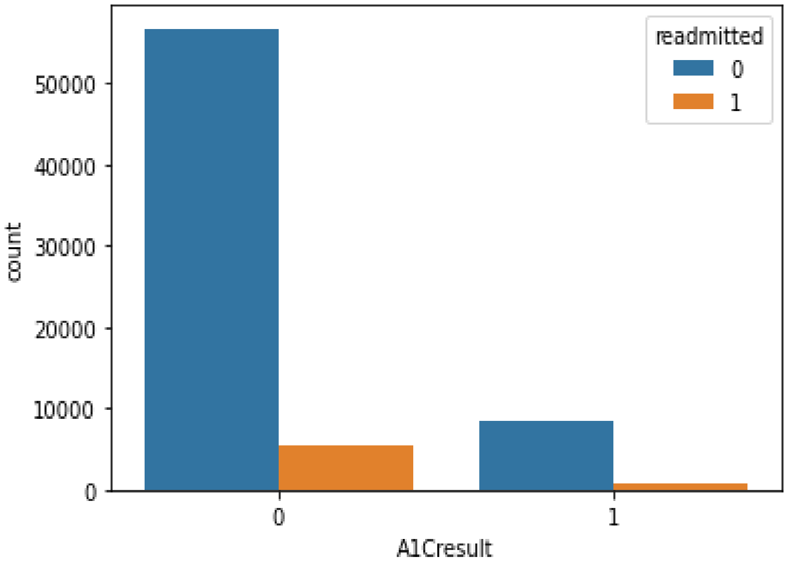


```
data['A1Cresult'] = data['A1Cresult'].replace({'>7': 1,'>8': 1,'Norm': 0,'None':0}).astype('object')
```

```
print(data['A1Cresult'].value_counts())
print('data_type ={}'.format(data['A1Cresult'].dtype))
sns.countplot(data['A1Cresult'],hue=data['readmitted'])
plt.show()
```

```
0    62320
1     9195
```

After Grouping



Those are our data grouping techniques. After cleaning and grouping dummies are created for respective features.
At first a set is created to “drop”, once cleaning is made it is added to the bucket and by using this function duplicates can be dropped and other existing features since dummies were created.

Once every process is completed the post EDA dataset will be saved as “Cleaned_dataset” as new excel file to work on feature selection process.

2.4 Outlier Treatment

Why no outlier treatment?

There are no outliers in healthcare ... only real people.

The patient outcomes for both medical outliers were observed for 30-day mortality. Approximately 11% of the total readmissions to the hospital was to patients who were medical outliers at some point during their care. Univariate analysis of medical outliers on the patient outcome revealed that medical outliers are not associated with in-hospital or 30-day mortality, but they do affect the readmission probabilities with statistically significant. That is why we are doing grouping and data cleaning to maintain the statistical significance with each other.

2.5 Statistical Significance Test:

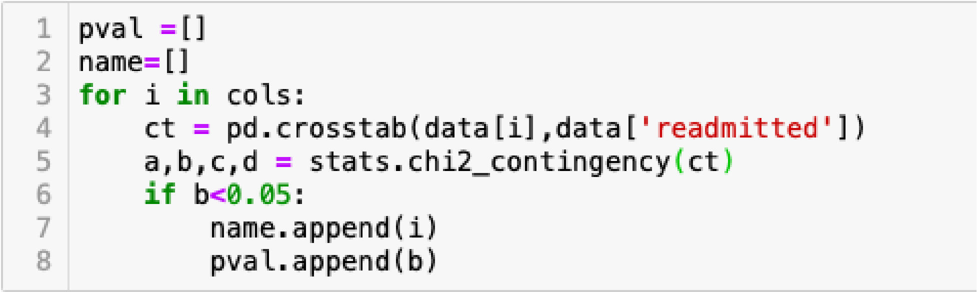
The dataset consists of all categorical features in this case and hence the only statistical test that is used to find the significance of a feature for a categorical target variable (Categorical vs Categorical) is the Chi-Square test.

Chi-square test:

The Chi-Square Test of Independence determines whether there is an association between categorical variables (i.e., whether the variables are independent or related). It is a nonparametric test.This test is also known as Chi-Square Test of Association.

This test utilises a contingency table to analyse the data. A contingency table (also known as a cross-tabulation, cross-tab, or two-way table) is an arrangement in which data is classified according to two categorical variables. The categories for one variable appear in the rows, and the categories for the other variable appear in columns. Each variable must have two or more categories. Each cell reflects the total count of cases for a specific pair of categories.

The test is done on the cleaned data in a loop for all the features and its significance is checked for 95% confidence limit and hence the attributes with P-value less than 0.05 are printed and are found out to be 44 features and they are as follows.



```
pval =[]
name=[]
for i in cols:
    ct = pd.crosstab(data[i],data['readmitted'])
    a,b,c,d = stats.chi2_contingency(ct)
    if b<0.05:
        name.append(i)
        pval.append(b)
```

Features with P-value less than 0.05:

'age', 'time_in_hospital', 'num_lab_procedures', 'num_procedures', 'num_medications', 'number_diagnoses', 'max_glu_serum', 'A1Cresult', 'metformin', 'repaglinide', 'glipizide', 'insulin', 'change', 'diabetesMed', 'readmitted', 'number_of_revisit', 'critical_visits', 'num_med_change', 'admission_source_id_7', 'admission_type_id_3', 'diag_1_5.0', 'diag_1_7.0', 'diag_1_8.0', 'diag_1_9.0', 'diag_1_11.0', 'diag_1_16.0', 'diag_1_17.0', 'diag_2_2.0', 'diag_2_3.0', 'diag_2_11.0', 'diag_2_12.0', 'diag_3_2.0', 'diag_3_3.0', 'diag_3_6.0', 'diag_3_8.0', 'diag_3_11.0', 'diag_3_12.0', 'diag_3_16.0', 'discharge_disposition_id_2', 'discharge_disposition_id_10', 'discharge_disposition_id_13', 'discharge_disposition_id_28', 'race_Caucasian', 'race_Other'

No action is taken on the less significant features at this moment, the feature selection is done based on elimination methods in the later parts using Backward Selection and Variance Inflation Factor(VIF).

2.6 Feature Selection and Data Balancing:

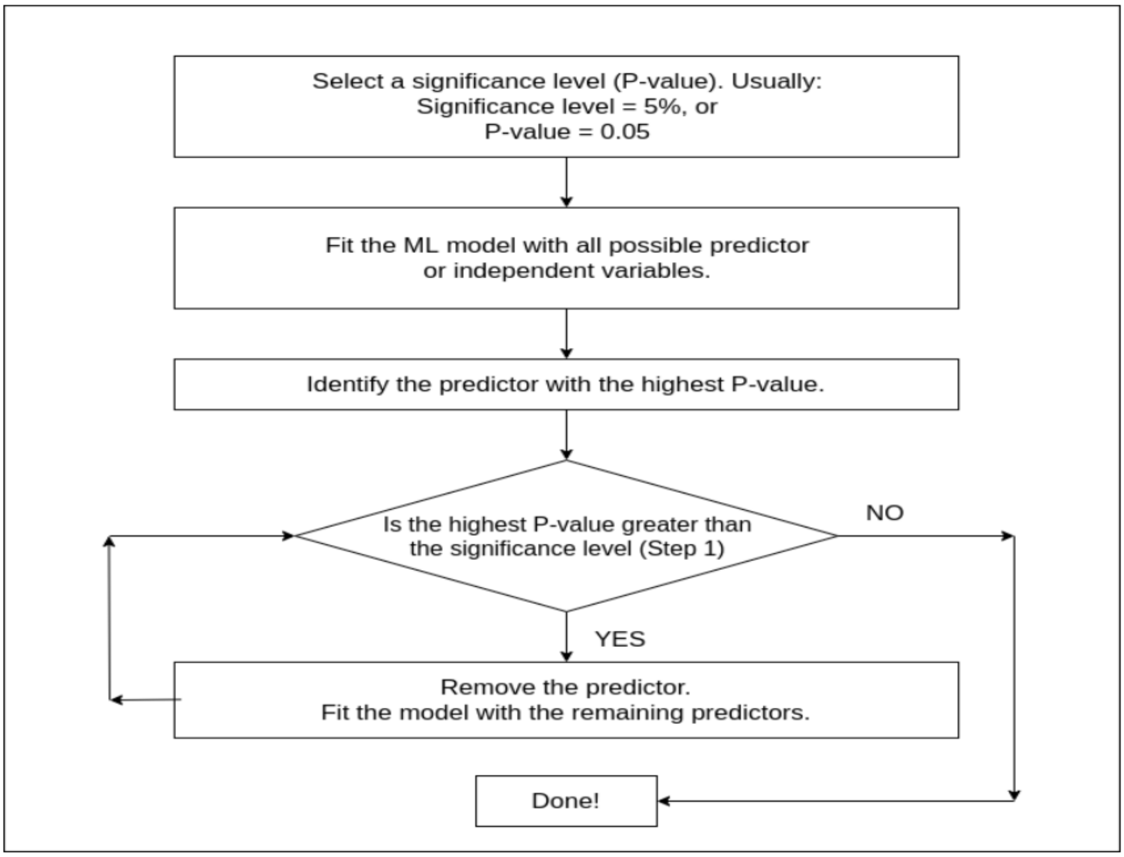
The Feature Selection as mentioned above is done with two methods which are Backward elimination and Variance Inflation Factor (VIF).

Backward Elimination: -

- In backward elimination, it is started with all the features by removing the least significant feature at each iteration which improves the performance of the model. This is repeated until no improvement is observed on the removal of features.

- When a machine learning model is built, it is very important to select only those features or predictors which are necessary Supposing there are 101 features or predictors in the dataset. That doesn't necessarily mean that all 101 features have to be kept in the model. This is because not all 101 features will have significant influence on the model. But then again, this doesn't mean it will be true for all cases. It depends entirely on the data in hand.

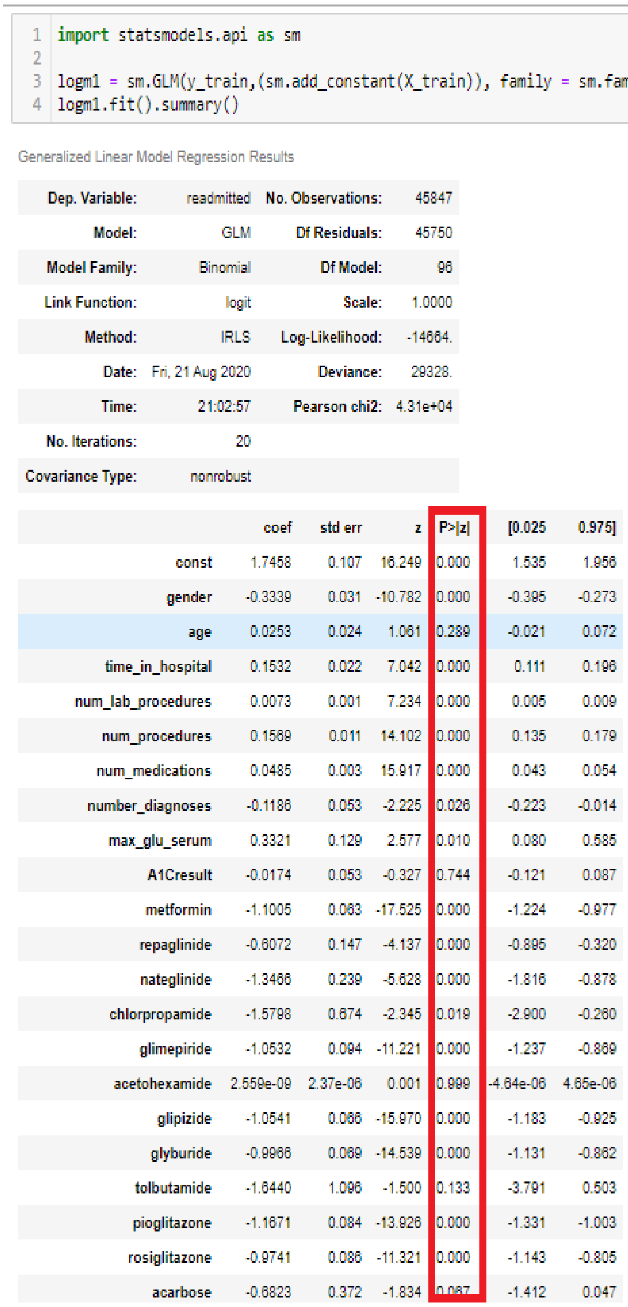
Backward Elimination Process:


Select a significance level (P-value). Usually:
Significance level = 5%, or
P-value = 0.05
Fit the ML model with all possible predictor
or independent variables.
Identify the predictor with the highest P-value.
Is the highest P-value greater than
the significance level (Step 1)
NO
YES
Remove the predictor.
Fit the model with the remaining predictors.
Done!

- There are various ways in which one can find out which features have very little impact on the model and which ones can be removed from the dataset. Backward Elimination does this in a step by step manner as shown in the flowchart above.

- In backward elimination just a significance level is selected, or the P-value. Usually, in most cases, a 5% significance level is selected. This means the P-value will be 0.05. This value can be changed depending on the project.

- ➢ If the P-value of this feature is greater than the significance level, the feature is removed from dataset.
- ➢ Null hypothesis - The feature has no significance in predicting the target variable.
- ➢ Alternate hypothesis - The feature has significance in predicting the target variable
- ➢ One of the most commonly used p-value is 0.05. If the calculated p-value turns out to be less than 0.05, the null hypothesis is considered to be false, or nullified (hence the name null hypothesis). And if the value is greater than 0.05, the null hypothesis is considered to be true.



```python
import statsmodels.api as sm

logm1 = sm.GLM(y_train,(sm.add_constant(X_train)), family = sm.families.Binomial())
logm1.fit().summary()
```

Generalized Linear Model Regression Results

Dep. Variable:	readmitted	No. Observations:	45847
Model:	GLM	Df Residuals:	45750
Model Family:	Binomial	Df Model:	96
Link Function:	logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-14664.
Date:	Fri, 21 Aug 2020	Deviance:	29328.
Time:	21:02:57	Pearson chi2:	4.31e+04
No. Iterations:	20		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	1.7458	0.107	16.249	0.000	1.535	1.956
gender	-0.3339	0.031	-10.782	0.000	-0.395	-0.273
age	0.0253	0.024	1.061	0.289	-0.021	0.072
time_in_hospital	0.1532	0.022	7.042	0.000	0.111	0.196
num_lab_procedures	0.0073	0.001	7.234	0.000	0.005	0.009
num_procedures	0.1569	0.011	14.102	0.000	0.135	0.179
num_medications	0.0485	0.003	15.917	0.000	0.043	0.054
number_diagnoses	-0.1186	0.053	-2.225	0.026	-0.223	-0.014
max_glu_serum	0.3321	0.129	2.577	0.010	0.080	0.585
A1Cresult	-0.0174	0.053	-0.327	0.744	-0.121	0.087
metformin	-1.1005	0.063	-17.525	0.000	-1.224	-0.977
repaglinide	-0.6072	0.147	-4.137	0.000	-0.895	-0.320
nateglinide	-1.3466	0.239	-5.628	0.000	-1.816	-0.878
chlorpropamide	-1.5798	0.674	-2.345	0.019	-2.900	-0.260
glimepiride	-1.0532	0.094	-11.221	0.000	-1.237	-0.869
acetohexamide	2.559e-09	2.37e-06	0.001	0.999	-4.64e-06	4.65e-06
glipizide	-1.0541	0.066	-15.970	0.000	-1.183	-0.925
glyburide	-0.9966	0.069	-14.539	0.000	-1.131	-0.862
tolbutamide	-1.6440	1.096	-1.500	0.133	-3.791	0.503
pioglitazone	-1.1671	0.084	-13.926	0.000	-1.331	-1.003
rosiglitazone	-0.9741	0.086	-11.321	0.000	-1.143	-0.805
acarbose	-0.6823	0.372	-1.834	0.067	-1.412	0.047

- ➢ P-value is the probability that will show the same results as the null hypothesis, and in the example, the threshold for that probability is 0.05. So if the calculated p-value is less than 0.05, it means that there’s very less probability that the same results as the null hypothesis will be got. And if the p-value is more than 0.05, then the probability of getting the same results as null hypothesis is very high, so it can be considered for the null hypothesis to be true.

- ➢ The features eliminated in Backward elimination are as follows,

['age', 'A1Cresult', 'acetohexamide', 'tolbutamide', 'acarbose', 'troglitazone', 'glipizide-metformin', ‘glimepiride-pioglitazone', 'metformin-rosiglitazone', 'metformin-pioglitazone', 'admission_source_id_9', 'admission_source_id_12', 'admission_type_id_5', 'diag_3_14.0', 'discharge_disposition_id_9', 'discharge_disposition_id_18', 'race_Hispanic', 'race_Other']

Variance Inflation Factor(VIF):

The Variance Inflation Factor(VIF) detects multicollinearity in classification analysis. Multicollinearity is when there’s correlation between predictors (i.e. independent variables) in a model; its presence can adversely affect classification results. The VIF estimates how much the variance of a classification coefficient is inflated due to multicollinearity in the model.

$$\mathrm{VIF} = \frac{1}{1 - R_i^2}$$

- ➢ R - R squared Statistic
- ➢ i - the particular feature

- 1 = not correlated.
- Between 1 and 5 = moderately correlated.
- Greater than 5 = highly correlated

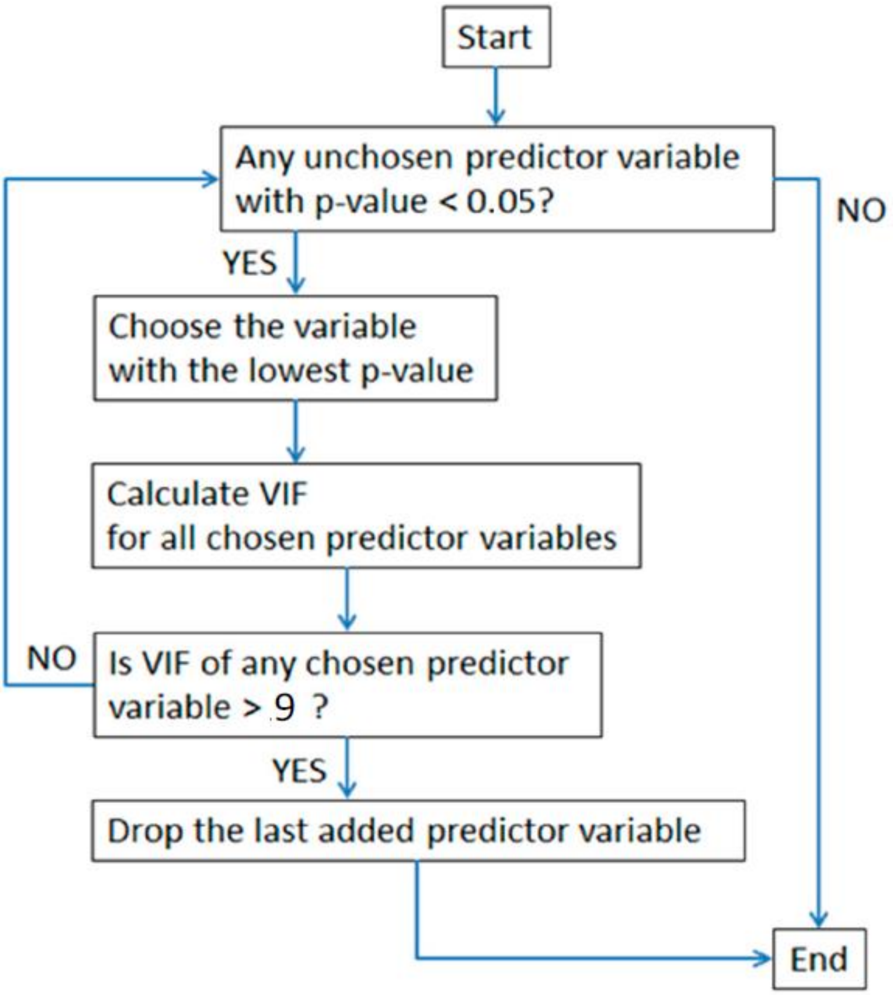
A VIF can be computed for each predictor in a predictive model. A value of 1 means that the predictor is not correlated with other variables. The higher the value, the greater the correlation of the variable with other variables. Values of more than 4 or 5 are sometimes regarded as being moderate to high, with values of 10 or more being regarded as very high. These numbers are just rules of thumb; in some contexts a VIF of 2 could be a great problem. This value varies from case to case.

In this case the VIF cutoff is chosen as 9 and the features having VIF more than 9 are to be removed and the others are kept. The columns that are removed are shown below.

The features eliminated in VIF are as follows,

array(['num_med_change', 'diag_1_7.0', 'diag_2_7.0', 'number_diagnoses', ‘diabetesMed’])

VIF Process:


Start
Any unchosen predictor variable with p-value < 0.05?
NO
YES
Choose the variable with the lowest p-value
Calculate VIF for all chosen predictor variables
NO
Is VIF of any chosen predictor variable > 9 ?
YES
Drop the last added predictor variable
End

2.7 Data Balancing

Since the dataset’s target variable consists of classes which are in the ratio of 92:8 with 92% to class ‘0’ and 8% to class ‘1’, it is highly imbalanced and hence is restricting the models in learning enough of the ‘1s’ .This can be overcome by using resampling techniques such as “SMOTE” for upsampling and “Nearmiss” for downsampling.

In this case a hybrid approach is done where both downsampling and upsampling are done in the order where downsampling is done first, followed by upsampling. The downsampling ratio is given as 15:85 and the upsampling ratio is given as 30:70. Where the downsampling reduces the instances from ‘0s’(the dominant class) and upsampling increases the instances in ‘1s’ (the submissive class).

A humble ratio of 15:85 and 30:70 are used to not be overoptimistic about the metric results to be obtained in the modelling sections. As a ratio of 50:50 on both cases would give way better results but would be overoptimistic in its predictions and may not perform remotely well in a new production data when given.

The dataset target class combinations before and after resampling are shown in the description below,

```
Before hybrid sampling Counter({0: 65222, 1: 6293})
After hybrid sampling Counter({0: 56637, 1: 24273})
```

With this data it can be proceeded to modelling where all the classification models are trained and tested in the data.

3. Classification

3.1 Analysis

Classification is a supervised learning concept which basically categorises a set of data into classes. It can be either binary classification problem or multi-class class classification problem. It specifies the class to which data elements belong to and its best used when the output has finite and discrete values. It predicts a class for an input variable as well.

Various models on classification were performed, they are:

- Logistic Regression
- Random Forest classifier
- Decision tree classifier
- KNN classifier
- Naïve Bayes classifier
- Bagging classifiers
- Boosting classifiers

3.2 Choosing model metrics

In classification model there are several metrics used to evaluate the predictions, some are used here:

- Classification accuracy
- Area Under ROC Curve
- Confusion matrix
- Classification report

Classification Accuracy

This is the most important and common evaluation metric for classification problems. It gives us the number of correct predictions made as ratio of all predictions made.

Area Under ROC Curve

Area under ROC Curve (or AUC for short) is a performance metric for binary classification problems. The AUC represents a model's ability to discriminate between positive and negative classes. ROC can be broken down into sensitivity and specificity. Sensitivity is the true positive rate also called the recall. It is the number of instances from the positive (first) class that actually predicted correctly whereas Specificity is also called the true negative rate. Is the number of instances from the negative (second) class that were actually predicted correctly.

Confusion Matrix

The confusion matrix is a handy presentation of the accuracy of a model with two or more classes. The table presents predictions on the x-axis and accuracy outcomes on the y-axis. The cells of the table are the number of predictions made by a machine learning algorithm.

For example, a machine learning algorithm can predict 0 or 1 and each prediction may actually have been a 0 or 1. Predictions for 0 that were actually 0 appear in the cell for prediction = 0 and actual = 0, whereas predictions for 0 that were actually 1 appear in the cell for prediction = 0 and actual = 1. And so on.

Classification Report

The scikit-learn library provides a convenience report when working on classification problems to give you a quick idea of the accuracy of a model using a number of measures. The classification report() function displays the precision, recall, F1-score and support for each class.

3.3 Optimum Cut-off

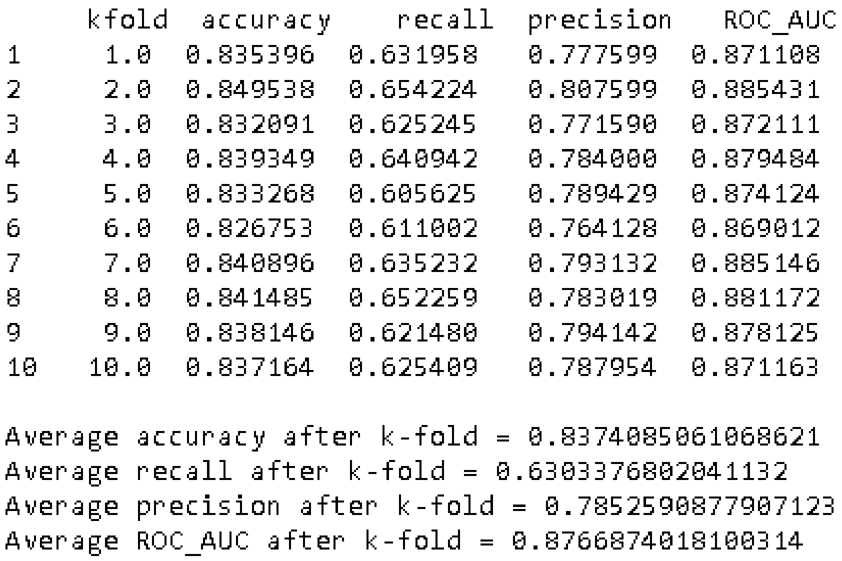
Optimum cut-off point is a point where the standard deviation of accuracy, precision and recall is minimum. When the standard deviation among accuracy, recall and precision is minimum, the maximum possible value of all the three without a tradeoff among each other is attained.

3.4 Spot Check

Spot-checking is a way of discovering which algorithms perform well on your machine learning problem. It cannot be known which algorithms are best suited to your problem beforehand. Several methods should be tried and focused on those that prove themselves the most promising. Six Classification Algorithms can be checked and verified for their metrics in the dataset. Starting with linear machine learning algorithm, Logistic Regression.

Logistic Regression:

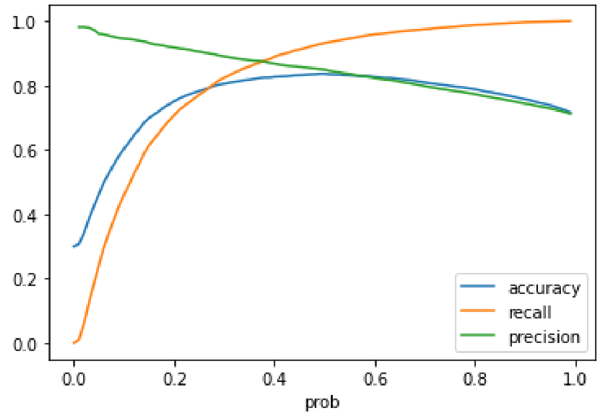
Logistic Regression assumes a Gaussian distribution for the numeric input variables and can model binary classification problems.



```
     kfold  accuracy    recall  precision   ROC_AUC
1      1.0  0.835396  0.631958   0.777599  0.871108
2      2.0  0.849538  0.654224   0.807599  0.885431
3      3.0  0.832091  0.625245   0.771590  0.872111
4      4.0  0.839349  0.640942   0.784000  0.879484
5      5.0  0.833268  0.605625   0.789429  0.874124
6      6.0  0.826753  0.611002   0.764128  0.869012
7      7.0  0.840896  0.635232   0.793132  0.885146
8      8.0  0.841485  0.652259   0.783019  0.881172
9      9.0  0.838146  0.621480   0.794142  0.878125
10    10.0  0.837164  0.625409   0.787954  0.871163

Average accuracy after k-fold = 0.8374085061068621
Average recall after k-fold = 0.6303376802041132
Average precision after k-fold = 0.7852590877907123
Average ROC_AUC after k-fold = 0.8766874018100314
```

Finding Optimal Cutoff Point based on accuracy, recall, precision

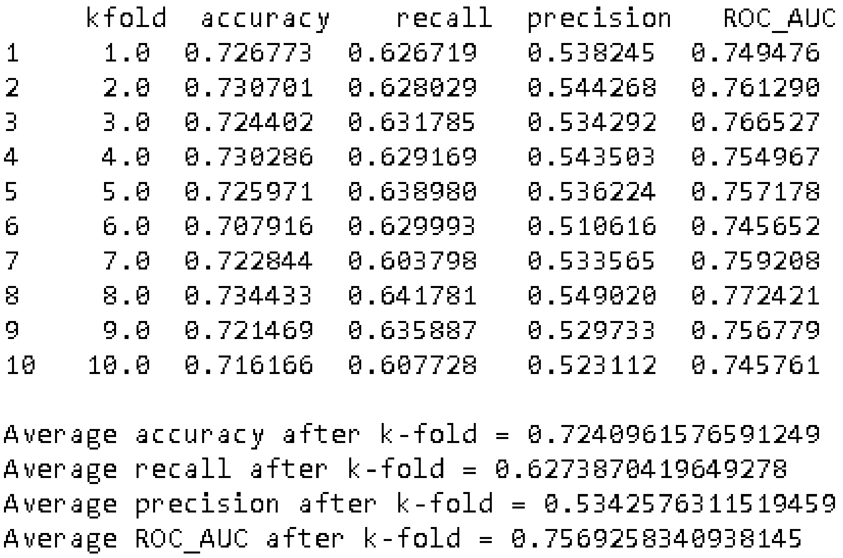


```
Optimum cutoff probability is 0.38
Accuracy is 0.82
Precision is 0.71
Recall is 0.70
ROC_AUC Score is 0.79
```

Then looking at some non-linear machine learning algorithms, k-Nearest Neighbors, Naïve Bayes, Decision Trees, Random Forest. Along with ensemble techniques like Boosting and Bagging models for those.

k-Nearest Neighbors:

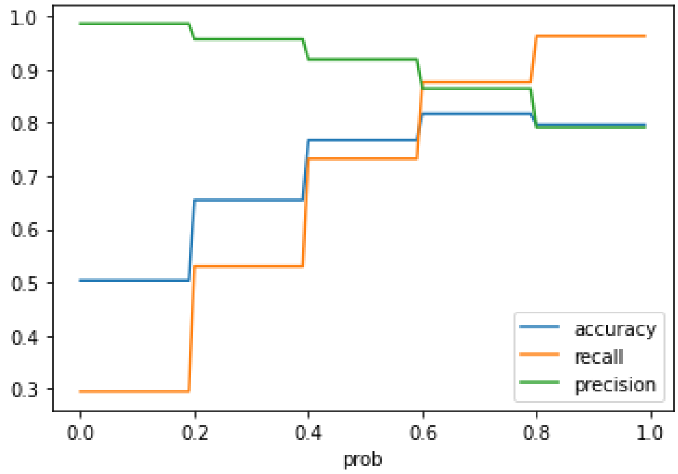
The *k*-Nearest Neighbors Classifier algorithm (or KNN) uses a distance metric to find the *k* most similar instances in the training data for a new instance and takes the dominant outcome of the neighbors as the prediction.



```
     kfold  accuracy    recall  precision   ROC_AUC
1      1.0  0.726773  0.626719   0.538245  0.749476
2      2.0  0.730701  0.628029   0.544268  0.761290
3      3.0  0.724402  0.631785   0.534292  0.766527
4      4.0  0.730286  0.629169   0.543503  0.754967
5      5.0  0.725971  0.638980   0.536224  0.757178
6      6.0  0.707916  0.629993   0.510616  0.745652
7      7.0  0.722844  0.603798   0.533565  0.759208
8      8.0  0.734433  0.641781   0.549020  0.772421
9      9.0  0.721469  0.635887   0.529733  0.756779
10    10.0  0.716166  0.607728   0.523112  0.745761

Average accuracy after k-fold = 0.7240961576591249
Average recall after k-fold = 0.6273870419649278
Average precision after k-fold = 0.5342576311519459
Average ROC_AUC after k-fold = 0.7569258340938145
```

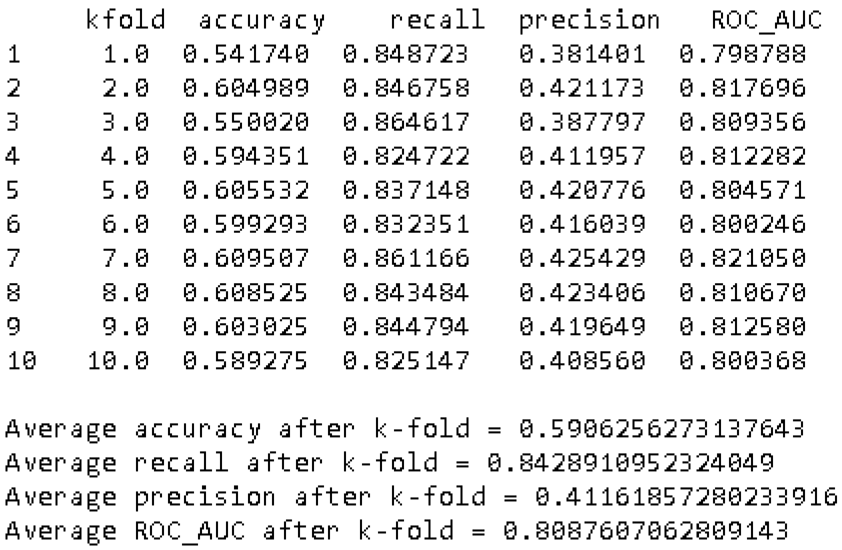
```
Finding Optimal Cutoff Point based on accuracy, recall, precision
```



```
Optimum cutoff probability is 0.60
Accuracy is 0.82
Precision is 0.70
Recall is 0.68
ROC_AUC Score is 0.78
```

Naive Bayes:

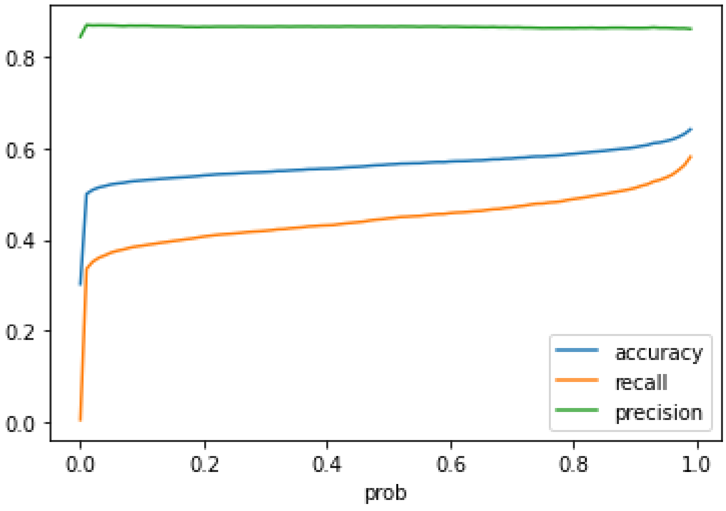
Naive Bayes calculates the probability of each class and the conditional probability of each class given each input value. These probabilities are estimated for new data and multiplied together, assuming that they are all independent.

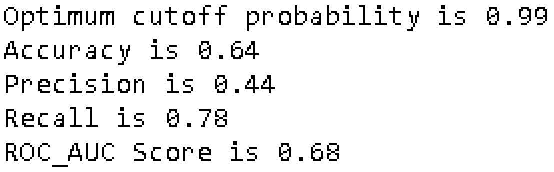


```
     kfold  accuracy    recall  precision   ROC_AUC
1      1.0  0.541740  0.848723   0.381401  0.798788
2      2.0  0.604989  0.846758   0.421173  0.817696
3      3.0  0.550020  0.864617   0.387797  0.809356
4      4.0  0.594351  0.824722   0.411957  0.812282
5      5.0  0.605532  0.837148   0.420776  0.804571
6      6.0  0.599293  0.832351   0.416039  0.800246
7      7.0  0.609507  0.861166   0.425429  0.821050
8      8.0  0.608525  0.843484   0.423406  0.810670
9      9.0  0.603025  0.844794   0.419649  0.812580
10    10.0  0.589275  0.825147   0.408560  0.800368

Average accuracy after k-fold = 0.5906256273137643
Average recall after k-fold = 0.8428910952324049
Average precision after k-fold = 0.41161857280233916
Average ROC_AUC after k-fold = 0.8087607062809143
```

```
Finding Optimal Cutoff Point based on accuracy, recall, precision
```



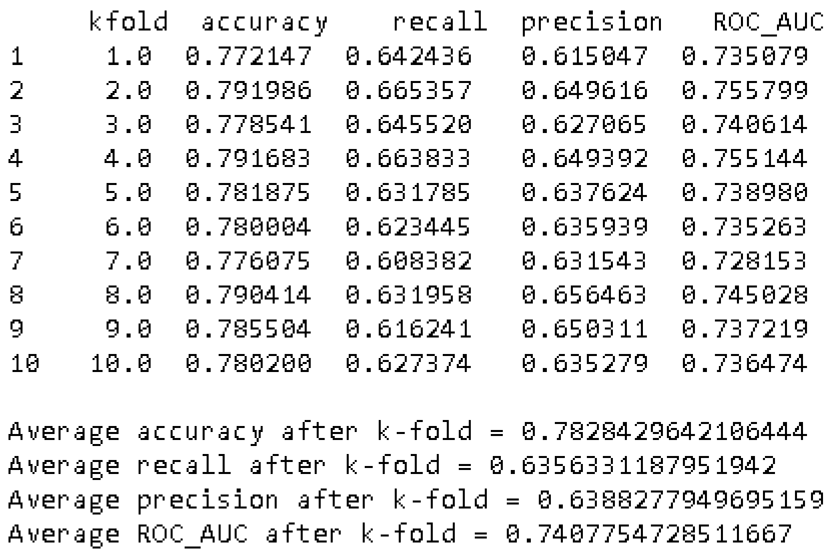


```
Optimum cutoff probability is 0.99
Accuracy is 0.64
Precision is 0.44
Recall is 0.78
ROC_AUC Score is 0.68
```

Since the optimum cutoff probability is 0.99 which is too biased and hence it is not recommended to consider this cutoff probability.

Decision tree:

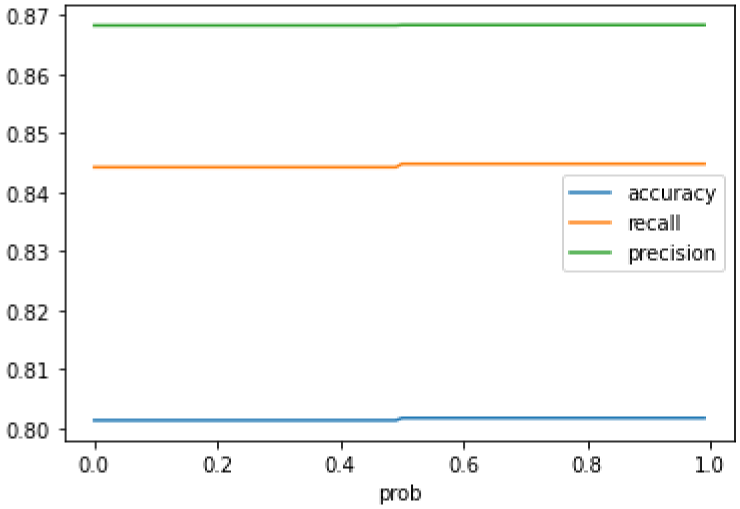
Classification Trees (Decision trees/Random Forest) construct a binary tree from the training data. Split points are chosen greedily by evaluating each attribute and each value of each attribute in the training data in order to minimize a cost function (like the Gini /Entropy index).



```
      kfold  accuracy    recall  precision   ROC_AUC
1       1.0  0.772147  0.642436   0.615047  0.735079
2       2.0  0.791986  0.665357   0.649616  0.755799
3       3.0  0.778541  0.645520   0.627065  0.740614
4       4.0  0.791683  0.663833   0.649392  0.755144
5       5.0  0.781875  0.631785   0.637624  0.738980
6       6.0  0.780004  0.623445   0.635939  0.735263
7       7.0  0.776075  0.608382   0.631543  0.728153
8       8.0  0.790414  0.631958   0.656463  0.745028
9       9.0  0.785504  0.616241   0.650311  0.737219
10     10.0  0.780200  0.627374   0.635279  0.736474

Average accuracy after k-fold = 0.7828429642106444
Average recall after k-fold = 0.6356331187951942
Average precision after k-fold = 0.6388277949695159
Average ROC_AUC after k-fold = 0.7407754728511667
```

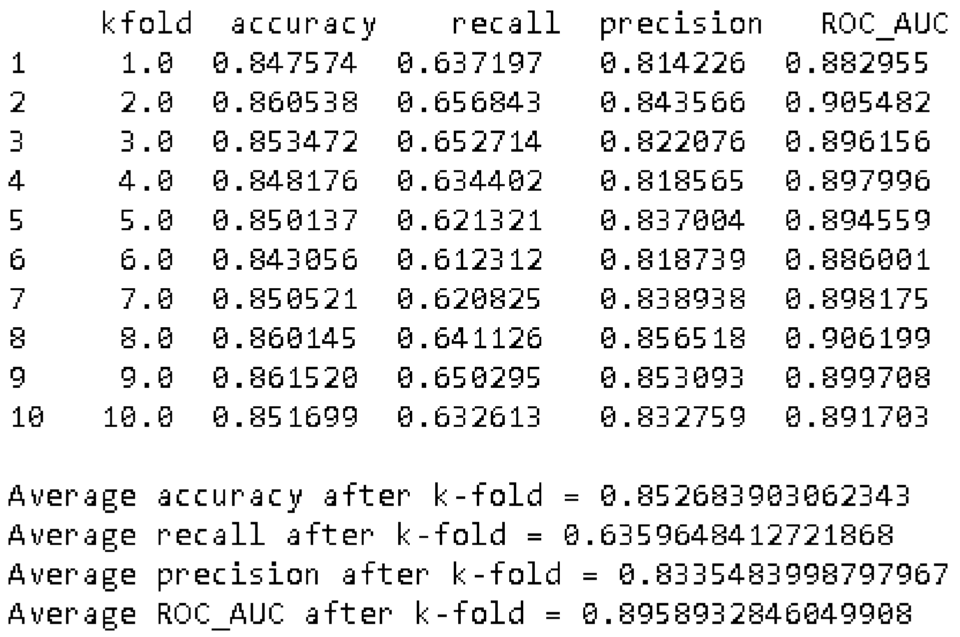
```
Finding Optimal Cutoff Point based on accuracy, recall, precision
```



```
Optimum cutoff probability is 0.50
Accuracy is 0.80
Precision is 0.66
Recall is 0.70
ROC_AUC Score is 0.77
```

Random Forest:

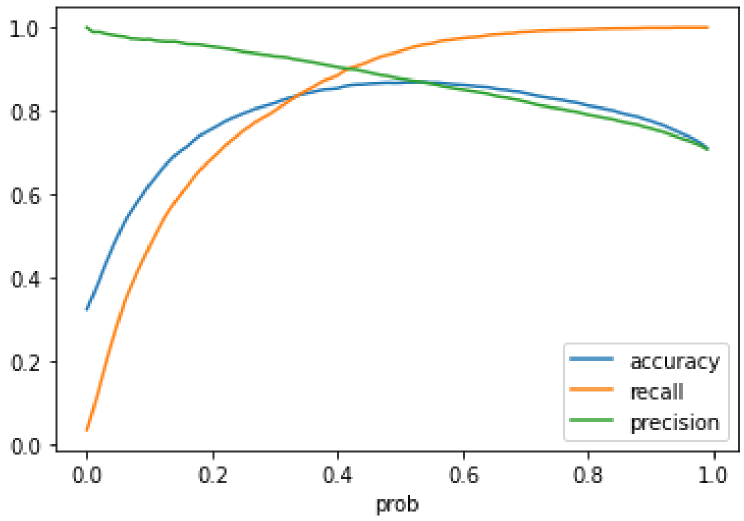
Random Forest classifier that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees.

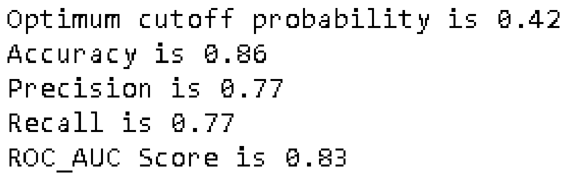


```
     kfold  accuracy    recall  precision   ROC_AUC
1      1.0  0.847574  0.637197   0.814226  0.882955
2      2.0  0.860538  0.656843   0.843566  0.905482
3      3.0  0.853472  0.652714   0.822076  0.896156
4      4.0  0.848176  0.634402   0.818565  0.897996
5      5.0  0.850137  0.621321   0.837004  0.894559
6      6.0  0.843056  0.612312   0.818739  0.886001
7      7.0  0.850521  0.620825   0.838938  0.898175
8      8.0  0.860145  0.641126   0.856518  0.906199
9      9.0  0.861520  0.650295   0.853093  0.899708
10    10.0  0.851699  0.632613   0.832759  0.891703

Average accuracy after k-fold = 0.852683903062343
Average recall after k-fold = 0.6359648412721868
Average precision after k-fold = 0.8335483998797967
Average ROC_AUC after k-fold = 0.8958932846049908
```

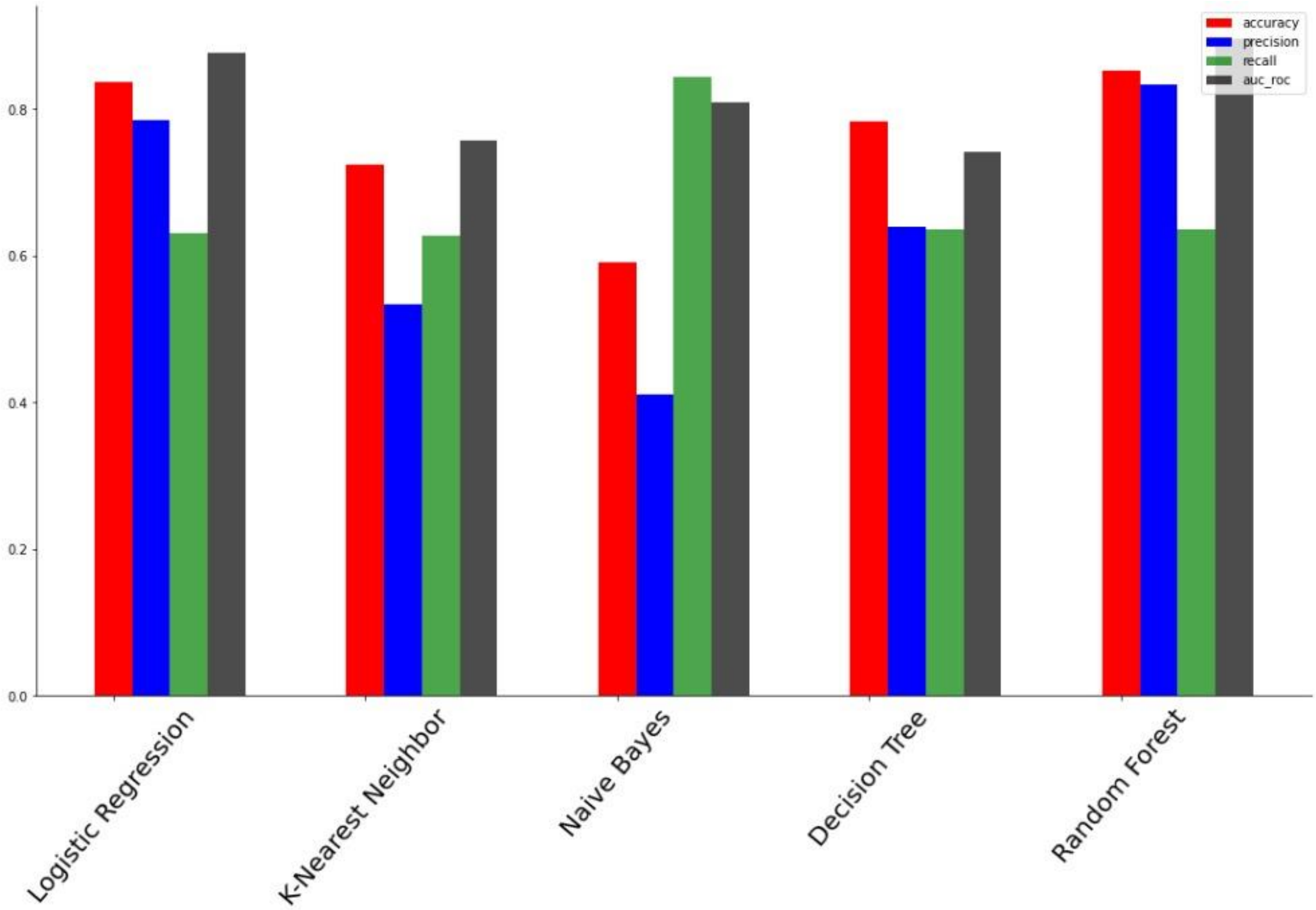
```
Finding Optimal Cutoff Point based on accuracy, recall, precision
```





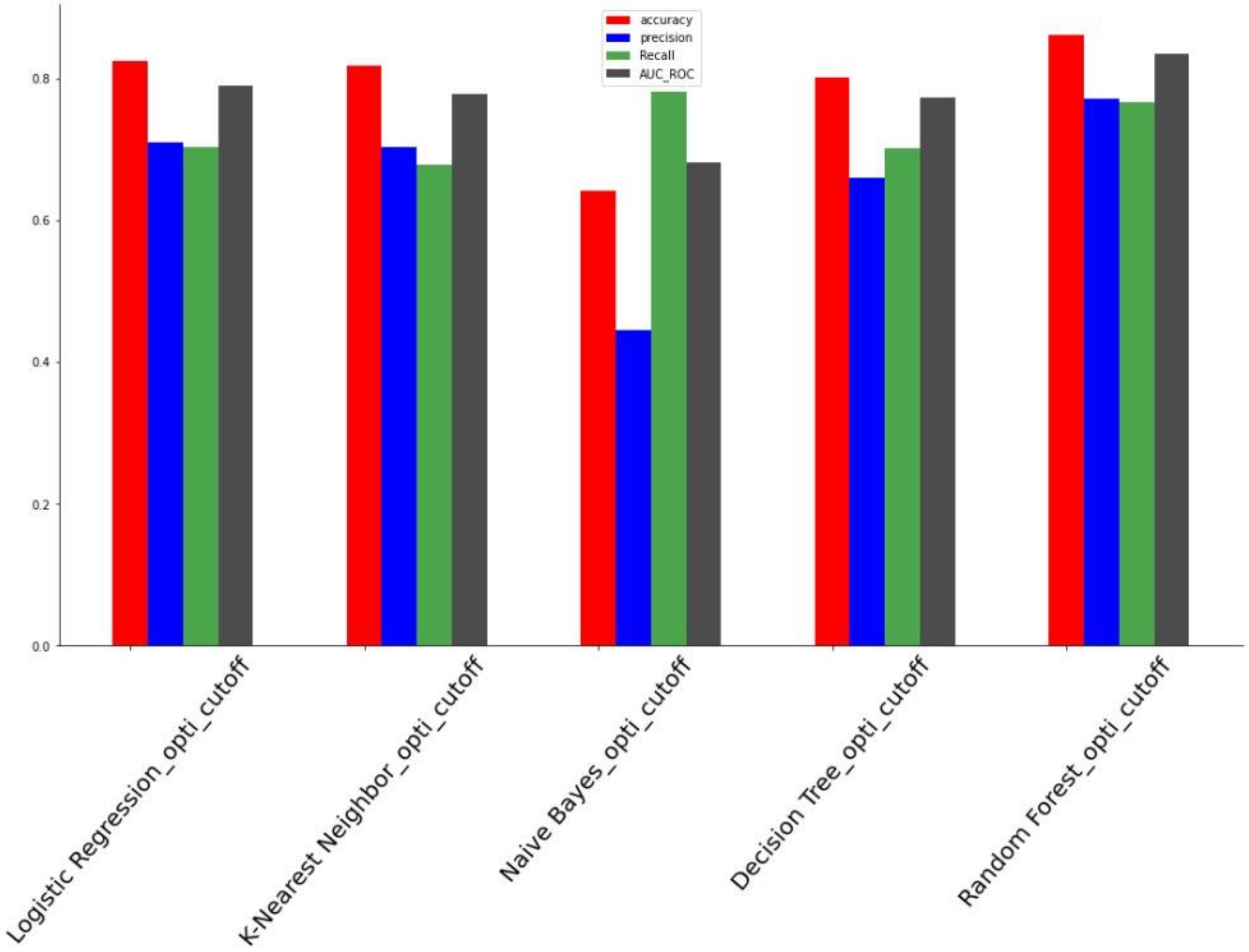
```
Optimum cutoff probability is 0.42
Accuracy is 0.86
Precision is 0.77
Recall is 0.77
ROC_AUC Score is 0.83
```

3.5 Base Model Comparison



The above graph represents the Accuracy, Precision, Recall and AUC ROC Curve for all the base models in a bar graph.

Base Model Comparison with optimum cut-off



From the above graph the Accuracy, Recall, Precision and AUC ROC Curve for optimum cut-off for all the base models have been plotted using bar graph.

3.6 Model Parameter Tuning

Why hyperparameter tuning?

Hyperparameters are different from parameters, which are the internal coefficients or weights for a model found by the learning algorithm. Unlike parameters, hyperparameters are specified by the practitioner when configuring the model. Typically, it is challenging to know what values to use for the hyperparameters of a given algorithm on a given dataset, therefore it is common to use random or grid search strategies for different hyperparameter values.

Why Random Search CV?

Since random search is a technique where random combinations of the hyperparameters are used to find the best solution for the built model. Though grid search cv has higher chances of finding the optimal parameter, the chances are comparatively similar in random search because of the random search pattern where the model might end up being trained on the optimized parameters without any aliasing. It is similar to grid search, and yet it has proven to yield good results in a comparatively shorter run time.

Finding the best params for all the base models.

Logistic Regression:

Logistic regression uses random search with Fitting of 10 folds for each of 6 candidates, total ling 60 fits for it to predic t accuracy, recall, precision and ROC_AUC Score.

The best params is {‘C’:2}

C *as default is 1.0*, Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.

K – Nearest Neighbors:

KNN uses Fitting 10 folds for each of 400 candidates, totaling 4000 fits for it to predict the accuracy, recall, precision and ROC_AUC Score.

The best params are {'weights': 'distance', 'n_neighbors': 5}

Weights : ‘*distance’* weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away. n_neighbors: *int*, *default=5*, Number of neighbors to use by default for kneighbors queries.

Naïve Bayes

There is no need for hyperparameter tuning because the default parameters are already well optimized for modeling. The ensemble techniques such as bagging, and boosting have been done.

Decision Tree:

Decision Tree has fitting 10 folds for each of 1000 candidates, totalling 10000 fits for it to predict the accuracy, recall, precision and ROC_AUC Score.

best_parameters are {'min_samples_split': 3, 'min_samples_leaf': 1, 'max_depth': 35, 'criterion': 'entropy'}

min_samples_split: *int or float, default=2*, The minimum number of samples required to split an internal node:If int, then consider min_samples_split as the minimum number.

min_samples_leaf: *int or float*, *default=1*, The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches.

max_depth: *int*, *default=None*, The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Criterion: *{“gini”, “entropy”}, default=”gini”*, The function to measure the quality of a split. This case “entropy” for the information gain.

Random Forest:

RF uses fitting 10 folds for each of 1000 candidates, totalling 10000 fits fits for it to predict the accuracy, recall, precision and ROC_AUC Score.

best_parameters {'min_samples_split': 35, 'min_samples_leaf': 14, 'max_leaf_nodes': 15, 'max_depth': 35, 'criterion': 'entropy'}

Similar to Decision tree,

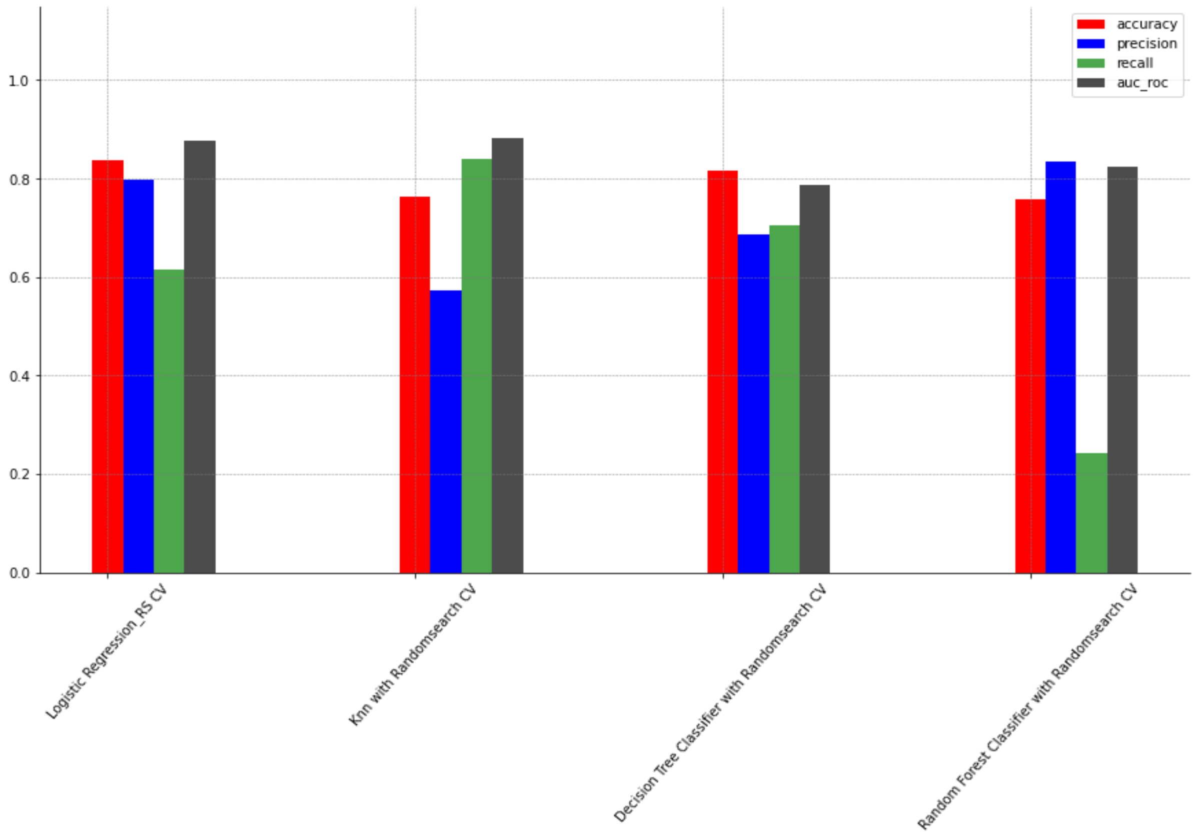
min_samples_split: *int or float, default=2*, The minimum number of samples required to split an internal node:If int, then consider min_samples_split as the minimum number.

min_samples_leaf: *int or float, default=1*, The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches.

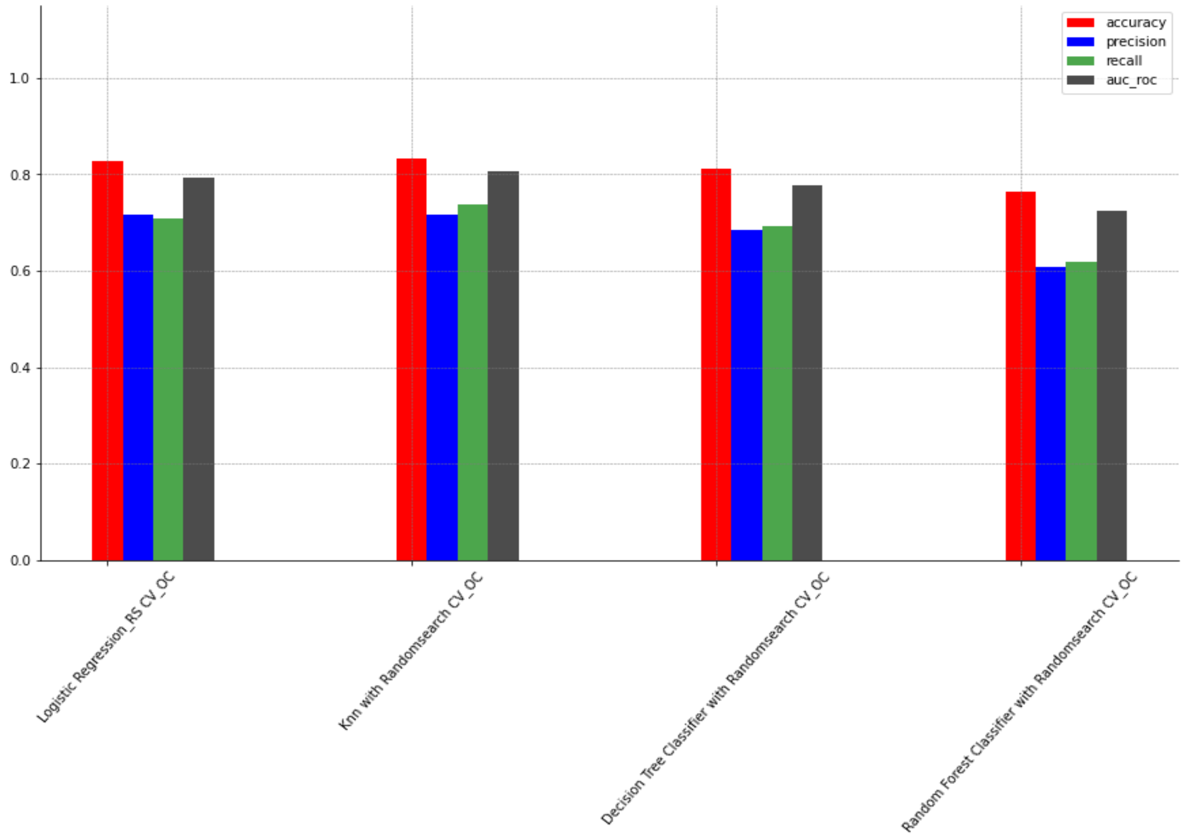
max_depth: *int, default=None*, The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Criterion: *{"gini", "entropy"}, default="gini"*, The function to measure the quality of a split. This case "entropy" for the information gain.

Below graphs shows the comparison of all model using randomized search cv with respective of their accuracy, precision, recall and auc_roc score.



Since in health care sector “Recall” is a priority, Precision has been traded-off ,which is shown clearly below using a graph,



3.7 ENSEMBLING TECHNIQUES

The selected model from the above procedures are then applied ensemble techniques. These techniques include Bagging and Boosting which focuses on the variance error and the bias error respectively. And then a tradeoff must be done by looking at the bias and variance error results of the models. After a reasonable tradeoff, the final results reflect the optimum metrics for bagged and boosted models and their comparison is also visualized.

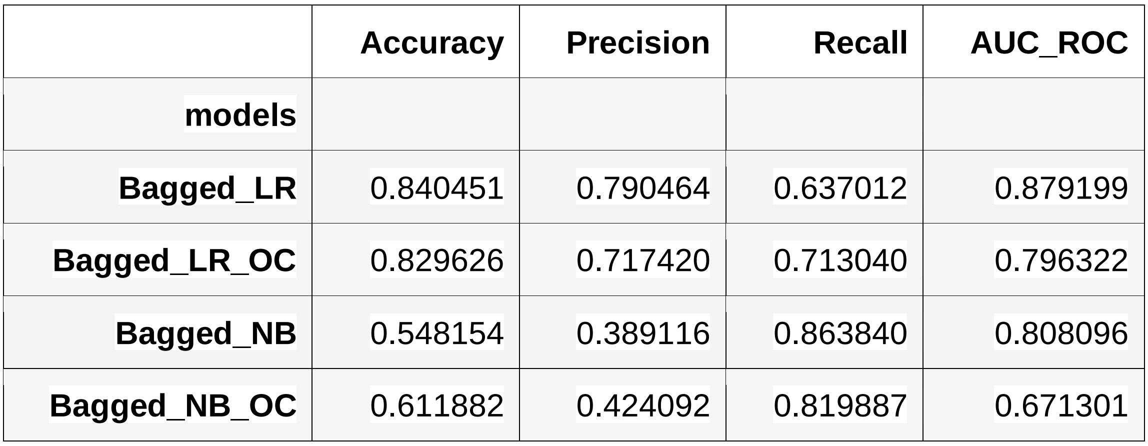
Bagging:

Bagging is used when the goal is to reduce the variance of a decision tree classifier. Here the objective is to create several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their models. As a result, we get an ensemble of different models. Average of all the predictions from different models are used which is more robust than a single model classifier.

Models for which Bagging is done and their results:

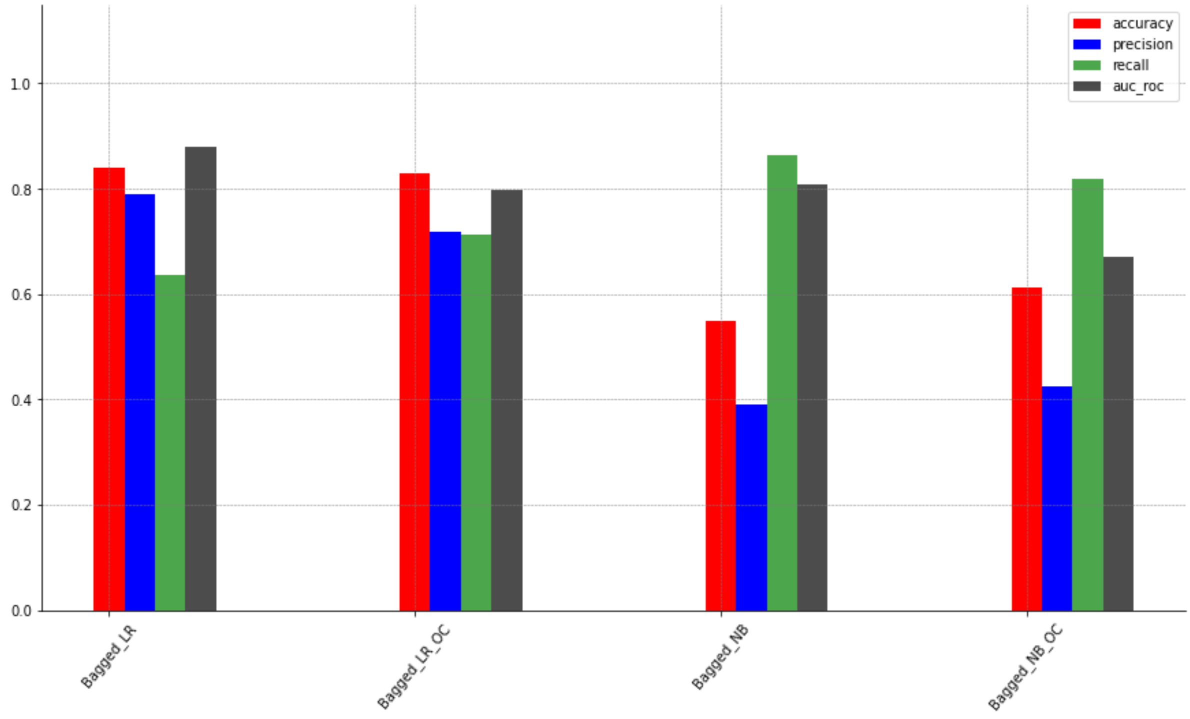
The models like Logistic Regression, Naive Bayes are Bagged and the other three models like K-Nearest Neighbours, Decision Tree and Random Forest are not bagged as it is meaningless to perform bagging to K-Nearest Neighbours due to 63% sampling issue, Decision tree is not bagged as a bagged decision tree would always perform worse than a Random forest which is much more efficient. Random Forest is not bagged since it already involves a bagging mechanism for decision trees.

Bagged Models Comparison:



	Accuracy	Precision	Recall	AUC_ROC
models				
Bagged_LR	0.840451	0.790464	0.637012	0.879199
Bagged_LR_OC	0.829626	0.717420	0.713040	0.796322
Bagged_NB	0.548154	0.389116	0.863840	0.808096
Bagged_NB_OC	0.611882	0.424092	0.819887	0.671301

The results of Bagging in Logistic Regression and Naive Bayes with and without Optimum cutoff are shown in the table below:



Boosting:

Boosting is used to create a collection of predictors. In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analysing data for errors. Consecutive models (random sample) are fit and at every step, the goal is to improve the accuracy from the prior model. When an input is misclassified by a hypothesis, its weight is increased so that next hypothesis is more likely to classify it correctly. This process converts weak learners into better performing model.

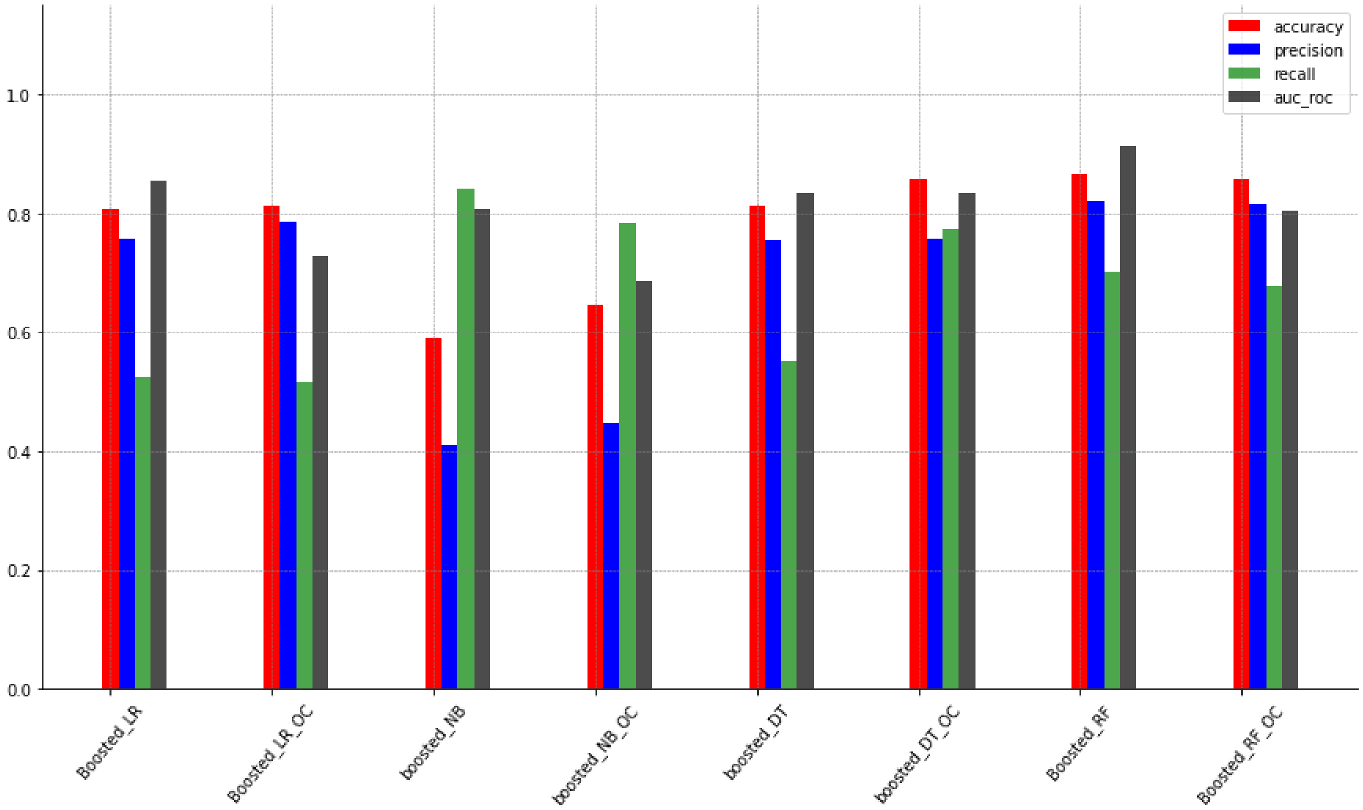
Models for which Boosting is done and their results:

All the models like Logistic Regression, Naive Bayes, Decision Tree and Random Forest are boosted and k-Nearest Neighbors model alone is not bagged as it starts training only when the test record arrives and hence it cannot be boosted at all. The boosted models are then compared and the best among those is selected based on appropriate metrics.

The results of boosting with and without Optimum cutoff in the models are shown in the table below:

Boosted models comparison:

Models	Accuracy	Precision	Recall	AUC_ROC
Boosted_LR	0.806719	0.757048	0.523567	0.855315
Boosted_LR_OC	0.812614	0.785288	0.516790	0.728109
boosted_NB	0.590291	0.411326	0.843021	0.808558
boosted_NB_OC	0.645512	0.448104	0.783253	0.684859
boosted_DT	0.811863	0.754999	0.552240	0.832885
boosted_DT_OC	0.857891	0.758458	0.772351	0.833455
Boosted_RF	0.864387	0.819823	0.702409	0.912479
Boosted_RF_OC	0.857105	0.814563	0.678151	0.805985



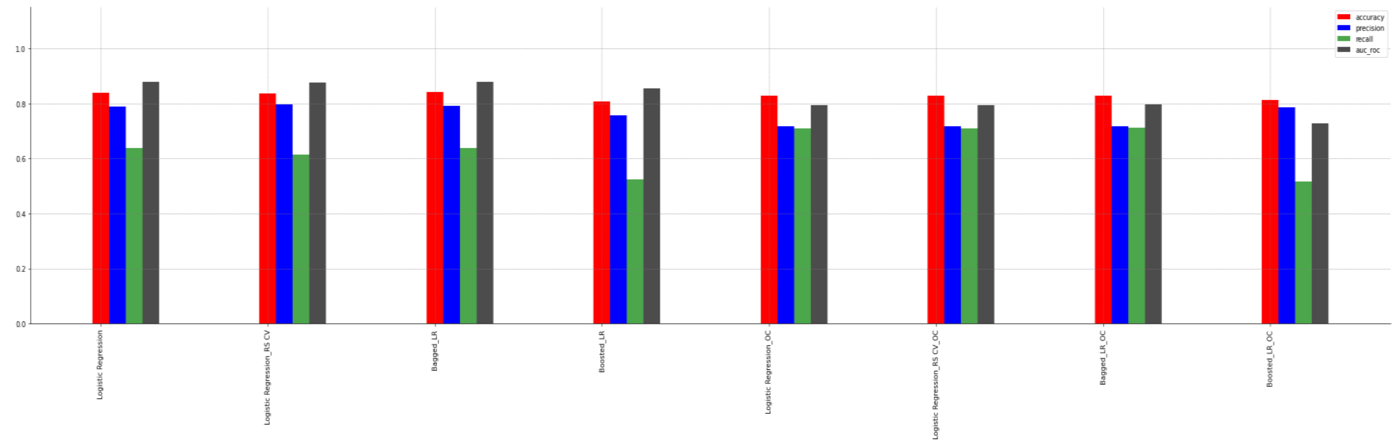
3.8 FINAL COMPARISON

All the models have displayed different metrics and hence the optimum models out of all those are selected and compared here. The optimum models are selected in each algorithm as per the appropriate metrics that are on focus. These models are compared and visualized in charts for better understanding of their performance.

Performance Comparison for Logistic Regression:

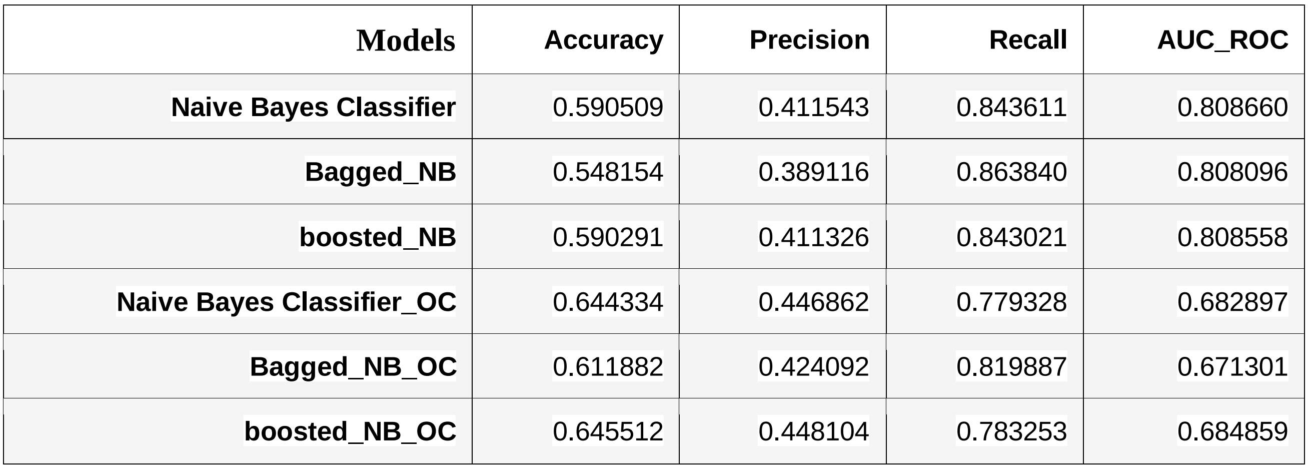
In this section the several models built with Logistic Regression algorithm are compared and visualised. So that the best model out of those can be selected by giving proper weightage to appropriate metrics.

Models	Accuracy	Precision	Recall	AUC_ROC
Logistic Regression	0.840137	0.789227	0.637340	0.878732
Logistic Regression_RS CV	0.837739	0.797626	0.615351	0.877188
Bagged_LR	0.840451	0.790464	0.637012	0.879199
Boosted_LR	0.806719	0.757048	0.523567	0.855315
Logistic Regression_OC	0.828448	0.716490	0.708679	0.794235
Logistic Regression_RS CV_OC	0.828448	0.716490	0.708679	0.794235
Bagged_LR_OC	0.829626	0.717420	0.713040	0.796322
Boosted_LR_OC	0.812614	0.785288	0.516790	0.728109

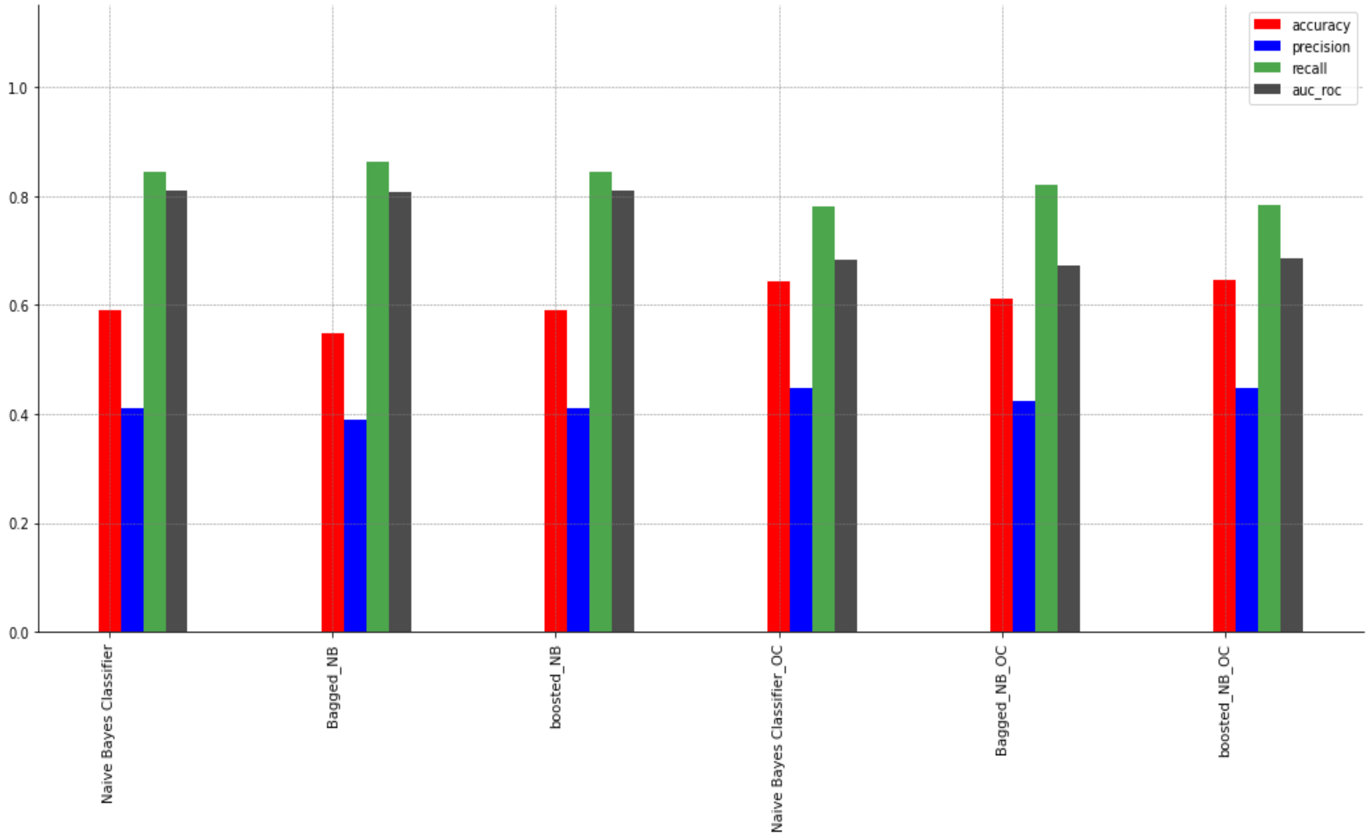


Performance Comparison for Naive Bayes:

In this section the several models built with Naive Bayes algorithm are compared and visualised. So that the best model out of those can be selected by giving proper weightage to appropriate metrics.



Models	Accuracy	Precision	Recall	AUC_ROC
Naive Bayes Classifier	0.590509	0.411543	0.843611	0.808660
Bagged_NB	0.548154	0.389116	0.863840	0.808096
boosted_NB	0.590291	0.411326	0.843021	0.808558
Naive Bayes Classifier_OC	0.644334	0.446862	0.779328	0.682897
Bagged_NB_OC	0.611882	0.424092	0.819887	0.671301
boosted_NB_OC	0.645512	0.448104	0.783253	0.684859

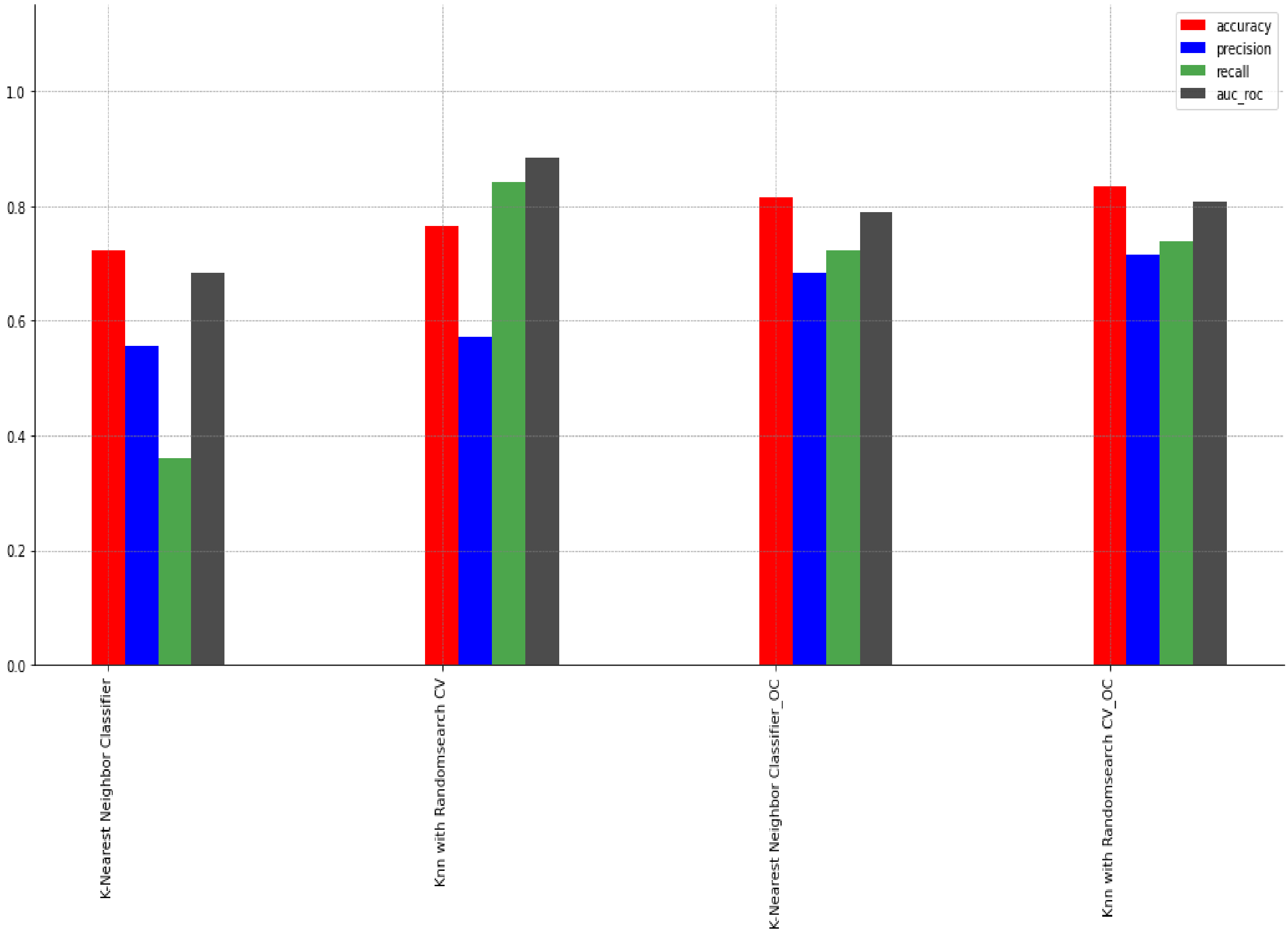


Boosted NB looks reasonable.

Performance Comparison for k-Nearest Neighbours:

In this section the several models built with k-Nearest Neighbours algorithm are compared and visualised. So that the best model out of those can be selected by giving proper weightage to appropriate metrics.

Models	Accuracy	Precision	Recall	AUC_ROC
K-Nearest Neighbor Classifier	0.721761	0.555890	0.359718	0.682774
Knn with Randomsearch CV	0.764256	0.573021	0.840693	0.882952
K-Nearest Neighbor Classifier_OC	0.816409	0.683128	0.723942	0.789995
Knn with Randomsearch CV_OC	0.833252	0.715146	0.738306	0.806127

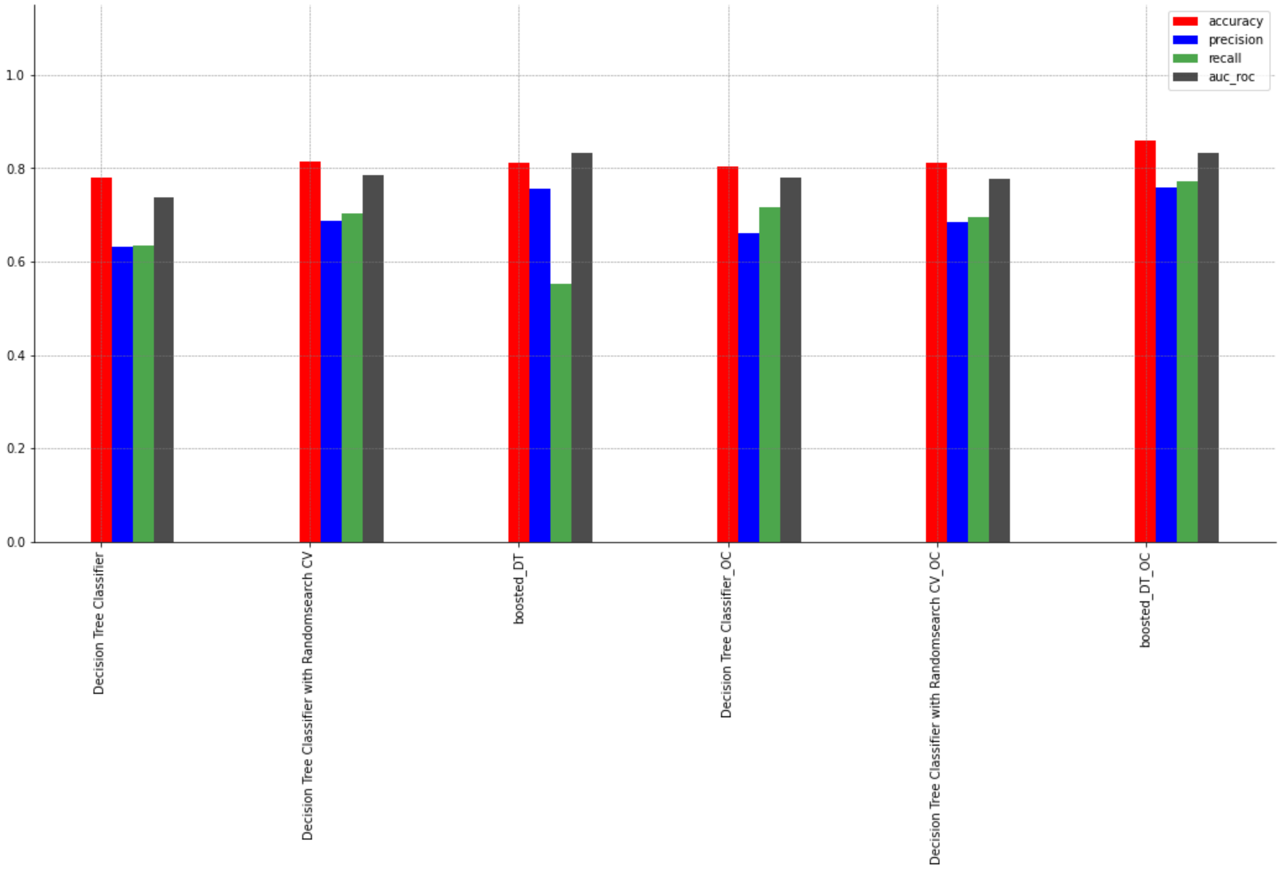


Knn with Random search CV OC Looks reasonable

Performance Comparison for Decision Tree:

In this section the several models built with Decision Tree algorithm are compared and visualized. So that the best model out of those can be selected by giving proper weightage to appropriate metrics.

Models	Accuracy	Precision	Recall	AUC_ROC
Decision Tree Classifier	0.779131	0.631316	0.632951	0.737395
Decision Tree Classifier with Randomsearch CV	0.814997	0.687101	0.703958	0.785702
boosted_DT	0.811863	0.754999	0.552240	0.832885
Decision Tree Classifier_OC	0.804501	0.660635	0.716529	0.779371
Decision Tree Classifier with Randomsearch CV_OC	0.811954	0.683973	0.693817	0.778204
boosted_DT_OC	0.857891	0.758458	0.772351	0.833455

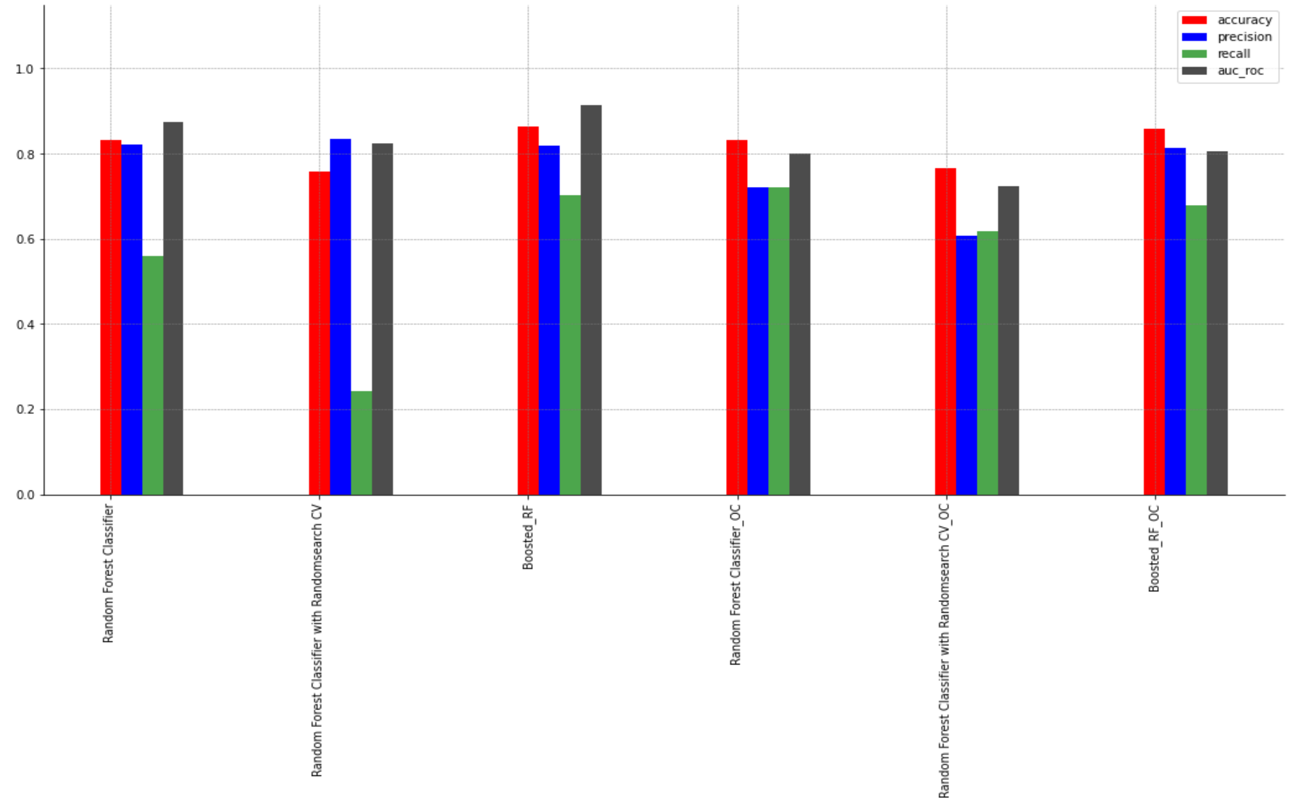


Boosted DT OC looks reasonable.

Performance Comparison for Random Forest:

In this section the several models built with Random Forest algorithm are compared and visualised. So that the best model out of those can be selected by giving proper weightage to appropriate metrics.

Models	Accuracy	Precision	Recall	AUC_ROC
Random Forest Classifier	0.831753	0.822453	0.559963	0.873447
Random Forest Classifier with Randomsearch CV	0.758072	0.835601	0.241086	0.823796
Boosted_RF	0.864387	0.819823	0.702409	0.912479
Random Forest Classifier_OC	0.832243	0.720454	0.720454	0.800309
Random Forest Classifier with Randomsearch CV_OC	0.765139	0.606684	0.617599	0.722989
Boosted_RF_OC	0.857105	0.814563	0.678151	0.805985



Boosted RF seems reasonable.

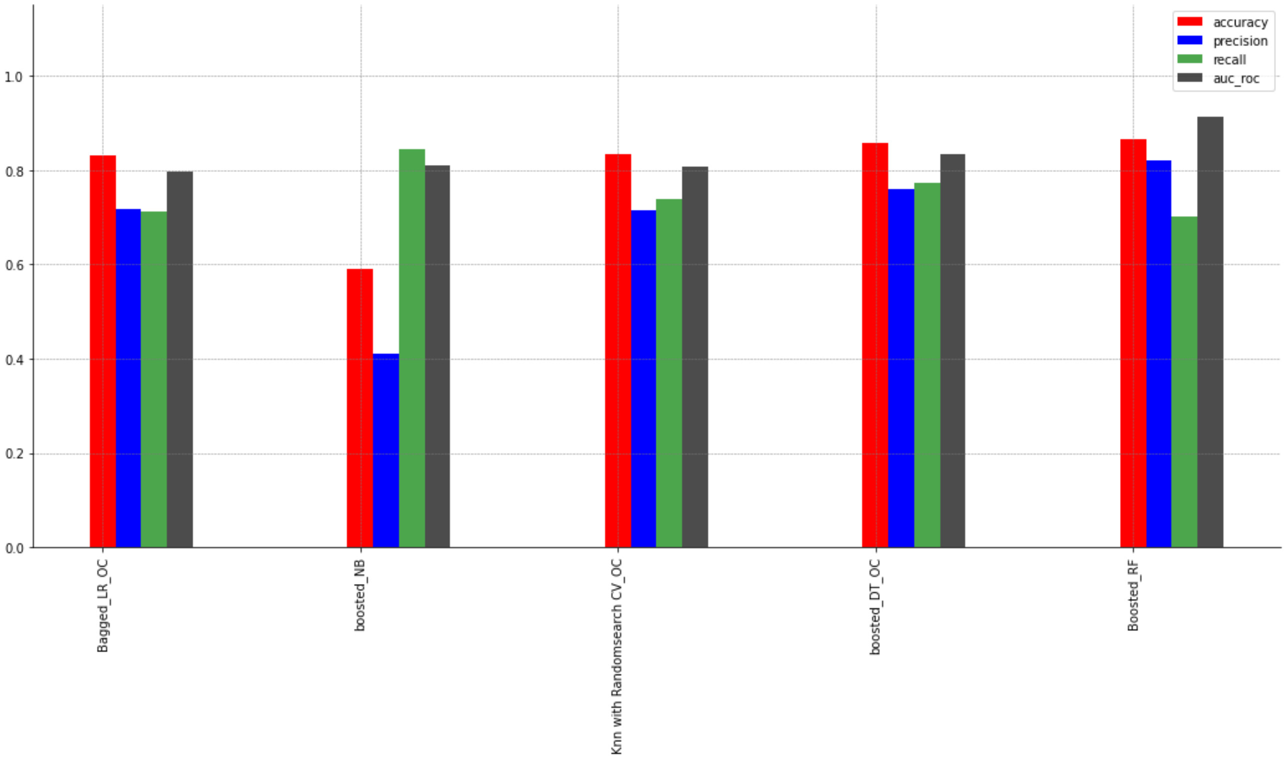
3.9 CHOOSING THE BEST MODEL

All the selected models from each algorithm are compared here so that the best among those can be selected by giving proper weightage to appropriate metrics.

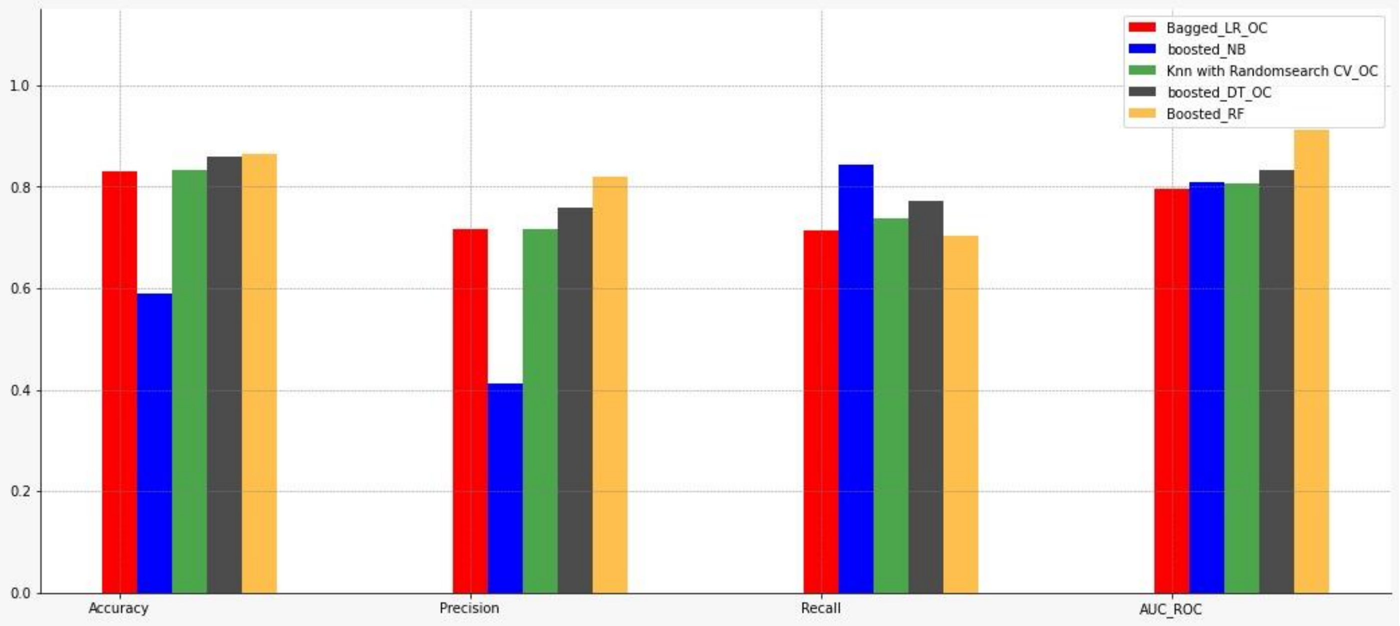
In this case weightage should be given to recall as classifying the patients who will be readmitted before 30 days plays a major role when compared with classifying the patients who will be readmitted after 30 days along with the ones with no readmission at all.

The best models from all the algorithms are represented in the table and chart below:

Models	Accuracy	Precision	Recall	AUC_ROC
Bagged_LR_OC	0.829626	0.71742	0.71304	0.796322
Boosted_NB	0.590291	0.411326	0.843021	0.808558
KNN with Randomsearch CV_OC	0.833252	0.715146	0.738306	0.806127
Boosted_DT_OC	0.857891	0.758458	0.772351	0.833455
Boosted_RF	0.864387	0.819823	0.702409	0.912479



How each metrics varies with respective to different models



4. CONCLUSION

The best model chosen from the previous section is Boosted Decision Tree with Optimum Cutoff and hence it can be taken into production where a different set of test data will be used and the classification can be performed with the obtained confidence as told by the model's metrics.

Hence we can expect our model to classify the patients who are about to be readmitted within 30 days with the confidence of 77.23% and the patients who will be readmitted after 30 days along with the ones with no readmission at all with the confidence of 75.84%. The overall accuracy of prediction can be expressed with the confidence of 85.78%.

From the analysis, it allows health centers to better anticipate and address unplanned readmissions while improving their quality of care and cost-efficiency.

5. FUTURE SCOPE

The patient readmission probability in US hospital Classification can be made use by hospitals by deploying the model in a large scale and hence the doctors are expected to enter the similar patient details so that they can get those patients classified into categories where they will be readmitted before 30 days and the patients who will be readmitted after 30 days along with the ones with no readmission at all.

This can help them cut costs in penalty which can further be diverted to funds that can support free medical camps which can help get a patient free diagnosis for his medical complications.

It can also result in the hospitals finding the problematic division within the hospital which has proven to get a higher readmission probability and take proper measures to solve the issue with that division thereby cutting costs even more and making patients comfortable with those divisions.

The reduced costs from penalty can support in developing the infrastructure of hospitals and afford advanced equipment’s for better healthcare practices.

6. REFERENCES

1. https://www.beckershospitalreview.com/finance/cms-penalizes-2-583-hospitals-for-high-readmissions-5-things-to-know.html

2. https://towardsdatascience.com/sampling-techniques-for-extremely-imbalanced-data-part-i-under-sampling-a8dbc3d8d6d8

3. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3258730

4. https://heartfailure.onlinejacc.org/content/8/1/1

5. https://www.sciencedirect.com/science/article/pii/S2213177919306754

6. https://www.researchgate.net/publication/325516717_Novel_Approach_to_Predict_Hospital_Readmissions_Using_Feature_Selection_from_Unstructured_Data_with_Class_Imbalance

7. https://ieeexplore.ieee.org/document/8440171

8. https://arxiv.org/ftp/arxiv/papers/1812/1812.11028.pdf

9. https://machinelearningmastery.com/statistical-hypothesis-tests-in-python-cheat-sheet/

10. https://machinelearningmastery.com/hyperparameters-for-classification-machine-learning-algorithms/