Final Report

Batch details	July’22 Online Batch
Team members	Aditya Aryan Mahima Bhardwaj Morvi Kohad Shailja Barsaiyan Shrika Garg
Domain of Project	Finance & Risk Analytics
Proposed project title	New York State Hospital Inpatient Discharge
Group Number	6
Team Leader	Shailja Barsaiyan
Mentor Name	Vidhya K

Date: 11/06/2023

Signature of the Mentor

Signature of the Team Leader

Table of Contents

Problem Statement and Current Insights

Background Research:

The healthcare industry generates profound amounts of data every day, this data can be used to gain insights and effectively improve the quality of patient care. Healthcare data analytics help improve how medical facilities function by using medical data analytics from a variety of sources. The primary objective of healthcare data analytics is to improve and streamline healthcare processes and optimize clinical, operational, financial, and experimental measures.

Healthcare analytics can help detect and predict healthcare fraud, thus mitigating risks and strengthening security. By analyzing clinical data through various sources, such as electronic medical records and personal health records, healthcare data analytics can be used for diagnosis and for enhancing clinical processes.

Literature Review:

- The healthcare sector is widely considered one of the most important industries in information technology. More and more, information technology has been considered as a practice that facilitates healthcare performance through using data and information efficiently within the healthcare sectors.
- Healthcare prediction is another data analytics method focusing on reducing future medical costs. The predictive technique uses the patient’s medical history to evaluate all the potential health risks and predict a future medical treatment in advance.
- Predictive analytics supports healthcare sectors to achieve a high level of effective overall care and preventive care, as predictive systems’ results allow treatments and actions to be taken when all the risks are recognized in the early stages, which aids for minimizing costs.

Problem Statement

The healthcare sector in New York (NY) is struggling to provide equal access to healthcare for all communities across the state. To address this issue, one approach is to examine regional disparities in healthcare utilization and outcomes, particularly for specific diagnoses or procedure codes.

The objective of this project is broken into the following points:

i. To **identify areas where additional healthcare coverage is needed** for **specific diagnoses** by understanding the **demographic variations**.

ii. To predict the **risk of patient mortality and duration of stay** by considering factors like age, gender, diagnosis, treatment, the severity of illness, and hospital information.

iii. By analyzing **demographic variations,** identify the specific diagnoses that require additional healthcare coverage in certain areas.

Dataset and Data Pre-processing

○ Dataset Description:

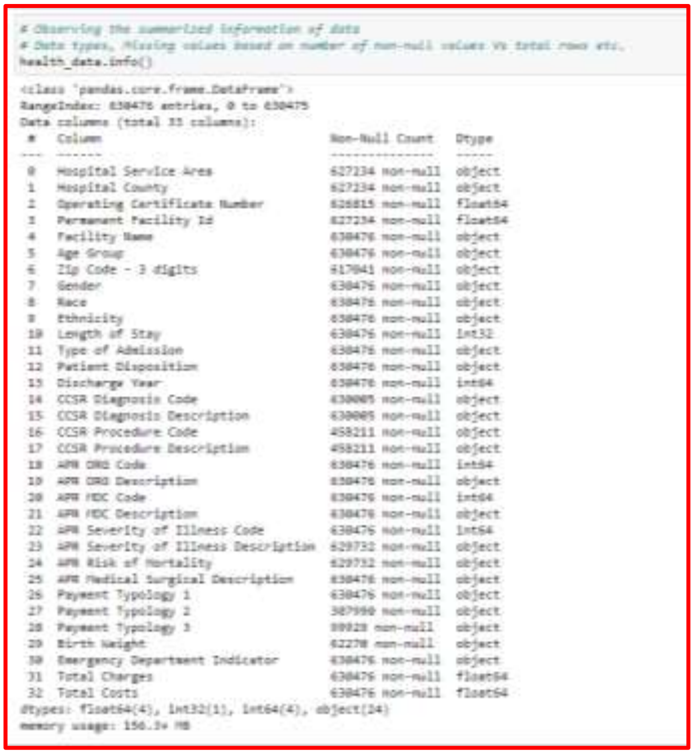
This is the public dataset made available by the Dept of Health of New York State. The dataset consists of 2101588 rows, and 33 columns which describe healthcare records on different types of parameters.

In order to simplify calculations and model building given the limitations of our machine's specifications, we have decided to work with a subset of 600,000 rows from our dataset, which originally contains over 2.1 million rows.

For our dataset, we have used numerical and categorical types of categorization. Using pandas function **dataframe.info(),** we have checked for the Data types, Missing values based on the number of non-null values and total rows for each column in the dataset.

a. In total, we have 9 columns having quantitative variables.

b. There are a total of 24 columns present having object data type.



```
# Observing the summarized information of data
# Data types, Missing values based on number of non-null values Vs total rows etc.
health_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 630476 entries, 0 to 630475
Data columns (total 33 columns):
 #   Column                               Non-Null Count   Dtype  
---  ------                               --------------   -----  
 0   Hospital Service Area                627234 non-null  object 
 1   Hospital County                      627234 non-null  object 
 2   Operating Certificate Number         626815 non-null  float64
 3   Permanent Facility Id                627234 non-null  float64
 4   Facility Name                        630476 non-null  object 
 5   Age Group                            630476 non-null  object 
 6   Zip Code - 3 digits                  617041 non-null  object 
 7   Gender                               630476 non-null  object 
 8   Race                                 630476 non-null  object 
 9   Ethnicity                            630476 non-null  object 
 10  Length of Stay                       630476 non-null  int32  
 11  Type of Admission                    630476 non-null  object 
 12  Patient Disposition                  630476 non-null  object 
 13  Discharge Year                       630476 non-null  int64  
 14  CCSR Diagnosis Code                  630005 non-null  object 
 15  CCSR Diagnosis Description           630005 non-null  object 
 16  CCSR Procedure Code                  458211 non-null  object 
 17  CCSR Procedure Description           458211 non-null  object 
 18  APR DRG Code                         630476 non-null  int64  
 19  APR DRG Description                  630476 non-null  object 
 20  APR MDC Code                         630476 non-null  int64  
 21  APR MDC Description                  630476 non-null  object 
 22  APR Severity of Illness Code         630476 non-null  int64  
 23  APR Severity of Illness Description  629732 non-null  object 
 24  APR Risk of Mortality                629732 non-null  object 
 25  APR Medical Surgical Description     630476 non-null  object 
 26  Payment Typology 1                   630476 non-null  object 
 27  Payment Typology 2                   387990 non-null  object 
 28  Payment Typology 3                   99928 non-null   object 
 29  Birth Weight                         62278 non-null   object 
 30  Emergency Department Indicator       630476 non-null  object 
 31  Total Charges                        630476 non-null  float64
 32  Total Costs                          630476 non-null  float64
dtypes: float64(4), int32(1), int64(4), object(24)
memory usage: 156.3+ MB
```

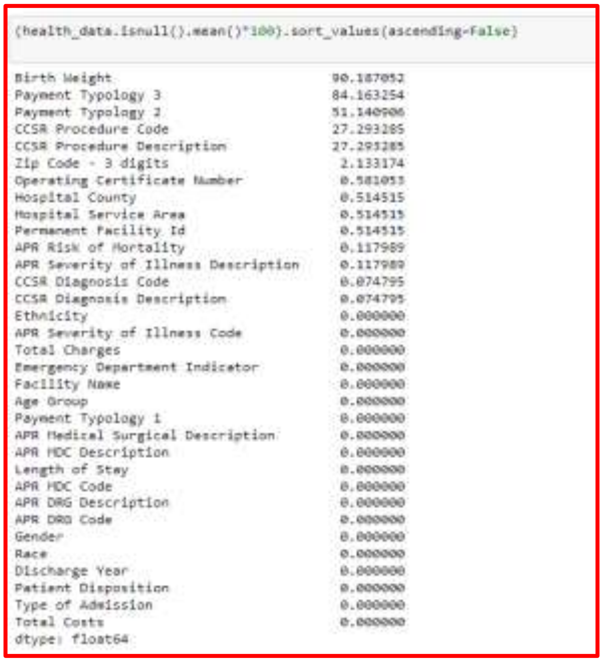
- ## Pre-processing

a. Check Null Values and Unnecessary Columns

In the process of analyzing data, preprocessing plays a vital role as it involves performing activities such as cleaning, transforming, and refining the data to make it suitable for analysis. The primary objective of this step is to ensure the accuracy, completeness, and readiness of the data for further analysis.

From the above-mentioned numbers and the image, we have null values in our dataset. So, in order to identify the percentage of null values we have used the pandas dataframe function isnull(). Further, to find the percentage of the null values, we have used mean().

Screenshot from the notebook:



```
(health_data.isnull().mean()*100).sort_values(ascending=False)
```

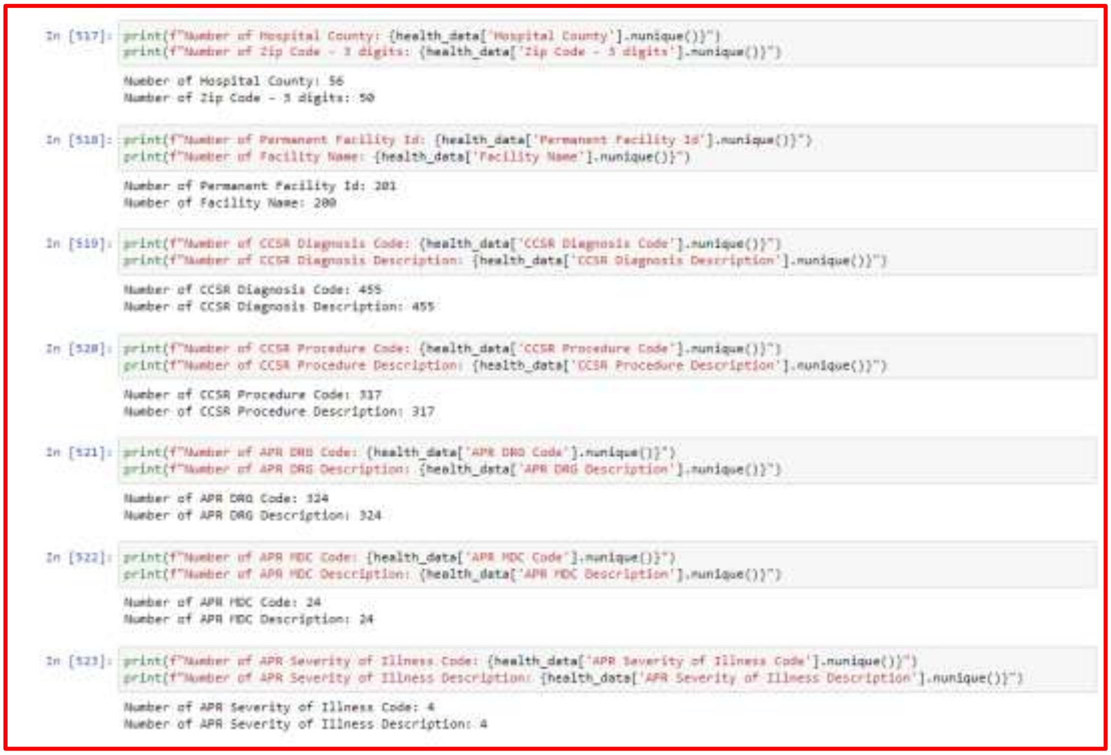
```
Birth Weight                               90.187052
Payment Typology 3                         84.163254
Payment Typology 2                         51.140906
CCSR Procedure Code                        27.293285
CCSR Procedure Description                 27.293285
Zip Code - 3 digits                         2.133174
Operating Certificate Number                0.581053
Hospital County                             0.514515
Hospital Service Area                       0.514515
Permanent Facility Id                       0.514515
APR Risk of Mortality                       0.117989
APR Severity of Illness Description         0.117989
CCSR Diagnosis Code                         0.074795
CCSR Diagnosis Description                  0.074795
Ethnicity                                   0.000000
APR Severity of Illness Code                0.000000
Total Charges                               0.000000
Emergency Department Indicator              0.000000
Facility Name                               0.000000
Age Group                                   0.000000
Payment Typology 1                          0.000000
APR Medical Surgical Description            0.000000
APR MDC Description                         0.000000
Length of Stay                              0.000000
APR MDC Code                                0.000000
APR DRG Description                         0.000000
APR DRG Code                                0.000000
Gender                                      0.000000
Race                                        0.000000
Discharge Year                              0.000000
Patient Disposition                         0.000000
Type of Admission                           0.000000
Total Costs                                 0.000000
dtype: float64
```

*From the above output, it is feasible to drop **Birth Weight**, **Payment Typology 2**, **Payment Typology 3** columns as it contains more than 50% null values and it will be difficult to extract value from them. Also, dropping rows from Zip Code - 3 digit, Hospital County, Hospital Service Area, Permanent Facility Id, APR Risk of Mortality, APR Severity of Illness Description, CCSR Diagnosis Code, CCSR Diagnosis Description columns which contains NaN values, as it is relatively small number of NaN values and thus will be insignificant to drop.*

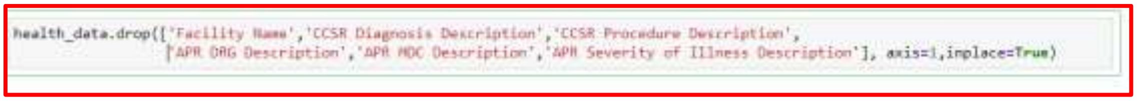
b. Check for Redundant columns

On inspection, we found there are redundant columns, so dropped Facility Name, CCSR Diagnosis Description, CCSR Procedure Description, APR DRG Description, APR MDC Description columns to have more clear analysis using the above steps as they are forming redundancy.

Screenshot from the notebook:



```
In [517]: print(f"Number of Hospital County: {health_data['Hospital County'].nunique()}")
          print(f"Number of Zip Code - 3 digits: {health_data['Zip Code - 3 digits'].nunique()}")

          Number of Hospital County: 56
          Number of Zip Code - 3 digits: 50

In [518]: print(f"Number of Permanent Facility Id: {health_data['Permanent Facility Id'].nunique()}")
          print(f"Number of Facility Name: {health_data['Facility Name'].nunique()}")

          Number of Permanent Facility Id: 201
          Number of Facility Name: 200

In [519]: print(f"Number of CCSR Diagnosis Code: {health_data['CCSR Diagnosis Code'].nunique()}")
          print(f"Number of CCSR Diagnosis Description: {health_data['CCSR Diagnosis Description'].nunique()}")

          Number of CCSR Diagnosis Code: 455
          Number of CCSR Diagnosis Description: 455

In [520]: print(f"Number of CCSR Procedure Code: {health_data['CCSR Procedure Code'].nunique()}")
          print(f"Number of CCSR Procedure Description: {health_data['CCSR Procedure Description'].nunique()}")

          Number of CCSR Procedure Code: 317
          Number of CCSR Procedure Description: 317

In [521]: print(f"Number of APR DRG Code: {health_data['APR DRG Code'].nunique()}")
          print(f"Number of APR DRG Description: {health_data['APR DRG Description'].nunique()}")

          Number of APR DRG Code: 324
          Number of APR DRG Description: 324

In [522]: print(f"Number of APR MDC Code: {health_data['APR MDC Code'].nunique()}")
          print(f"Number of APR MDC Description: {health_data['APR MDC Description'].nunique()}")

          Number of APR MDC Code: 24
          Number of APR MDC Description: 24

In [523]: print(f"Number of APR Severity of Illness Code: {health_data['APR Severity of Illness Code'].nunique()}")
          print(f"Number of APR Severity of Illness Description: {health_data['APR Severity of Illness Description'].nunique()}")

          Number of APR Severity of Illness Code: 4
          Number of APR Severity of Illness Description: 4
```

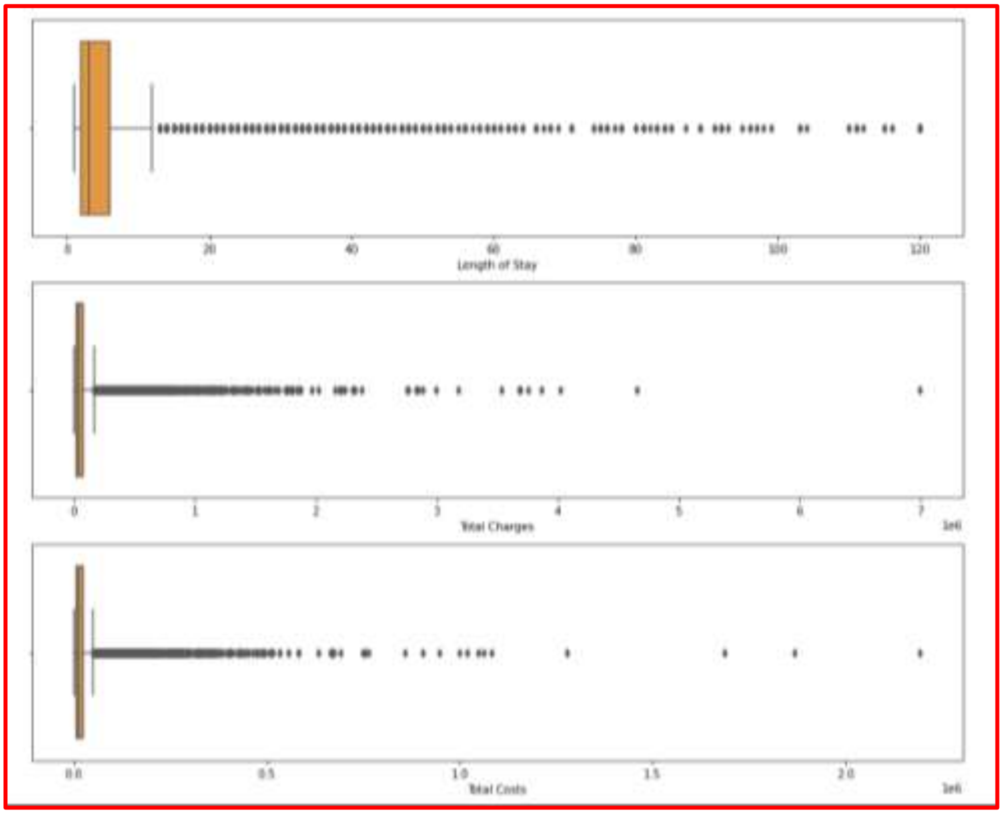


```
health_data.drop(['Facility Name','CCSR Diagnosis Description','CCSR Procedure Description',
                 'APR DRG Description','APR MDC Description','APR Severity of Illness Description'], axis=1,inplace=True)
```

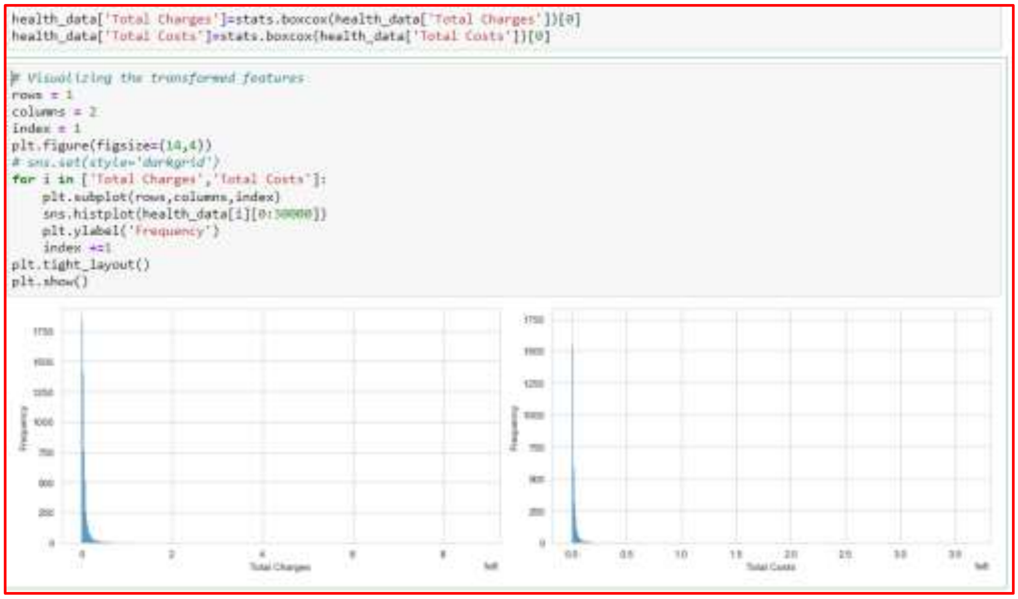
c. Check for Outliers

Outliers are data points that are significantly different from other data points in a dataset. They can occur due to measurement errors, data entry errors, or real-world phenomena that deviate from the norm. Outliers can significantly affect the results of statistical analyses, as they can skew the mean and standard deviation of a dataset and lead to incorrect conclusions.

Screenshot from the notebook:

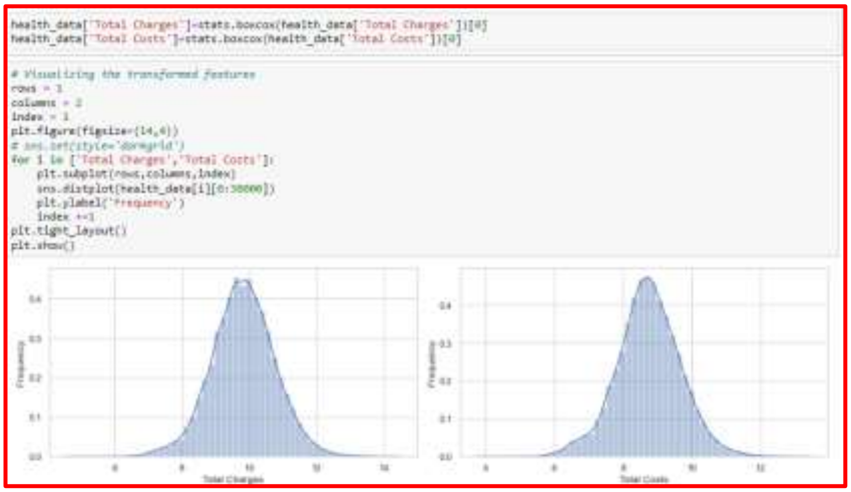


From the above graphs, we can see that there are outliers in the Total Charges and Total Costs, which were handled through transformation.



```
health_data['Total Charges']=stats.boxcox(health_data['Total Charges'])[0]
health_data['Total Costs']=stats.boxcox(health_data['Total Costs'])[0]
```

```
# Visualizing the transformed features
rows = 1
columns = 2
index = 1
plt.figure(figsize=(14,4))
# sns.set(style='darkgrid')
for i in ['Total Charges','Total Costs']:
    plt.subplot(rows,columns,index)
    sns.histplot(health_data[i][0:30000])
    plt.ylabel('Frequency')
    index +=1
plt.tight_layout()
plt.show()
```



```
health_data['Total Charges']=stats.boxcox(health_data['Total Charges'])[0]
health_data['Total Costs']=stats.boxcox(health_data['Total Costs'])[0]
```

```
# Visualizing the transformed features
rows = 1
columns = 2
index = 1
plt.figure(figsize=(14,4))
# sns.set(style='darkgrid')
for i in ['Total Charges','Total Costs']:
    plt.subplot(rows,columns,index)
    sns.distplot(health_data[i][0:30000])
    plt.ylabel('Frequency')
    index +=1
plt.tight_layout()
plt.show()
```

After applying the transformation technique, we can see that the data has now been normally transformed.

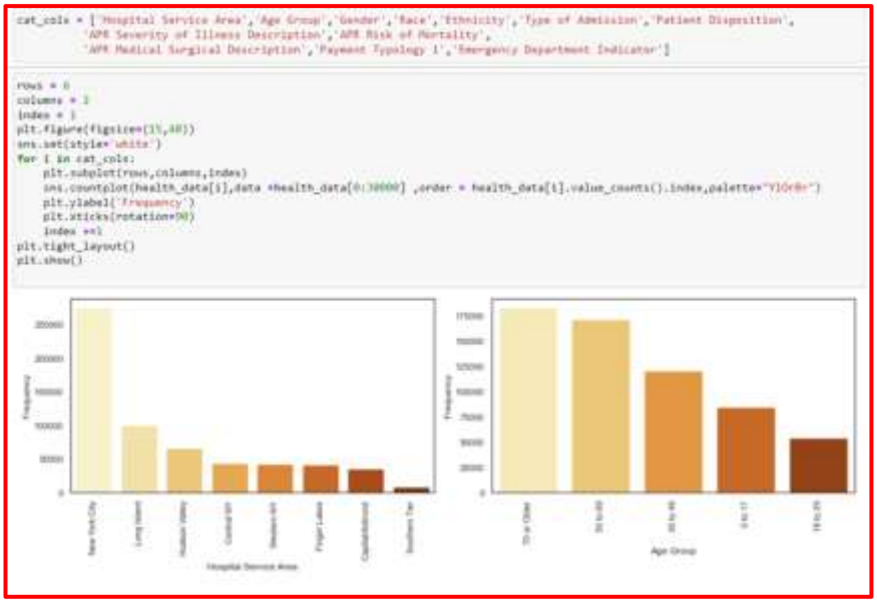
○ Analysis of the data

a. Univariate Analysis

To better understand the variables, we have plotted graphs for every column in the dataset, also called univariate analysis. It involves analyzing a single variable in isolation, without considering its relationship with other variables in the dataset. Univariate analysis can be used to determine the central tendency of the variable, its dispersion or variability, and its shape or distribution.

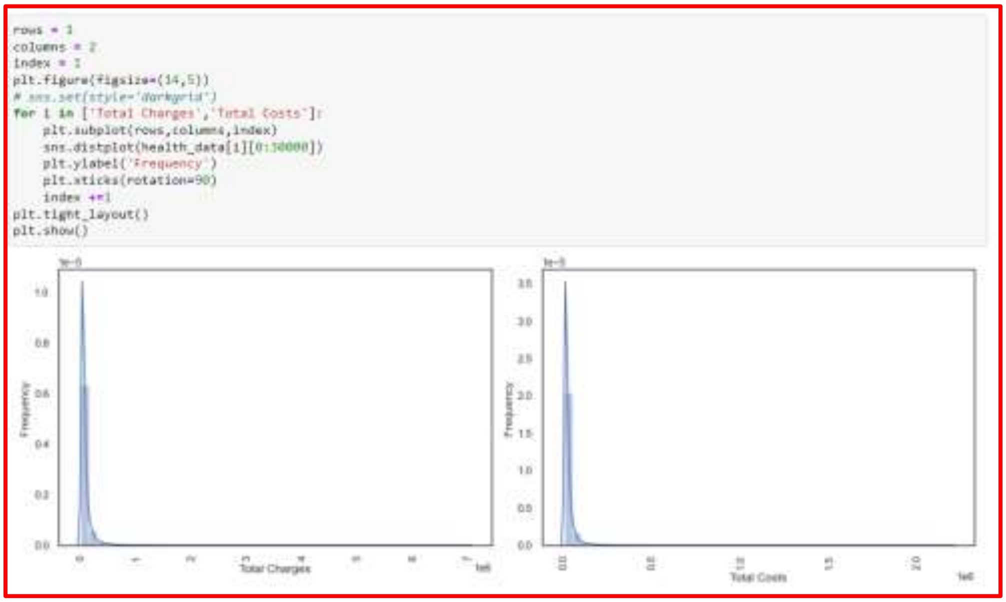
The first step for univariate analysis is that we have plotted graphs for categorical columns.

Screenshot from the notebook:



```
cat_cols = ['Hospital Service Area','Age Group','Gender','Race','Ethnicity','Type of Admission','Patient Disposition',
          'APR Severity of Illness Description','APR Risk of Mortality',
          'APR Medical Surgical Description','Payment Typology 1','Emergency Department Indicator']
```

```
rows = 6
columns = 2
index = 1
plt.figure(figsize=(15,40))
sns.set(style='white')
for i in cat_cols:
    plt.subplot(rows,columns,index)
    sns.countplot(health_data[i],data =health_data[0:30000] ,order = health_data[i].value_counts().index,palette="YlOrBr")
    plt.ylabel('Frequency')
    plt.xticks(rotation=90)
    index +=1
plt.tight_layout()
plt.show()
```



```
rows = 1
columns = 2
index = 1
plt.figure(figsize=(14,5))
# sns.set(style='darkgrid')
for i in ['Total Charges','Total Costs']:
    plt.subplot(rows,columns,index)
    sns.distplot(health_data[i][0:30000])
    plt.ylabel('Frequency')
    plt.xticks(rotation=90)
    index +=1
plt.tight_layout()
plt.show()
```

From the above plot came across with the inferences:

1. *No of patients are quietly more in New York city than other cities.*
2. *More number of patients fall in the age category 70 years or older.*
3. *Female patients are more in number as compare to male patients.*
4. *Patients enrolled in Emergency are huge in number.*
5. *Patient Disposition i.e., patients' destination after discharge, mostly are in Home or Self Care prescription.*
6. *Mostly patients have Minor Risk of Mortality.*
7. *Patients have Medicare payment typology more.*
8. *Total charges and Total Costs have been positively skewed.*

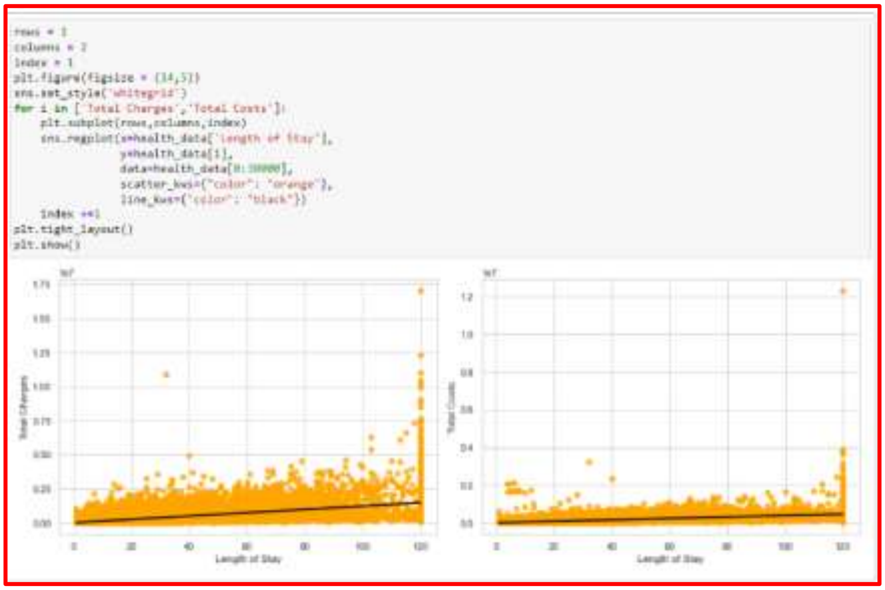
b. Bivariate Analysis

The relationship between variables refers to how two or more variables are associated or related to each other. Understanding the relationship between variables is important in data analysis and can help identify patterns, trends, and associations in the data.

Firstly, we did bivariate analysis for the numerical variables. In the below image, first graph represents the relationship between length of stay and total charges. On the other hand, second graph concludes the relationship of length of stay with total costs.

From both of the graphs, it can be concluded that Total charges and Total Costs have a significance positive relation.

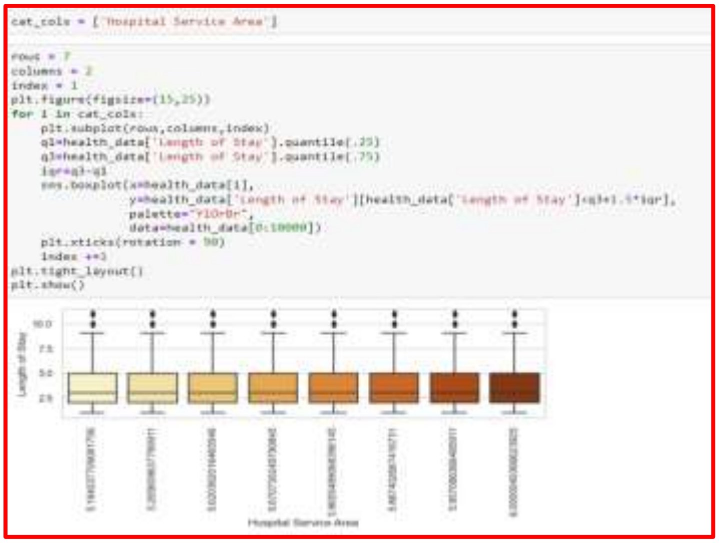
Screenshot from the notebook:



```
rows = 1
columns = 2
index = 1
plt.figure(figsize = (14,5))
sns.set_style('whitegrid')
for i in ['Total Charges','Total Costs']:
    plt.subplot(rows,columns,index)
    sns.regplot(x=health_data['Length of Stay'],
                y=health_data[i],
                data=health_data[0:30000],
                scatter_kws={"color": "orange"},
                line_kws={"color": "black"})
    index +=1
plt.tight_layout()
plt.show()
```

Secondly, we did bivariate analysis for categorical variables.
Here, in cat_cols, we have passed the following column names: Age Group, Gender, Race, Ethnicity, Type of Admission, Patient Disposition, APR Severity of Illness Description, APR Risk of Mortality, APR Medical Surgical Description, Payment Typology 1, Emergency Department Indicator to get the plots for every variable against target variable (Length of stay).

Screenshot from the notebook:



```
cat_cols = ['Hospital Service Area']
```

```
rows = 7
columns = 2
index = 1
plt.figure(figsize=(15,25))
for i in cat_cols:
    plt.subplot(rows,columns,index)
    q1=health_data['Length of Stay'].quantile(.25)
    q3=health_data['Length of Stay'].quantile(.75)
    iqr=q3-q1
    sns.boxplot(x=health_data[i],
                y=health_data['Length of Stay'][health_data['Length of Stay']<q3+1.5*iqr],
                palette="YlOrBr",
                data=health_data[0:10000])
    plt.xticks(rotation = 90)
    index +=1
plt.tight_layout()
plt.show()
```

From the graphs we have plotted for the length of stay with our categorical variables, the following inferences can be made:

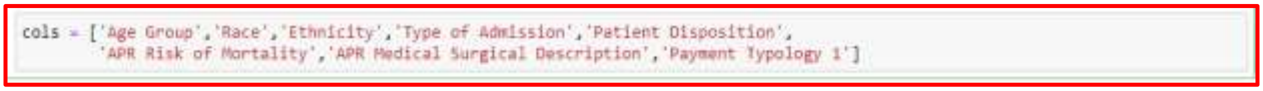
1. *Children aged up to 17 years have the shortest lengths of stay.*

2. *Length of stay for newborns is usually 2 days.*
3. *Patients disposed to home care or critical access hospitals after discharge, and patients leaving against medical advice stay the shortest.*
4. *Patients disposed to nursing homes, rehabilitation facilities or long term care hospitals usually stay longer.*
5. *People with least severe ailments have lowest lengths of stay while people with extremely severe illnesses have the highest.*
6. *People with least severe ailments have lowest lengths of stay while people with extremely severe illnesses have the highest.*
7. *Similarly, people with a low risk of mortality have shortest stays.*
8. *Patients admitted in an emergency have higher lengths of stay than otherwise.*

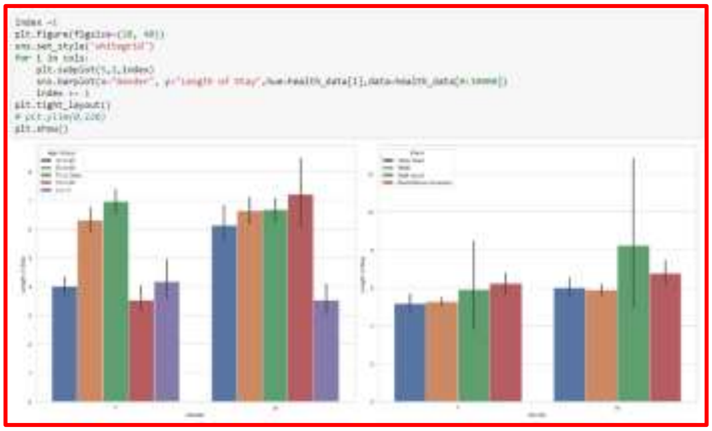
c. Multivariate Analysis

And lastly, we did multivariate analysis for the columns, Age Group, Race, Ethnicity, Type of Admission, Patient Disposition, APR Risk of Mortality, APR Medical Surgical Description, Payment Typology 1 with gender and length of stay.

Screenshot from the notebook:



```
cols = ['Age Group','Race','Ethnicity','Type of Admission','Patient Disposition',
        'APR Risk of Mortality','APR Medical Surgical Description','Payment Typology 1']
```



```
index =1
plt.figure(figsize=(18, 40))
sns.set_style('whitegrid')
for i in cols:
    plt.subplot(5,1,index)
    sns.barplot(x='Gender', y='Length of Stay',hue=health_data[i],data=health_data[0:10000])
    index += 1
plt.tight_layout()
# plt.ylim(0,220)
plt.show()
```

The following inferences are made from the above graphs:

1. *Males of each group 0 to 17 are staying more as compared to females.*
2. *Multi-racial and Multi-ethnic patients stay longer, with not much effect of gender.*
3. *Patients admitted to Trauma, Emergency, and Urgent category have a longer stay.*

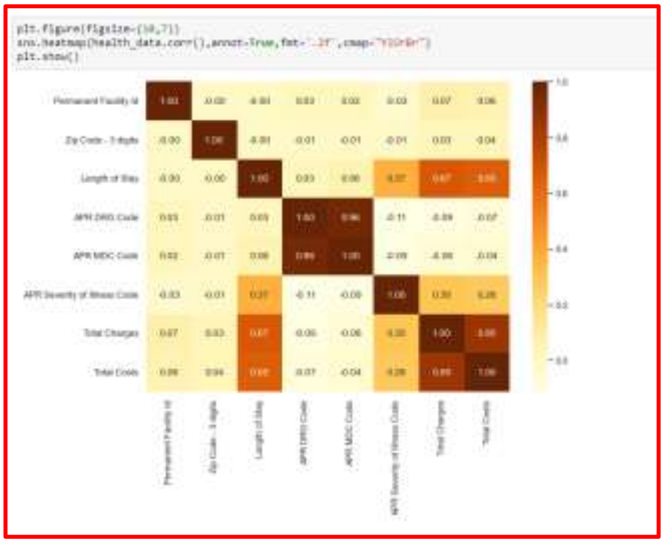
4. Cancer Center or Children's Hospital Females stayed longer than men on the same prescription.

5. Males with surgical description had longer stay than females in the same.

6. Patients who opted for Payment Typology Medicare or Medicaid stayed longer in the hospital.

Plotted heatmap to find the correlation between the features.

Screenshot from the notebook:



```
plt.figure(figsize=(10,7))
sns.heatmap(health_data.corr(),annot=True,fmt='.2f',cmap="YlOrBr")
plt.show()
```

From the above plot, Total Charges and Total Costs have a strong positive correlation with each other, and these two features are also positively correlated with the Length of Stay column.

APR MDC code and APR DRG Code are also strongly positively correlated but they are more like a serial number.

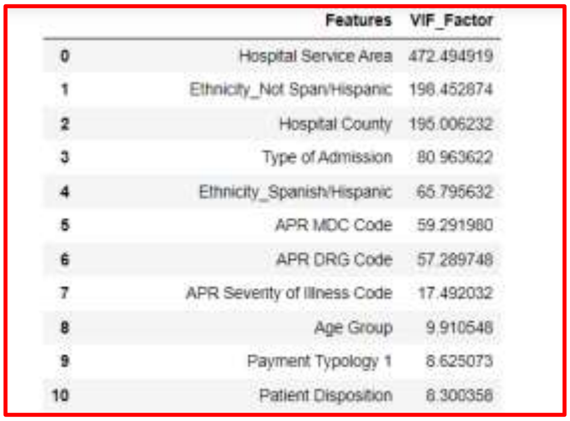
- ## Significant Variables

a. Removing highly correlated Independent Variables

Multicollinearity is a statistical phenomenon where two or more independent variables in a regression model are highly correlated with each other. Multicollinearity can lead to unstable or unreliable regression coefficients and inflated standard errors, making it difficult to assess the statistical significance of individual variables in the model.

To detect multicollinearity, we have used variance inflation factors (VIF). Since, multicollinearity is detected, it is addressed by removing one of the highly correlated variables.

Screenshot from the notebook:



	Features	VIF_Factor
0	Hospital Service Area	472.494919
1	Ethnicity_Not Span/Hispanic	198.452874
2	Hospital County	195.006232
3	Type of Admission	80.963622
4	Ethnicity_Spanish/Hispanic	65.795632
5	APR MDC Code	59.291980
6	APR DRG Code	57.289748
7	APR Severity of Illness Code	17.492032
8	Age Group	9.910548
9	Payment Typology 1	8.625073
10	Patient Disposition	8.300358

*Removed Features Hospital Service Area, Ethnicity, Not Span/Hispanic, Type of Admission, APR MDC Code, Hospital County***,** *APR Severity of Illness Code using VIF.*

The output shows that the variable Ethnicity Not Span/Hispanic has the highest VIF. Removing this feature from the dataset and set the threshold of VIF as to 10, it means considering feature having VIF less than or equal to 10 (can be changed as per business requirement.

b. Removing Insignificant Variables

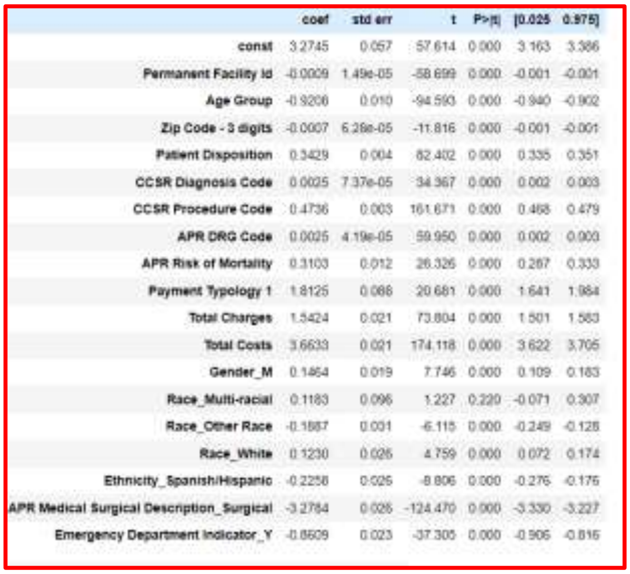
Feature selection is the process of selecting a subset of relevant features from a larger set of features to be used in a machine learning model. The goal of feature selection is to reduce the dimensionality of the data and to improve the performance of the model by removing irrelevant or redundant features.

Statistical significance of variables refers to the probability that the estimated regression coefficient is different from zero in a statistical sense. A variable is considered statistically significant if its coefficient is significantly different from zero at a given level of confidence (usually 95% or 99%).

To assess the statistical significance of variables in a regression model, one can use hypothesis testing with a significance level (alpha) of 0.05. This involves calculating a p-value, which represents the probability of observing a coefficient as extreme as the one estimated in the model, assuming the null hypothesis (that the coefficient is zero) is true.

If the p-value is less than the chosen significance level, then the variable is considered statistically significant and can be interpreted as having a non-zero effect on the dependent variable. If the p-value is greater than the significance level, then the variable is considered not statistically significant and its effect on the dependent variable is considered to be uncertain or negligible.

Screenshot from the notebook:

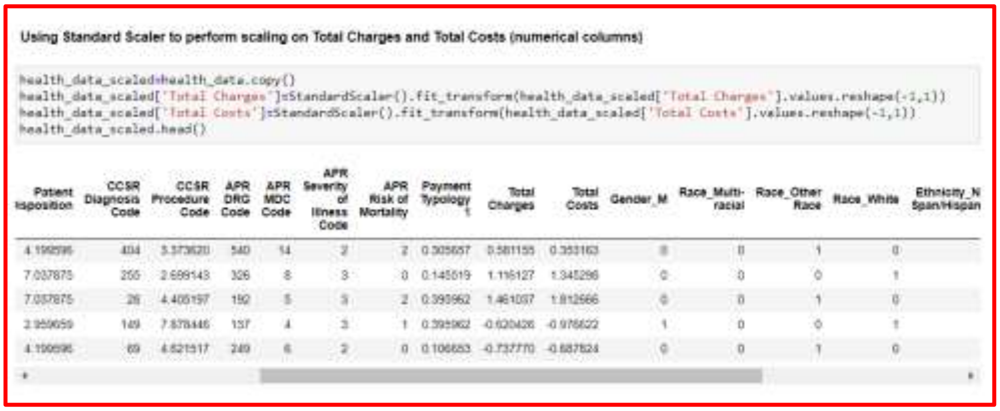


	coef	std err	t	P>|t|	[0.025	0.975]
const	3.2745	0.057	57.614	0.000	3.163	3.386
Permanent Facility Id	-0.0009	1.49e-05	-58.699	0.000	-0.001	-0.001
Age Group	-0.9206	0.010	-94.593	0.000	-0.940	-0.902
Zip Code - 3 digits	-0.0007	6.28e-05	-11.816	0.000	-0.001	-0.001
Patient Disposition	0.3429	0.004	82.402	0.000	0.335	0.351
CCSR Diagnosis Code	0.0025	7.37e-05	34.367	0.000	0.002	0.003
CCSR Procedure Code	0.4736	0.003	161.671	0.000	0.468	0.479
APR DRG Code	0.0025	4.19e-05	59.950	0.000	0.002	0.003
APR Risk of Mortality	0.3103	0.012	26.326	0.000	0.287	0.333
Payment Typology 1	1.8125	0.088	20.681	0.000	1.641	1.984
Total Charges	1.5424	0.021	73.804	0.000	1.501	1.583
Total Costs	3.6633	0.021	174.118	0.000	3.622	3.705
Gender_M	0.1464	0.019	7.746	0.000	0.109	0.183
Race_Multi-racial	0.1183	0.096	1.227	0.220	-0.071	0.307
Race_Other Race	-0.1887	0.031	-6.115	0.000	-0.249	-0.128
Race_White	0.1230	0.026	4.759	0.000	0.072	0.174
Ethnicity_Spanish/Hispanic	-0.2258	0.026	-8.806	0.000	-0.276	-0.176
APR Medical Surgical Description_Surgical	-3.2784	0.026	-124.470	0.000	-3.330	-3.227
Emergency Department Indicator_Y	-0.8609	0.023	-37.305	0.000	-0.906	-0.816

From the above output snippet, it is clear that the column Race_Multi-racial is having P value 0.833 which is greater than 0.05. So, we have to drop the column and rebuild the model.

- ## <u>Scaling of the dataset</u>

Scaling of data is a common preprocessing step in machine learning and data analysis. It involves transforming the features of the data so that they have a similar scale or range. Scaling is important because many machine learning algorithms are sensitive to the scale of the input features, and failure to scale the data can result in inaccurate or biased results. Since, data is normally distributed, so we have used standardization technique.



Using Standard Scaler to perform scaling on Total Charges and Total Costs (numerical columns)

```
health_data_scaled=health_data.copy()
health_data_scaled['Total Charges']=StandardScaler().fit_transform(health_data_scaled['Total Charges'].values.reshape(-1,1))
health_data_scaled['Total Costs']=StandardScaler().fit_transform(health_data_scaled['Total Costs'].values.reshape(-1,1))
health_data_scaled.head()
```

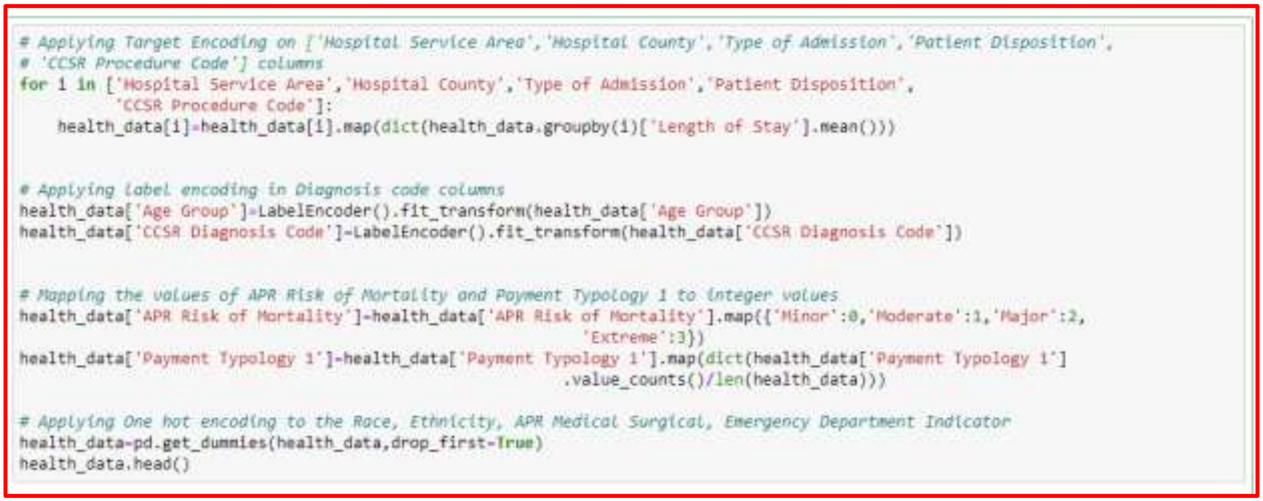
Patient lisposition	CCSR Diagnosis Code	CCSR Procedure Code	APR DRG Code	APR MDC Code	APR Severity of Illness Code	APR Risk of Mortality	Payment Typology 1	Total Charges	Total Costs	Gender_M	Race_Multi-racial	Race_Other Race	Race_White	Ethnicity_N Span/Hispan
4.199596	404	3.373620	540	14	2	2	0.305657	0.581155	0.353163	0	0	1	0	
7.037875	255	2.699143	326	8	3	0	0.145519	1.116127	1.345296	0	0	0	1	
7.037875	26	4.405197	192	5	3	2	0.393962	1.461037	1.812566	0	0	1	0	
2.959659	149	7.878446	157	4	3	1	0.395962	-0.620426	-0.976622	1	0	0	1	
4.199596	69	4.621517	249	6	2	0	0.106653	-0.737770	-0.687824	0	0	1	0	

Model Building and Evaluation

Regression Problem Statement for Predicting Length of Stay

Feature Encoding before performing Modeling

Categorical columns in the dataset were transformed into numerical values using Target Encoding, Label Encoding, and One Hot Encoding techniques. This conversion facilitated the modeling process by enabling the algorithms to work effectively with the categorical data.



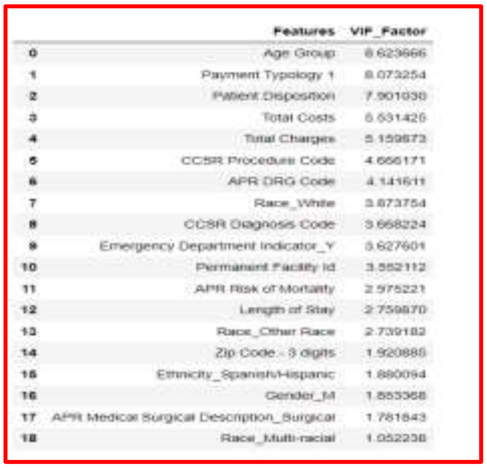
```
# Applying Target Encoding on ['Hospital Service Area','Hospital County','Type of Admission','Patient Disposition',
# 'CCSR Procedure Code'] columns
for i in ['Hospital Service Area','Hospital County','Type of Admission','Patient Disposition',
          'CCSR Procedure Code']:
    health_data[i]=health_data[i].map(dict(health_data.groupby(i)['Length of Stay'].mean()))


# Applying label encoding in Diagnosis code columns
health_data['Age Group']=LabelEncoder().fit_transform(health_data['Age Group'])
health_data['CCSR Diagnosis Code']=LabelEncoder().fit_transform(health_data['CCSR Diagnosis Code'])


# Mapping the values of APR Risk of Mortality and Payment Typology 1 to integer values
health_data['APR Risk of Mortality']=health_data['APR Risk of Mortality'].map({'Minor':0,'Moderate':1,'Major':2,
                                                                              'Extreme':3})
health_data['Payment Typology 1']=health_data['Payment Typology 1'].map(dict(health_data['Payment Typology 1']
                                                                        .value_counts()/len(health_data)))

# Applying One hot encoding to the Race, Ethnicity, APR Medical Surgical, Emergency Department Indicator
health_data=pd.get_dummies(health_data,drop_first=True)
health_data.head()
```

Base Model

Assumptions before performing Linear Regression

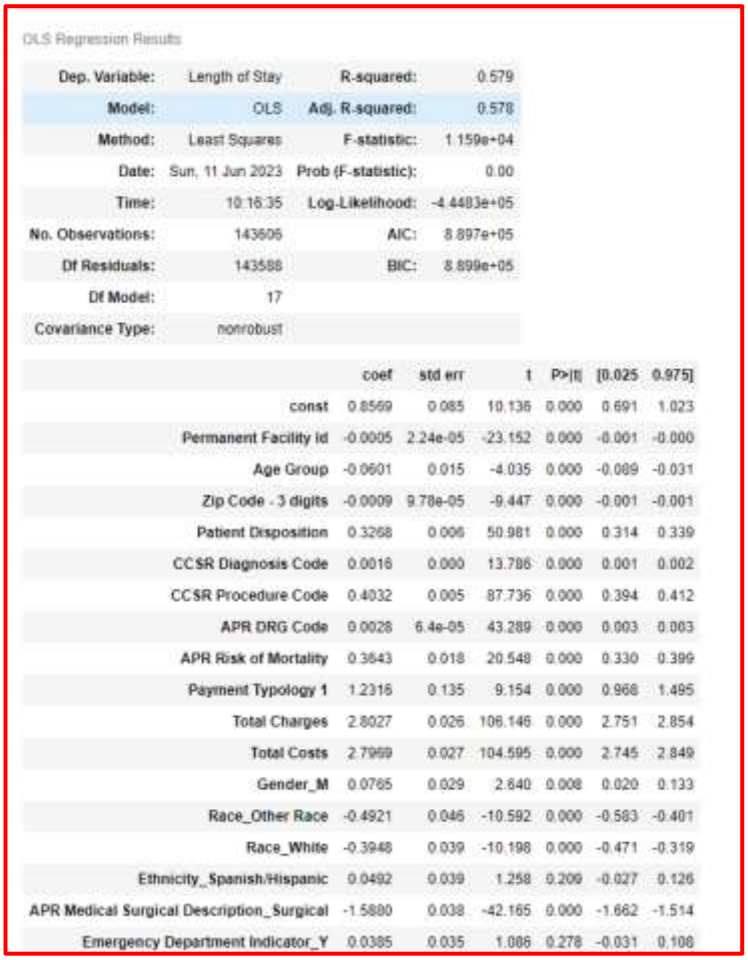
1. Target Variable should be numeric.
2. Predictors must not show multicollinearity.

Screenshot from the notebook:



	Features	VIF_Factor
0	Age Group	8.623666
1	Payment Typology 1	8.073254
2	Patient Disposition	7.901030
3	Total Costs	6.631426
4	Total Charges	5.159873
5	CCSR Procedure Code	4.666171
6	APR DRG Code	4.141611
7	Race_White	3.873754
8	CCSR Diagnosis Code	3.668224
9	Emergency Department Indicator_Y	3.627601
10	Permanent Facility Id	3.552112
11	APR Risk of Mortality	2.975221
12	Length of Stay	2.759870
13	Race_Other Race	2.739182
14	Zip Code - 3 digits	1.920886
15	Ethnicity_Spanish/Hispanic	1.880094
16	Gender_M	1.853368
17	APR Medical Surgical Description_Surgical	1.781843
18	Race_Multi-racial	1.052238

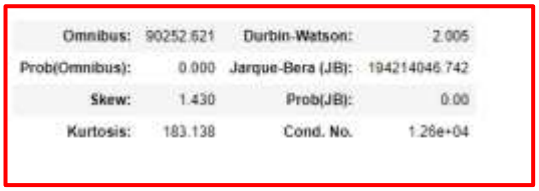
Removed features that had VIF less than the threshold from the dataset and set the threshold of VIF as to 10, which means considering features having VIF less than or equal to 10 (can be changed as per business requirement.



OLS Regression Results

Dep. Variable:	Length of Stay	R-squared:	0.579
Model:	OLS	Adj. R-squared:	0.578
Method:	Least Squares	F-statistic:	1.159e+04
Date:	Sun, 11 Jun 2023	Prob (F-statistic):	0.00
Time:	10:16:35	Log-Likelihood:	-4.4483e+05
No. Observations:	143606	AIC:	8.897e+05
Df Residuals:	143588	BIC:	8.899e+05
Df Model:	17		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.8569	0.085	10.136	0.000	0.691	1.023
Permanent Facility Id	-0.0005	2.24e-05	-23.152	0.000	-0.001	-0.000
Age Group	-0.0601	0.015	-4.035	0.000	-0.089	-0.031
Zip Code - 3 digits	-0.0009	9.78e-05	-9.447	0.000	-0.001	-0.001
Patient Disposition	0.3268	0.006	50.981	0.000	0.314	0.339
CCSR Diagnosis Code	0.0016	0.000	13.786	0.000	0.001	0.002
CCSR Procedure Code	0.4032	0.005	87.736	0.000	0.394	0.412
APR DRG Code	0.0028	6.4e-05	43.289	0.000	0.003	0.003
APR Risk of Mortality	0.3643	0.018	20.548	0.000	0.330	0.399
Payment Typology 1	1.2316	0.135	9.154	0.000	0.968	1.495
Total Charges	2.8027	0.026	106.146	0.000	2.751	2.854
Total Costs	2.7969	0.027	104.595	0.000	2.745	2.849
Gender_M	0.0765	0.029	2.640	0.008	0.020	0.133
Race_Other Race	-0.4921	0.046	-10.592	0.000	-0.583	-0.401
Race_White	-0.3948	0.039	-10.198	0.000	-0.471	-0.319
Ethnicity_Spanish/Hispanic	0.0492	0.039	1.258	0.209	-0.027	0.126
APR Medical Surgical Description_Surgical	-1.5880	0.038	-42.165	0.000	-1.662	-1.514
Emergency Department Indicator_Y	0.0385	0.035	1.086	0.278	-0.031	0.108



Omnibus:	90252.621	Durbin-Watson:	2.005
Prob(Omnibus):	0.000	Jarque-Bera (JB):	194214046.742
Skew:	1.430	Prob(JB):	0.00
Kurtosis:	183.138	Cond. No.	1.26e+04

Assumptions of MLR Model:

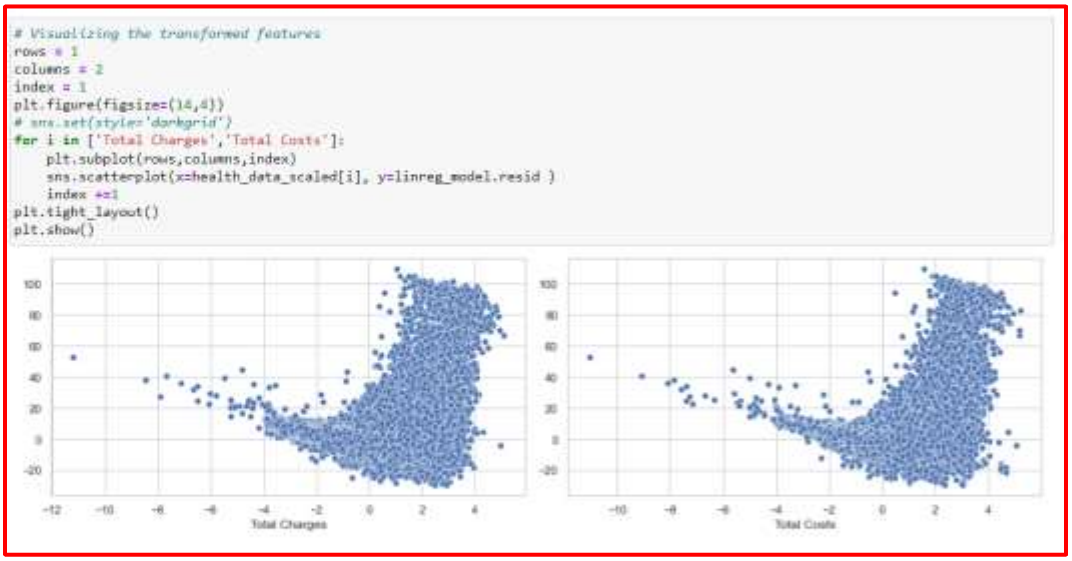
Now, use the model with significant variables to check some of the assumptions based on the residuals of linear regression:

1. Linear Relationship Between Dependent and Independent Variable
2. Autocorrelation
3. Heteroscedasticity
4. Tests of Normality

1. Linear Relationship between Dependent and Independent Variable

An assumption of linear regression is that it should be linear in the parameter. To check the linearity, we plotted a graph of residuals and each independent variable. If the plot shows no specific pattern, then we can conclude the presence of linearity.

Screenshot from the notebook:



```
# Visualizing the transformed features
rows = 1
columns = 2
index = 1
plt.figure(figsize=(14,4))
# sns.set(style='darkgrid')
for i in ['Total Charges','Total Costs']:
    plt.subplot(rows,columns,index)
    sns.scatterplot(x=health_data_scaled[i], y=linreg_model.resid )
    index +=1
plt.tight_layout()
plt.show()
```

The above plots show no specific pattern, implies that there is a linearity present in the data.

2. Autocorrelation

From the above summary, we can observe that the value obtained from the Durbin-Watson test statistic is close to 2 (= 2.001). Thus, we conclude that there is no autocorrelation.

3. Heteroscedasticity

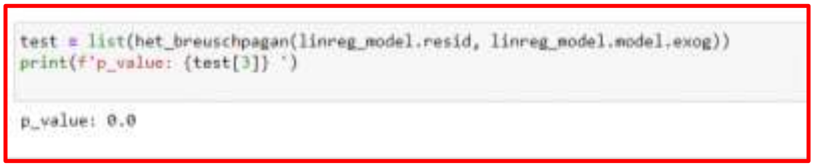
Breusch-Pagan is one of the tests for detecting heteroskedasticity in the residuals.

The test hypothesis for the Breusch-Pagan test is given as:

H_o: There is homoscedasticity present in the data

H_1: There is a heteroscedasticity present in the data

Screenshot from the notebook:



```
test = list(het_breuschpagan(linreg_model.resid, linreg_model.model.exog))
print(f'p_value: {test[3]} ')
```

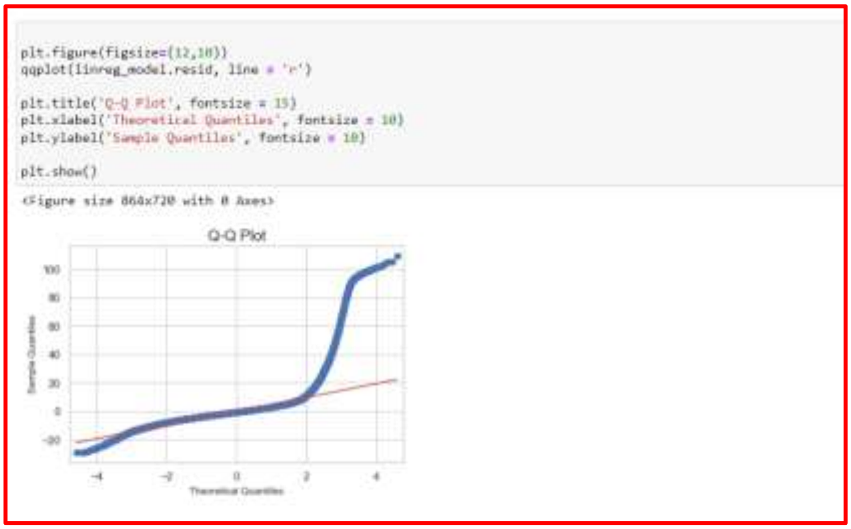
```
p_value: 0.0
```

We observed that the p-value is less than 0.05 thus, we conclude that there is heteroskedasticity present in the data.

5. **Test for Normality**

   A Q-Q plot, short for "quantile-quantile" is used to test for the normality of the residuals.

   Screenshot from the notebook:



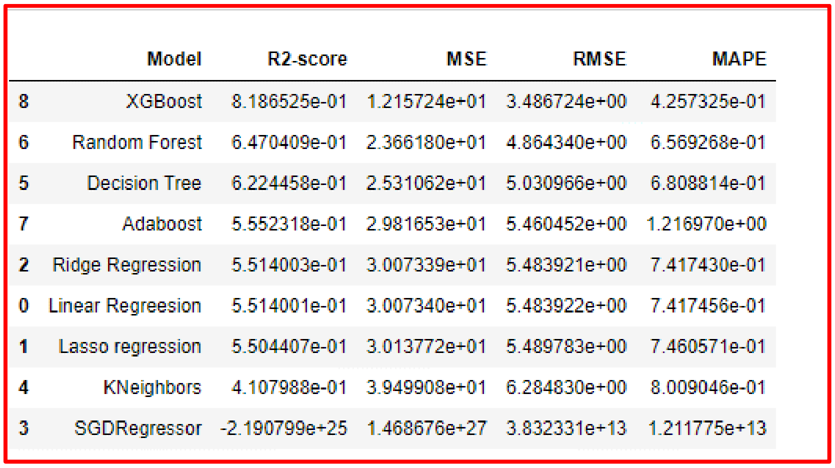
```
plt.figure(figsize=(12,10))
qqplot(linreg_model.resid, line = 'r')

plt.title('Q-Q Plot', fontsize = 15)
plt.xlabel('Theoretical Quantiles', fontsize = 10)
plt.ylabel('Sample Quantiles', fontsize = 10)

plt.show()
```

```
<Figure size 864x720 with 0 Axes>
```

From the above plot diagonal line (red line) is the regression line and the blue points are the cumulative distribution of the residuals. As some of the points are away from the diagonal line, we conclude that the residuals do not follow a normal distribution.

- <u>**Results from Other Models:**</u>

Performed different Regression Algorithms on the same dataset to find best performing model on basis of performance metrics.



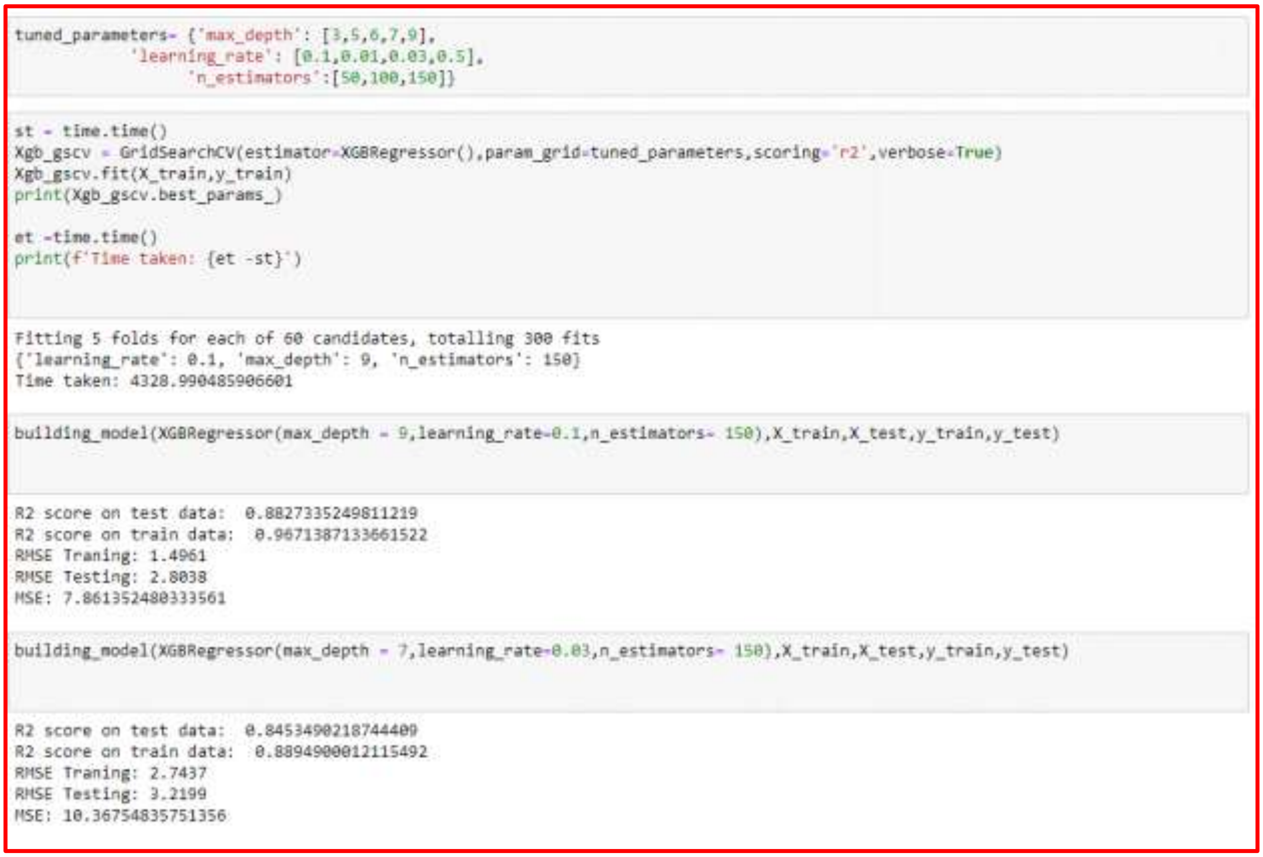
	Model	R2-score	MSE	RMSE	MAPE
8	XGBoost	8.186525e-01	1.215724e+01	3.486724e+00	4.257325e-01
6	Random Forest	6.470409e-01	2.366180e+01	4.864340e+00	6.569268e-01
5	Decision Tree	6.224458e-01	2.531062e+01	5.030966e+00	6.808814e-01
7	Adaboost	5.552318e-01	2.981653e+01	5.460452e+00	1.216970e+00
2	Ridge Regression	5.514003e-01	3.007339e+01	5.483921e+00	7.417430e-01
0	Linear Regreesion	5.514001e-01	3.007340e+01	5.483922e+00	7.417456e-01
1	Lasso regression	5.504407e-01	3.013772e+01	5.489783e+00	7.460571e-01
4	KNeighbors	4.107988e-01	3.949908e+01	6.284830e+00	8.009046e-01
3	SGDRegressor	-2.190799e+25	1.468676e+27	3.832331e+13	1.211775e+13

Based on the provided table, it can be observed that XGBoost and RandomForest algorithms demonstrate superior performance. These algorithms can be further enhanced by implementing hyperparameter tuning techniques.

Hyperparameter Tuning:

Hyperparameter tuning involves adjusting the settings of a machine learning algorithm to optimize its performance. It is a crucial step in improving the accuracy and generalization capabilities of the model. By systematically exploring different combinations of hyperparameters, the best configuration can be identified.

From the above result we performed Hyperparameter tuning to XGBoost Algorithm to find out best parameters on which model performs efficient.



```
tuned_parameters= {'max_depth': [3,5,6,7,9],
            'learning_rate': [0.1,0.01,0.03,0.5],
                  'n_estimators':[50,100,150]}
```

```
st = time.time()
Xgb_gscv = GridSearchCV(estimator=XGBRegressor(),param_grid=tuned_parameters,scoring='r2',verbose=True)
Xgb_gscv.fit(X_train,y_train)
print(Xgb_gscv.best_params_)

et =time.time()
print(f'Time taken: {et -st}')
```

```
Fitting 5 folds for each of 60 candidates, totalling 300 fits
{'learning_rate': 0.1, 'max_depth': 9, 'n_estimators': 150}
Time taken: 4328.990485906601
```

```
building_model(XGBRegressor(max_depth = 9,learning_rate=0.1,n_estimators= 150),X_train,X_test,y_train,y_test)
```

```
R2 score on test data:  0.8827335249811219
R2 score on train data:  0.9671387133661522
RMSE Traning: 1.4961
RMSE Testing: 2.8038
MSE: 7.861352480333561
```

```
building_model(XGBRegressor(max_depth = 7,learning_rate=0.03,n_estimators= 150),X_train,X_test,y_train,y_test)
```

```
R2 score on test data:  0.8453490218744409
R2 score on train data:  0.8894900012115492
RMSE Traning: 2.7437
RMSE Testing: 3.2199
MSE: 10.36754835751356
```

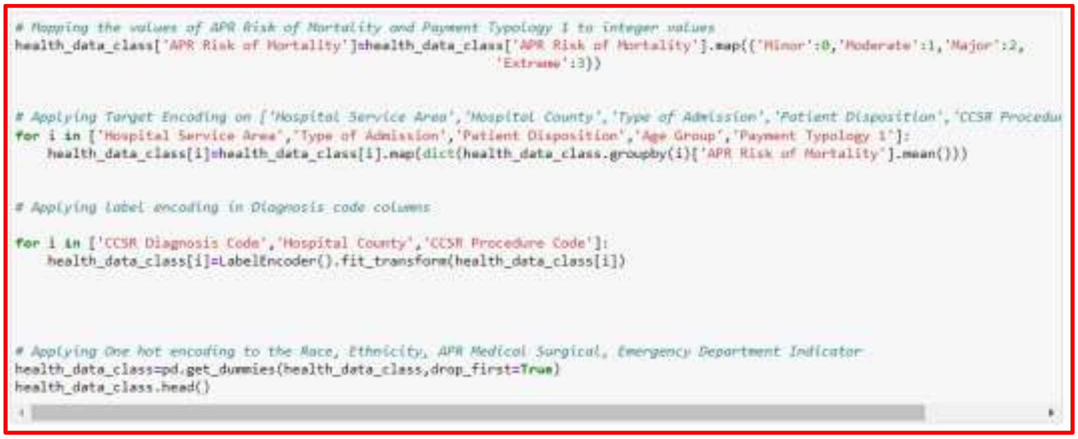
After conducting GridSearchCV, we obtained the best parameters; however, it resulted in overfitting. To address this issue, we made adjustments to the best parameters in order to find an optimal solution with reduced overfitting.

To summarize, we selected XGBoost as the preferred algorithm, achieving an R2 score of 0.845 on the test data and an RMSE of 2.7437.

Classification Problem Statement for Predicting Risk of Mortality

○ Feature Encoding Before Performing Modeling

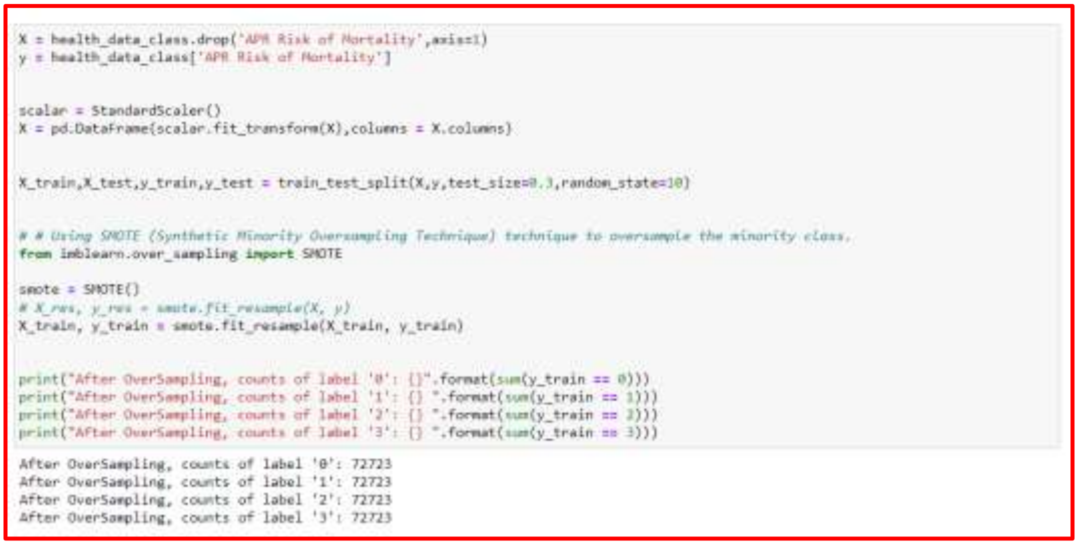
Categorical columns in the dataset were transformed into numerical values using Target Encoding, Label Encoding, and One Hot Encoding techniques. This conversion facilitated the modeling process by enabling the algorithms to work effectively with the categorical data.



```
# Mapping the values of APR Risk of Mortality and Payment Typology 1 to integer values
health_data_class['APR Risk of Mortality']=health_data_class['APR Risk of Mortality'].map({'Minor':0,'Moderate':1,'Major':2,
                                                                                          'Extreme':3})


# Applying Target Encoding on ['Hospital Service Area','Hospital County','Type of Admission','Patient Disposition','CCSR Procedu
for i in ['Hospital Service Area','Type of Admission','Patient Disposition','Age Group','Payment Typology 1']:
    health_data_class[i]=health_data_class[i].map(dict(health_data_class.groupby(i)['APR Risk of Mortality'].mean()))


# Applying label encoding in Diagnosis code columns

for i in ['CCSR Diagnosis Code','Hospital County','CCSR Procedure Code']:
    health_data_class[i]=LabelEncoder().fit_transform(health_data_class[i])




# Applying One hot encoding to the Race, Ethnicity, APR Medical Surgical, Emergency Department Indicator
health_data_class=pd.get_dummies(health_data_class,drop_first=True)
health_data_class.head()
```

○ Performing SMOTE to balance the class

Our target variable `Risk of Mortality` is imbalanced. There are more records in the data for the minor risk class than for all other classes. To overcome this imbalance, we shall perform the `Synthetic Minority Oversampling Technique (SMOTE)` on the data.

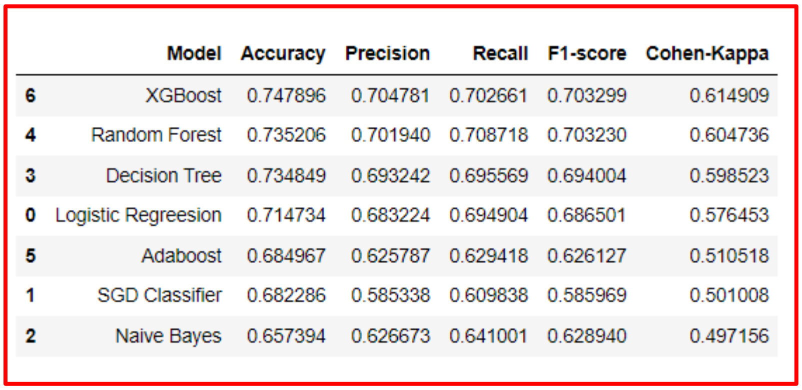


```
X = health_data_class.drop('APR Risk of Mortality',axis=1)
y = health_data_class['APR Risk of Mortality']


scalar = StandardScaler()
X = pd.DataFrame(scalar.fit_transform(X),columns = X.columns)


X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=10)


# # Using SMOTE (Synthetic Minority Oversampling Technique) technique to oversample the minority class.
from imblearn.over_sampling import SMOTE

smote = SMOTE()
# X_res, y_res = smote.fit_resample(X, y)
X_train, y_train = smote.fit_resample(X_train, y_train)


print("After OverSampling, counts of label '0': {}".format(sum(y_train == 0)))
print("After OverSampling, counts of label '1': {} ".format(sum(y_train == 1)))
print("After OverSampling, counts of label '2': {} ".format(sum(y_train == 2)))
print("After OverSampling, counts of label '3': {} ".format(sum(y_train == 3)))
```

```
After OverSampling, counts of label '0': 72723
After OverSampling, counts of label '1': 72723
After OverSampling, counts of label '2': 72723
After OverSampling, counts of label '3': 72723
```

○ Results from Other Models:

Performed different Classification Algorithms on the same dataset to find the best-performing model on the basis of performance metrics.



	Model	Accuracy	Precision	Recall	F1-score	Cohen-Kappa
6	XGBoost	0.747896	0.704781	0.702661	0.703299	0.614909
4	Random Forest	0.735206	0.701940	0.708718	0.703230	0.604736
3	Decision Tree	0.734849	0.693242	0.695569	0.694004	0.598523
0	Logistic Regreesion	0.714734	0.683224	0.694904	0.686501	0.576453
5	Adaboost	0.684967	0.625787	0.629418	0.626127	0.510518
1	SGD Classifier	0.682286	0.585338	0.609838	0.585969	0.501008
2	Naive Bayes	0.657394	0.626673	0.641001	0.628940	0.497156

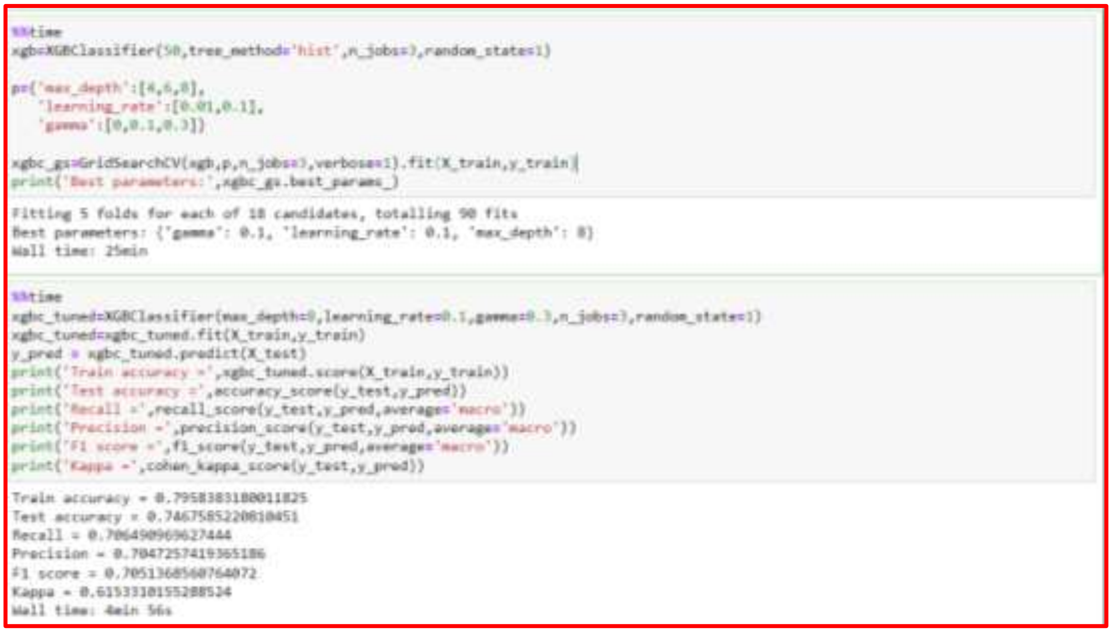
Based on the provided table, it can be observed that XGBoost and RandomForest algorithms demonstrate superior performance. These algorithms can be further enhanced by implementing hyperparameter tuning techniques.

Hyperparameter Tuning:

Hyperparameter tuning involves adjusting the settings of a machine learning algorithm to optimize its performance. It is a crucial step in improving the accuracy and generalization capabilities of the model. By systematically exploring different combinations of hyperparameters, the best configuration can be identified.

From the above result we performed Hyperparameter tuning to XGBoost Algorithm to find out best parameters on which model performs efficient.

After conducting GridSearchCV, we obtained the best parameters, and built the model for the resulted parameters.



```
%%time
xgb=XGBClassifier(50,tree_method='hist',n_jobs=3,random_state=1)

p={'max_depth':[4,6,8],
   'learning_rate':[0.01,0.1],
   'gamma':[0,0.1,0.3]}

xgbc_gs=GridSearchCV(xgb,p,n_jobs=3,verbose=1).fit(X_train,y_train)
print('Best parameters:',xgbc_gs.best_params_)
```

```
Fitting 5 folds for each of 18 candidates, totalling 90 fits
Best parameters: {'gamma': 0.1, 'learning_rate': 0.1, 'max_depth': 8}
Wall time: 25min
```

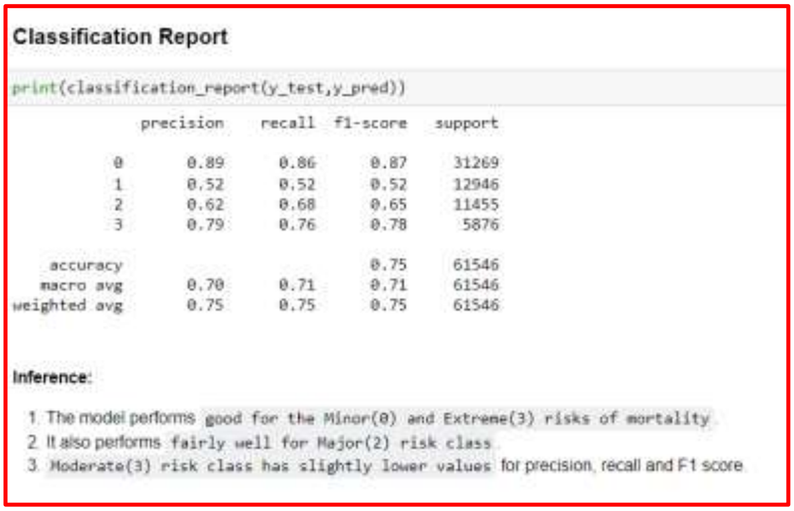
```
%%time
xgbc_tuned=XGBClassifier(max_depth=8,learning_rate=0.1,gamma=0.3,n_jobs=3,random_state=1)
xgbc_tuned=xgbc_tuned.fit(X_train,y_train)
y_pred = xgbc_tuned.predict(X_test)
print('Train accuracy =',xgbc_tuned.score(X_train,y_train))
print('Test accuracy =',accuracy_score(y_test,y_pred))
print('Recall =',recall_score(y_test,y_pred,average='macro'))
print('Precision =',precision_score(y_test,y_pred,average='macro'))
print('F1 score =',f1_score(y_test,y_pred,average='macro'))
print('Kappa =',cohen_kappa_score(y_test,y_pred))
```

```
Train accuracy = 0.7958383180011825
Test accuracy = 0.7467585220810451
Recall = 0.706490969627444
Precision = 0.7047257419365186
F1 score = 0.7051368560764072
Kappa = 0.6153310155288524
Wall time: 4min 56s
```

To summarize, we selected XGBoost as the preferred algorithm which summarizes the points:

a. The overall accuracies for training and testing data are close which suggests that there is no major overfitting of the model.
b. The accuracy of the testing data suggests that the model correctly classifies about 75% of records of unseen data.
c. The recall tells us that the model correctly classifies an average of about 71% of true classes.
d. The precision suggests that the model correctly classifies on an average 70% of classes.
e. A kappa of 0.614 tells us that there is more than moderate agreement between actual and predicted labels.

Screenshot from the notebook:



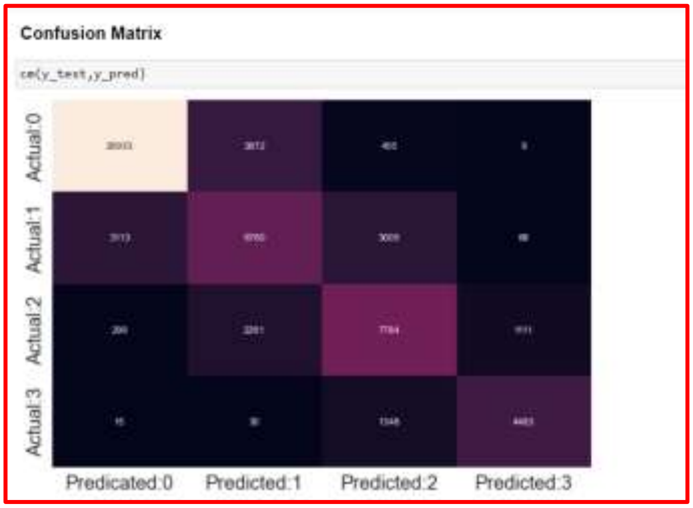
Classification Report

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.89      0.86      0.87     31269
           1       0.52      0.52      0.52     12946
           2       0.62      0.68      0.65     11455
           3       0.79      0.76      0.78      5876

    accuracy                           0.75     61546
   macro avg       0.70      0.71      0.71     61546
weighted avg       0.75      0.75      0.75     61546
```

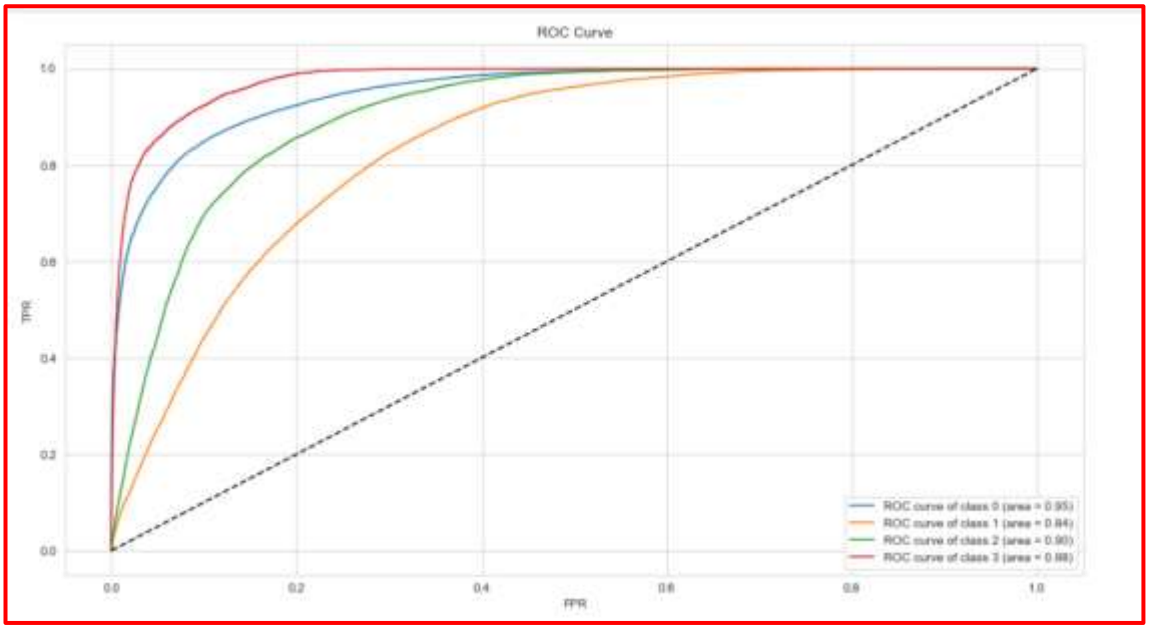
Inference:

1. The model performs good for the Minor(0) and Extreme(3) risks of mortality
2. It also performs fairly well for Major(2) risk class
3. Moderate(3) risk class has slightly lower values for precision, recall and F1 score



Confusion Matrix

```
cm(y_test,y_pred)
```



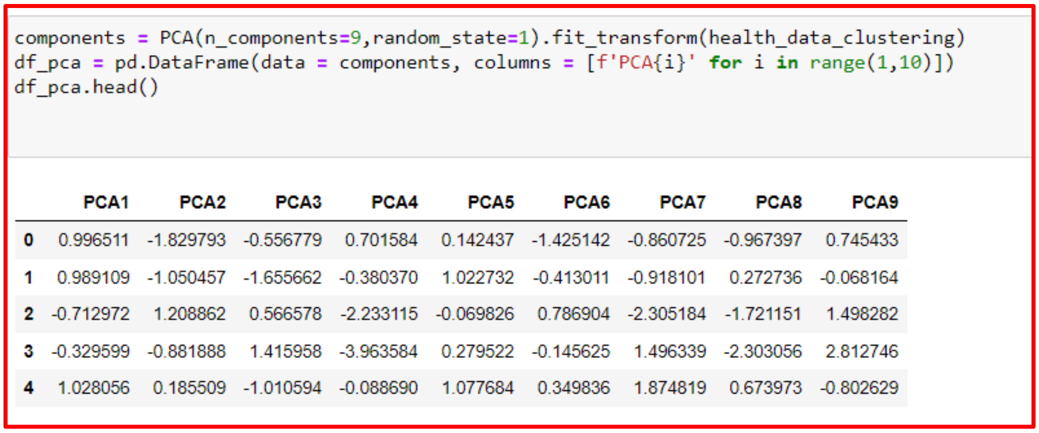
Inferences:

- 26977 observations in the Minor (0) class are correct while 4292 are wrongly classified.
- 6687 observations in the Moderate (1) class are correct while 6259 are wrongly classified.
- 7791 observations in the Major (2) class are correct while 3664 are wrongly classified.
- 4475 observations in the Extreme (3) class are correct while 1401 are wrongly classified.
- There is a good separation for all the different classes.

Clustering Problem Statement for Demographic Variations on Diagnosis

○ Performing PCA

Principal Component Analysis (PCA) is a dimensionality reduction technique used to extract relevant information from high-dimensional data. It transforms the data into a new coordinate system by identifying the directions of maximum variance, called principal components. By retaining the most significant principal components, PCA helps to simplify the dataset while preserving its key features.
The new dataset has 205152 records and 9 columns, i.e., we have decreased the number of features from 25 to 9 which explains more than 70% variation.



```
components = PCA(n_components=9,random_state=1).fit_transform(health_data_clustering)
df_pca = pd.DataFrame(data = components, columns = [f'PCA{i}' for i in range(1,10)])
df_pca.head()
```

	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6	PCA7	PCA8	PCA9
0	0.996511	-1.829793	-0.556779	0.701584	0.142437	-1.425142	-0.860725	-0.967397	0.745433
1	0.989109	-1.050457	-1.655662	-0.380370	1.022732	-0.413011	-0.918101	0.272736	-0.068164
2	-0.712972	1.208862	0.566578	-2.233115	-0.069826	0.786904	-2.305184	-1.721151	1.498282
3	-0.329599	-0.881888	1.415958	-3.963584	0.279522	-0.145625	1.496339	-2.303056	2.812746
4	1.028056	0.185509	-1.010594	-0.088690	1.077684	0.349836	1.874819	0.673973	-0.802629

○ K- Means Clustering

K-means clustering is an unsupervised machine learning algorithm used to partition data into distinct groups based on similarity. It iteratively assigns data points to clusters and updates the cluster centroids until convergence, aiming to minimize the within-cluster sum of squared distances. The result is a set of k clusters, where each data point belongs to the cluster with the nearest centroid.

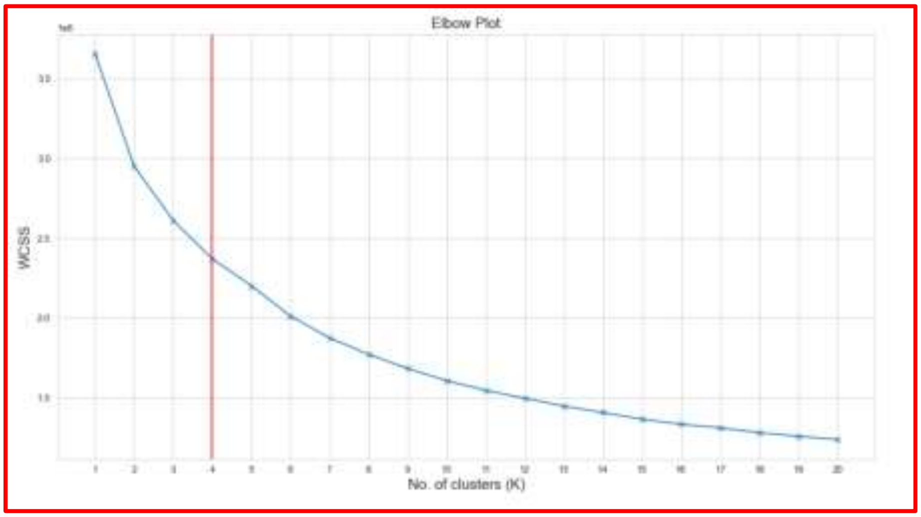
We consider two techniques (elbow/scree plot and Silhouette score) to decide the optimal value of K to perform the K-means clustering.

○ Elbow Method/ Scree Method

The elbow method is a technique used to determine the optimal number of clusters in a dataset for clustering algorithms.

It involves plotting the within-cluster sum of squared distances against the number of clusters and identifying the "elbow" point, which signifies the optimal number of clusters.

The elbow point represents a balance between reducing the within-cluster distance and avoiding excessive clustering complexity.



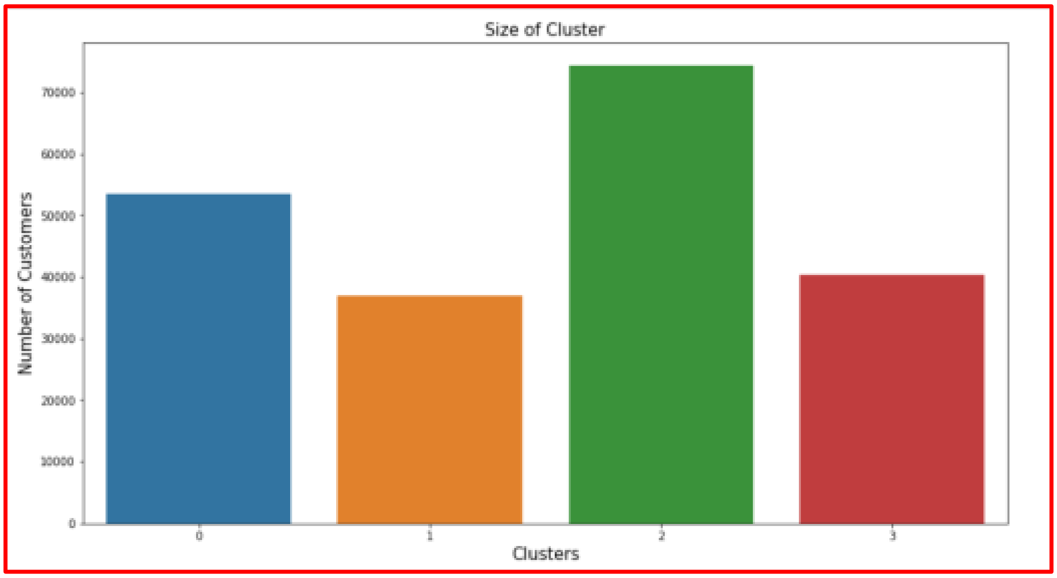
○ **Silhouette score**

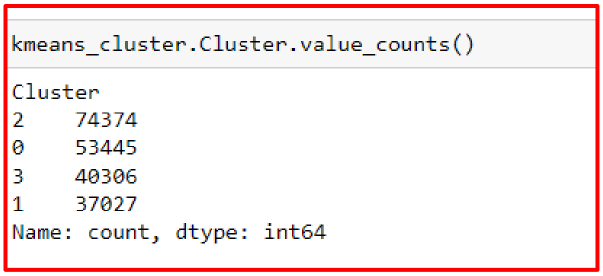
The silhouette score is a metric used to evaluate the quality of clustering results. It measures the compactness and separation of clusters by calculating the average silhouette coefficient for each data point.
Also, from the outputs of the Elbow & Silhouette Score method, we saw that the silhouette score is maximum for k = 4. Thus, we chose K = 4 as the optimal value of k.

○ **K- Means Model:**

Performed KMeans clustering model for 4 clusters on the dataset to find the best-performing model on the basis of performance metrics.

Following the execution of KMeans, we observed the distribution of cluster sizes.

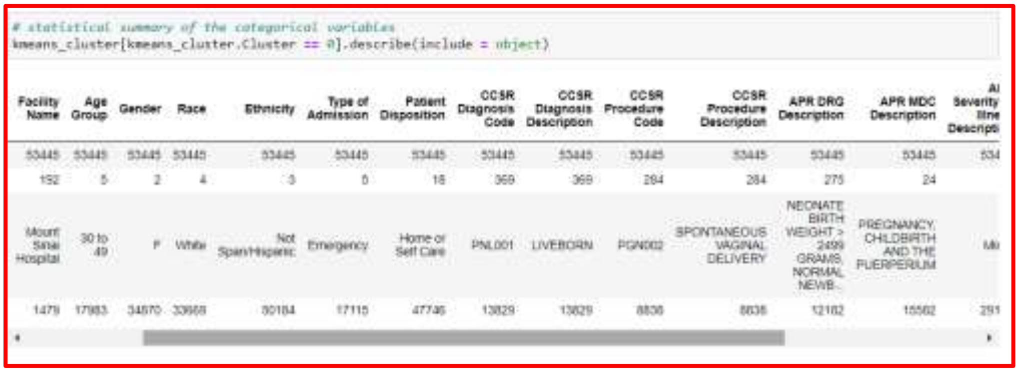




```
kmeans_cluster.Cluster.value_counts()

Cluster
2    74374
0    53445
3    40306
1    37027
Name: count, dtype: int64
```

Clusters Summary:

- **Cluster 1 Summary**



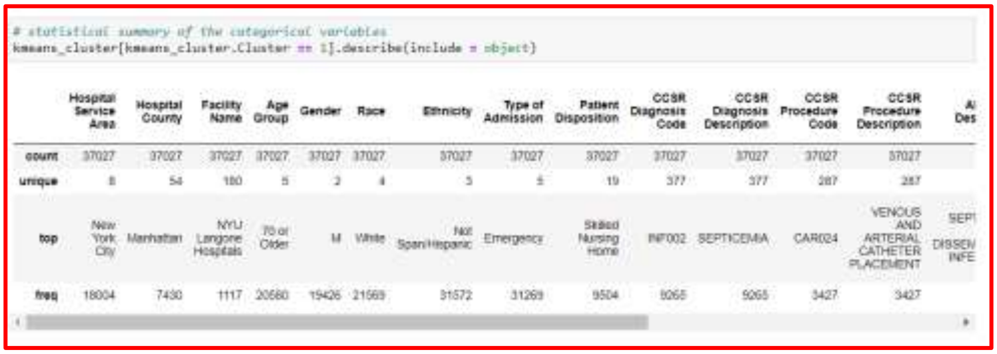
```
# statistical summary of the categorical variables
kmeans_cluster[kmeans_cluster.Cluster == 0].describe(include = object)
```

Facility Name	Age Group	Gender	Race	Ethnicity	Type of Admission	Patient Disposition	CCSR Diagnosis Code	CCSR Diagnosis Description	CCSR Procedure Code	CCSR Procedure Description	APR DRG Description	APR MDC Description	Al Severity Illne Descripti
53445	53445	53445	53445	53445	53445	53445	53445	53445	53445	53445	53445	53445	534
152	5	2	4	3	5	18	369	369	284	284	275	24	
Mount Sinai Hospital	30 to 49	F	White	Not Span/Hispanic	Emergency	Home or Self Care	PNL001	LIVEBORN	PGN002	SPONTANEOUS VAGINAL DELIVERY	NEONATE BIRTH WEIGHT > 2499 GRAMS, NORMAL NEWB...	PREGNANCY, CHILDBIRTH AND THE PUERPERIUM	Mi
1479	17983	34670	33668	50184	17115	47746	13829	13829	8836	8836	12182	15562	291

The above summary shows that the average Length of Stay in this cluster is approximately 4. On average, their total charges are 33187.66 dollars, and belong to the age group 30 to 49 years. Approximately 65% of the patients are female and 62% are white 93% are Not Span/Hispanic with frequent diagnoses of Liveborn which needs Home or Self-care after getting discharged.

Here, the cluster reveals valuable insights into the characteristics and trends of this subgroup, contributing to a deeper understanding of factors affecting **livebirth** outcomes with minor risk of mortality.

- **Cluster 2 Summary**

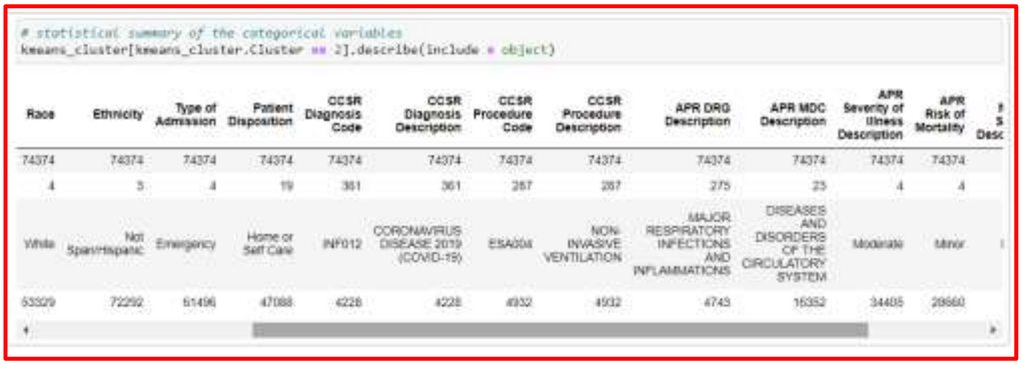


```
# statistical summary of the categorical variables
kmeans_cluster[kmeans_cluster.Cluster == 1].describe(include = object)
```

	Hospital Service Area	Hospital County	Facility Name	Age Group	Gender	Race	Ethnicity	Type of Admission	Patient Disposition	CCSR Diagnosis Code	CCSR Diagnosis Description	CCSR Procedure Code	CCSR Procedure Description	A Des
count	37027	37027	37027	37027	37027	37027	37027	37027	37027	37027	37027	37027	37027	
unique	8	54	180	5	2	4	3	5	19	377	377	287	287	
top	New York City	Manhattan	NYU Langone Hospitals	70 or Older	M	White	Not Span/Hispanic	Emergency	Skilled Nursing Home	INF002	SEPTICEMIA	CAR024	VENOUS AND ARTERIAL CATHETER PLACEMENT	SEPI DISSEM INFE
freq	18004	7430	1117	20680	19426	21569	31572	31269	9504	9265	9265	3427	3427	

The above summary shows that the average Length of Stay in this cluster is approximately 14. On average, their total charges are 198686 dollars, and belong to the age group 70 or Older. Approximately 52% of the patients are male with frequent diagnoses of SEPTICEMIA which needs Skilled Nursing Home care after getting discharged. Here, the cluster reveals valuable insights into the characteristics and trends of this subgroup, contributing to a deeper understanding of factors affecting a serious bloodstream infection with Extreme Risk of Mortality

- **Cluster 3 Summary**



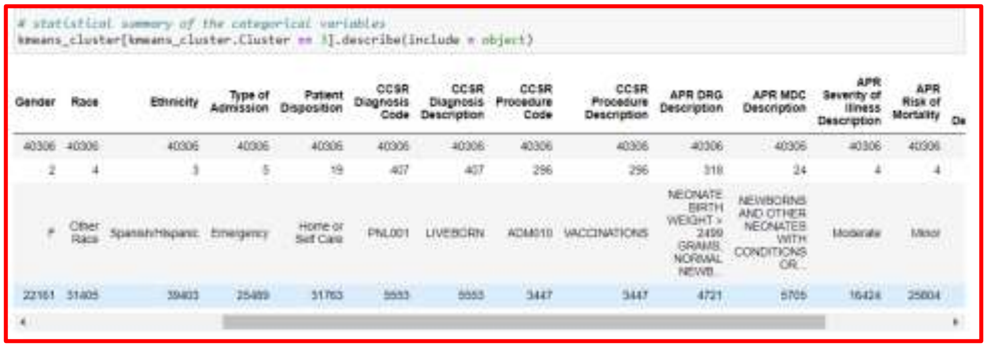
```
# statistical summary of the categorical variables
kmeans_cluster[kmeans_cluster.Cluster == 2].describe(include = object)
```

Race	Ethnicity	Type of Admission	Patient Disposition	CCSR Diagnosis Code	CCSR Diagnosis Description	CCSR Procedure Code	CCSR Procedure Description	APR DRG Description	APR MDC Description	APR Severity of Illness Description	APR Risk of Mortality	F S Desc
74374	74374	74374	74374	74374	74374	74374	74374	74374	74374	74374	74374	
4	3	4	19	381	361	287	287	275	23	4	4	
White	Not Span/Hispanic	Emergency	Home or Self Care	INF012	CORONAVIRUS DISEASE 2019 (COVID-19)	ESA004	NON-INVASIVE VENTILATION	MAJOR RESPIRATORY INFECTIONS AND INFLAMMATIONS	DISEASES AND DISORDERS OF THE CIRCULATORY SYSTEM	Moderate	Minor	I
53329	72292	51496	47088	4228	4228	4932	4932	4743	16352	34405	28660	

The above summary shows that the average Length of Stay in this cluster is approximately 4 and consists of the highest number of observations. On average, their total charges are 49930 dollars and belong to the age group 70 or Older. Approximately 71% of the patients are white and 97% are Not Span/Hispanic with frequent diagnoses of CORONAVIRUS DISEASE 2019 (COVID-19) which needs Home or Self Care or quarantine after getting discharged.

Here, the cluster yields valuable insights into the distinctive attributes and patterns of this subgroup, leading to a deep understanding of the factors that impact COVID disease outcomes with Minor to Moderate Risk of Mortality.

- **<u>Cluster 4 Summary</u>**



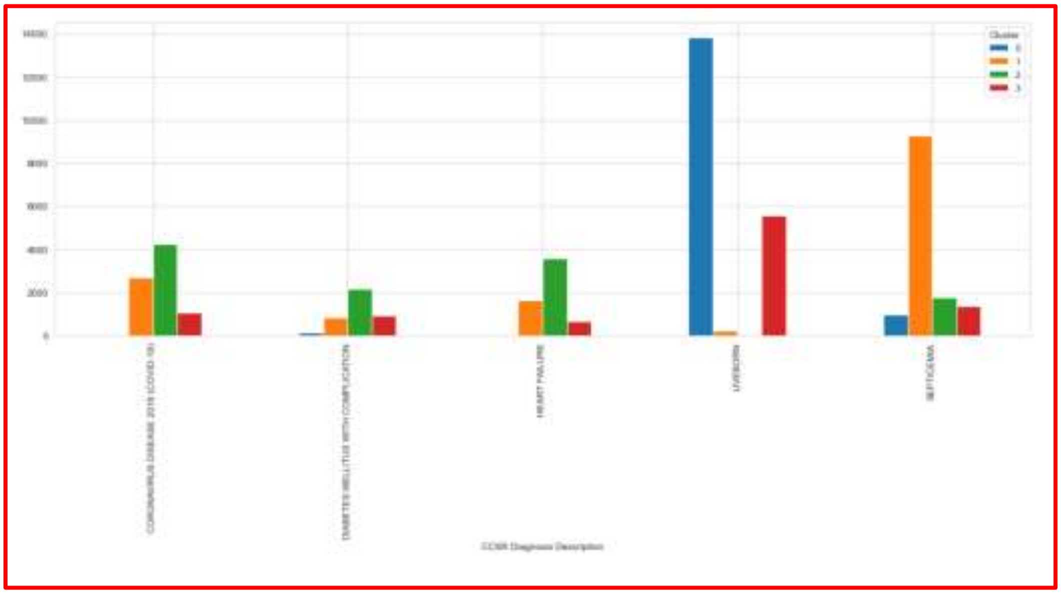
```
# statistical summary of the categorical variables
kmeans_cluster[kmeans_cluster.Cluster == 3].describe(include = object)
```

Gender	Race	Ethnicity	Type of Admission	Patient Disposition	CCSR Diagnosis Code	CCSR Diagnosis Description	CCSR Procedure Code	CCSR Procedure Description	APR DRG Description	APR MDC Description	APR Severity of Illness Description	APR Risk of Mortality	De
40306	40306	40306	40306	40306	40306	40306	40306	40306	40306	40306	40306	40306	
2	4	3	5	19	407	407	296	296	318	24	4	4	
F	Other Race	Spanish/Hispanic	Emergency	Home or Self Care	PNL001	LIVEBORN	ADM010	VACCINATIONS	NEONATE BIRTH WEIGHT > 2499 GRAMS, NORMAL NEWB...	NEWBORNS AND OTHER NEONATES WITH CONDITIONS OR...	Moderate	Minor	
22161	31405	39403	25489	31763	5553	5553	3447	3447	4721	5705	16424	25604	

The above summary shows that the average Length of Stay in this cluster is approximately 4. On average, their total charges are 53936 dollars and belong to the age group 30 to 49. Approximately 77% of the patients are of other races and 97% are Spain/Hispanic with frequent diagnoses of Liveborn as Emergency which needs Home or Self Care after getting discharged.

Here, the cluster yields valuable insights into the distinctive attributes and patterns of this subgroup, leading to a deep understanding of the factors that impact livebirth outcomes with Minor to Moderate Risk of Mortality.

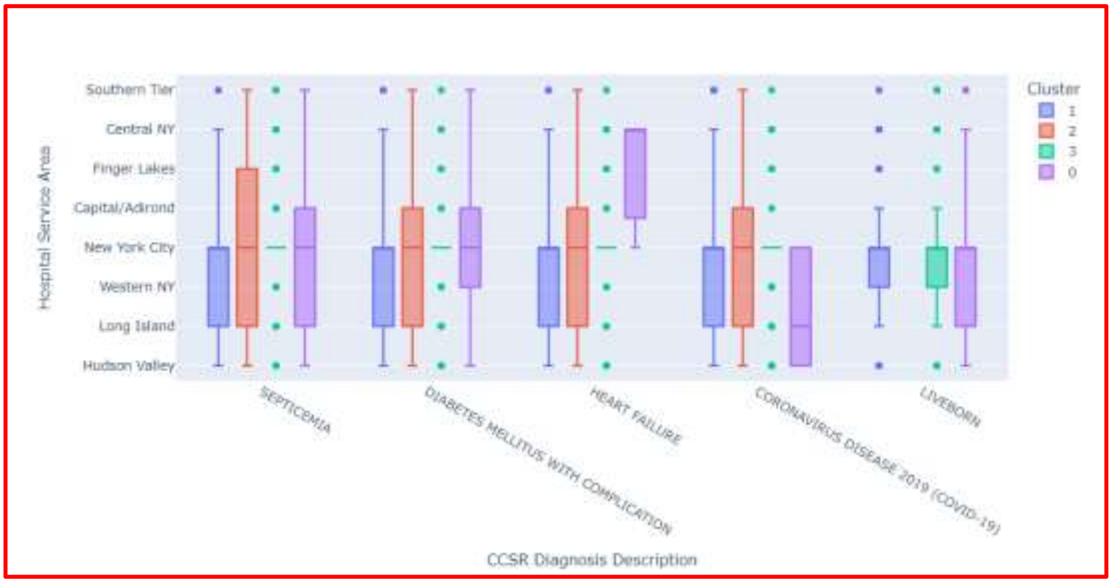
<u>Analysis on Top five Diagnosis for the dataset</u>



From the above visualization and clusters summary, we can formulate the conclusion:

- Cluster 1 = Cluster belongs to the subgroup, that impact livebirth outcomes for Not Spanish/Hispanic ethnicity.
- Cluster 2 = Cluster belongs to the subgroup, that impacts Septicemia outcomes.
- Cluster 3 = Cluster belongs to the subgroup, that impact Covid disease outcomes.
- Cluster 4 = Cluster belongs to the subgroup, that impact livebirth outcomes for Spanish/Hispanic ethnicity.

Analysis of Diagnosis on a Regional basis



From the above plot we can see Maximum Diagnosis Observation for the particular region i.e.

- Most Liveborn cases in Central NY.
- Most Septicemia cases in Southern Tier.
- People of Southern Tier are more prone to have COVID-19.

Implications

Prediction of Length of Stay

- The length of stay of a patient is of great significance in the case of healthcare. Several complexities can arise if the length of stay of patients is not given importance. The longer an individual occupies a bed in your acute facility, the longer that person will require clinical attention from staff whose time could be better spent elsewhere. The patient's experience can be severely compromised if the individual must wait in a bed for longer than is clinically necessary.
- With good level of confidence, we can say that children admitted aged below 18 would usually stay for less no. of days. Newborn admissions usually take a stay of 2 days. Hospitals can allocate resources beforehand for patients suffering from severe illnesses or having higher risks of mortality as they can be expected to stay longer. Similarly, patients admitted in an emergency will stay longer so rooms can be separately kept aside.

Prediction of Risk of Mortality

- In the healthcare sector, mortality risk prediction helps medical professionals make informed decisions about patient care and treatment plans. By identifying patients who are at a higher risk of mortality, healthcare providers can prioritize resources, interventions, and preventive measures to improve patient outcomes. This knowledge allows for early intervention, timely medical interventions, and tailored treatment strategies.
- Based on insights, we can recommend that hospitals should focus on providing specialized care for older patients, particularly those aged 70 and above, as they have a higher risk of mortality. Hospitals should implement risk assessment protocols that take into account the different types of admission. Patients with 'Medicare' payment typology have been identified as having higher risks of mortality. This may indicate the need for enhanced care coordination, chronic disease management, and proactive interventions for patients covered by Medicare. Hospitals should prioritize the implementation of strategies to reduce mortality risks for patients admitted in emergency situations.

Identifying Demographic Variations

- Identifying regions where additional healthcare coverage is needed is crucial for improving healthcare access and equity. By identifying areas with inadequate healthcare coverage, policymakers, healthcare providers, and public health agencies can strategically allocate resources to address the gaps and ensure that underserved populations receive the necessary care. This can involve establishing new healthcare facilities, expanding existing infrastructure, recruiting healthcare professionals, and implementing targeted healthcare programs and interventions.
- Manhattan County and New York City hospitals should collaborate with local health authorities, community organizations, and neighbouring healthcare facilities to address the higher risks of mortality collectively. Given the higher lengths of stay and severity of illness, hospitals should focus on optimizing capacity management and resource allocation. With the majority of admissions being emergency cases, hospitals should prioritize improvements in emergency care services. Given the prevalence of septicaemia diagnoses, hospitals should implement robust sepsis management protocols. Hospitals should prioritize patient safety initiatives to mitigate risks of mortality. This can include implementing quality improvement programs, enhancing infection prevention and control measures, and promoting a culture of safety among healthcare staff.

Limitations

Prediction of Length of Stay

- Regression assumes independence of attributes. We have seen through variation inflation factor (VIF) of the features in our base model that the features were not truly independent. In fact, in real life, it is very rare when the predictor variables are independent. We have seen through Breusch-Pagan Test that we have heteroscedasticity present in our data which can affect the efficiency and validity of our regression estimates. Regression models are generally not reliable for making predictions outside the range of the observed data. So, if we have testing data which pertains to a higher length of stay than captured in the training set, our model may provide inaccurate results.
- To enhance our predictions, we can use robust regression. In the presence of heteroscedasticity, robust regression techniques, such as weighted least squares or robust standard errors, can provide more reliable coefficient estimates and standard errors that account for the unequal variance of the errors.

Prediction of Risk of Mortality

- In the classification report for our final model, we saw that the Moderate Risk of Mortality class had lower values for precision, recall, and F1 score as well. This would suggest that our model does not perform as well in classifying the records of the moderate class. This means that our model may classify enough records actually belonging to the moderate class as a minor class, which can lead to delayed treatment and harm to patients.
- To overcome this, we can try to find out the cause of the low performance through EDA on the data and successively work to solve the issue.

Identifying Demographic Variations

- We have used K-means algorithm for identifying the demographic variations which assumes that the clusters are globular in shape. This is not always true for data in real life. It also assumes that the densities of clusters are roughly similar. In a data with clusters of varying densities, it may provide suboptimal results. It also forms different clusters when different initial centroid clusters are chosen. This may lead to different results and interpretations for the same analysis.

- We can use the DBSCAN algorithm to enhance the performance of our clustering. It is not only capable of forming clusters of any size and shape but also is insensitive to initial starting position of the algorithm.

Comparison to Benchmark & Takeaways

In comparison to the benchmark laid out at the outset, the final solution has shown improvements in various ways.

For the regression problem of predicting length of stay, the final model achieved a lower mean squared error (MSE) compared to the benchmark. This indicates that the model's predictions were closer to the actual length of stay, suggesting improved accuracy.

In the classification problem of predicting risk of mortality, the final model achieved a higher accuracy and a more balanced distribution across risk categories compared to the benchmark. This suggests that the model's predictions were more reliable and better captured the varying levels of mortality risk.

For the clustering problem of demographic area for diagnosis, the final clustering approach produced more distinct and cohesive clusters compared to the benchmark. This indicates that the final solution effectively grouped demographic areas based on relevant factors, providing more meaningful insights into healthcare needs across different regions.

These improvements could be attributed to various factors, including better feature selection, refinement of algorithms, and fine-tuning of hyperparameters. Additionally, incorporating more comprehensive and diverse datasets, as well as implementing advanced techniques such as ensemble methods or feature engineering, may have contributed to improved performance.

It is important to note that the specific improvements and comparisons would depend on the details of the benchmark and the evaluation metrics used.

From the process, we have learned the importance of considering the specific problem types and selecting appropriate algorithms accordingly. Building regression models for predicting length of stay requires careful feature selection and consideration of factors that impact the duration of hospital stays. Classification models for risk of mortality should take into account relevant medical history and vital signs to accurately classify patients into risk categories. Clustering demographic areas for diagnosis requires analyzing factors such as population demographics and healthcare accessibility.

In the future, we would focus on ensuring the availability and quality of relevant data to train the models effectively. Additionally, we would explore different feature engineering techniques to enhance the predictive power of the models. It would also be beneficial to evaluate various clustering algorithms and criteria to identify the most suitable approach for capturing demographic patterns in relation to diagnoses. Continuous learning and refining the models based on feedback and new insights would also be a valuable approach.