HEALTH CARE ANALYTICS

By Khaja Moiezuddin Mohammed

INFO 5082

INTRODUCTION:

A 2015 Commonwealth Fund brief showed that the United States had worse outcomes and spent more on health care, largely because of greater use of medical technology and higher prices, compared to other high-income countries. The U.S. spends more on health care as a share of the economy. It spends nearly twice as much as the average developed country and yet has the lowest life expectancy and highest suicide rates among the 11 nations.

Poor management of health care resources is one of the leading factors of increased death rates and increased expenses among the patients in US.

As many hospitals use very expensive technologies, if they know the health analytics, then they can allocate their resources perfectly and it can save many lives and cut many unnecessary expenses.

PROBLEM STATEMENT:

Recent Covid-19 Pandemic has raised alarms over one of the most overlooked area to focus: Healthcare Management. While healthcare management has various use cases for using data science, patient length of stay is one critical parameter to observe and predict if one wants to improve the efficiency of the healthcare management in a hospital.

This parameter helps hospitals to identify patients of high LOS risk (patients who will stay longer) at the time of admission. Once identified, patients with high LOS risk can have their treatment plan optimized to minimize LOS (length of stay) and lower the chance of staff/visitor infection. Also, prior knowledge of LOS can aid in logistics such as room and bed allocation planning.

The goal is to accurately predict the Length of Stay for each patient on case by case basis so that the Hospitals can use this information for optimal resource allocation and better functioning. The major goal is to predict the outcomes perfectly and to save lives.

ABOUT DATASET:

This dataset is taken from Kaggle.com and this dataset was uploaded Analytics vidya. It has a total of 18 columns and more 1,30,000 rows of data to it.

This dataset provides important information like “CASEID” which gives unique id to every patient in the hospital. This dataset provides information like “Hospital code” which is a unique code given to a hospital belonging to a specific area. It also provides a column called “department” which relates to different departments a patient will be admitted. Some other important details like age, severity of illness and ward type are also provided. All the column details are given below,

1: case_id: Case_ID registered in Hospital (int)
2: Hospital_code: Unique code for the Hospital (int)
3: Hospital_type_code: Unique code for the type of Hospital (object)
4: City_Code_Hospital: City Code of the Hospital (int)

5: Hospital_region_code: Region Code of the Hospital (object)
6: Available Extra Rooms in Hospital: Number of Extra rooms available in the Hospital
7: Department: Department overlooking the case (object)
8: Ward_Type: Code for the Ward type (object)
9: Ward_Facility_Code: Code for the Ward Facility (object)
10: Bed Grade: Condition of Bed in the Ward (float64)
11: patientid: Unique Patient Id (int)
12: City_Code_Patient: City Code for the patient (float64)
13: Type of Admission: Admission Type registered by the Hospital (object)
14: Severity of Illness: Severity of the illness recorded at the time of admission (object)
16: Visitors with Patient: Number of Visitors with the patient (int)
17: Age: Age of the patient (int)
18: Admission_Deposit: Deposit at the Admission Time (float64)
19: Stay: Stay Days by the patient (object)

As the column 'Stay' and 'Age' values are in ranges, two new columns were created to replace the range values with integer values.
The two new columns are **"new_stay"** and **"new_age".**

OBJECTIVES:

- Health analytics (with the given data, look for the trends like what kind of cases are registered daily in a particular area)
- Here we have dependent variables like Age, severity of illness. These are the factors responsible for higher staying period in the hospital. if we look for correlation between the major factors and perform hypothesis then we can achieve desired objectives.
- Creating different visualization to understand the trend.
- Applying feature engineering methods and to build a regression model to predict the duration of stay of a patient.
- The main objective of this project is to build a model which can predict the length of stay of a patient, so that the hospital can manage and allocate its resources accordingly, this can help to save many lives.

Research design and methodology:

- Breaking down the steps required for this project, I worked on data collection and data cleaning. I have used different tools like Excel and Python to clean the data and remove all the null and unwanted data.
- Another important step to perform is to draw meaningful insights from the data and to do visualizations. To perform visualizations, I have used tools like Tableau, Python.
- For creating a model to predict the outcomes I am using approaches like multivariate linear Regression Integrated and other suitable Machine Learning algorithms. To perform these tasks, we are going to use SAS studio and Python.

Data Cleaning:

The first step of data preprocessing is data cleaning. I performed some basic data cleaning steps like filling the null values and replaced it with appropriate values.

```
data.head(10)
```

	case_id	Hospital_code	Hospital_type_code	City_Code_Hospital	Hospital_region_code	Available Extra Rooms in Hospital	Department	Ward_Type	Ward_Facility_Code	Bed Grade
0	1	8	c	3	Z	3	radiotherapy	R	F	2.0
1	2	2	c	5	Z	2	radiotherapy	S	F	2.0
2	3	10	e	1	X	2	anesthesia	S	E	2.0
3	4	26	b	2	Y	2	radiotherapy	R	D	2.0
4	5	26	b	2	Y	2	radiotherapy	S	D	2.0

As the columns “Stay” and “Age” are of object types, I have replaced those with columns with integer values. Below is the code which I used to perform the conversion

```
age_lst = data["Age"].unique()

age_lst.sort()
age_dict = dict(zip(age_lst, range(len(age_lst))))
data["new_age"]=data["Age"].replace(age_dict)
print(age_dict)
```

```
{'0-10': 0, '11-20': 1, '21-30': 2, '31-40': 3, '41-50': 4, '51-60': 5, '61-70': 6, '71-80': 7, '81-90': 8, '91-100': 9}
```

```
stay_list = data["Stay"].unique()
stay_list.sort()
dept_Stay = dict(zip(stay_list, range(len(stay_list))))
data["new_stay"]= data["Stay"].replace(dept_Stay)
print(dept_Stay)
```

```
{'0-10': 0, '11-20': 1, '21-30': 2, '31-40': 3, '41-50': 4, '51-60': 5, '61-70': 6, '71-80': 7, '81-90': 8, '91-100': 9, 'More than 100 Days': 10}
```

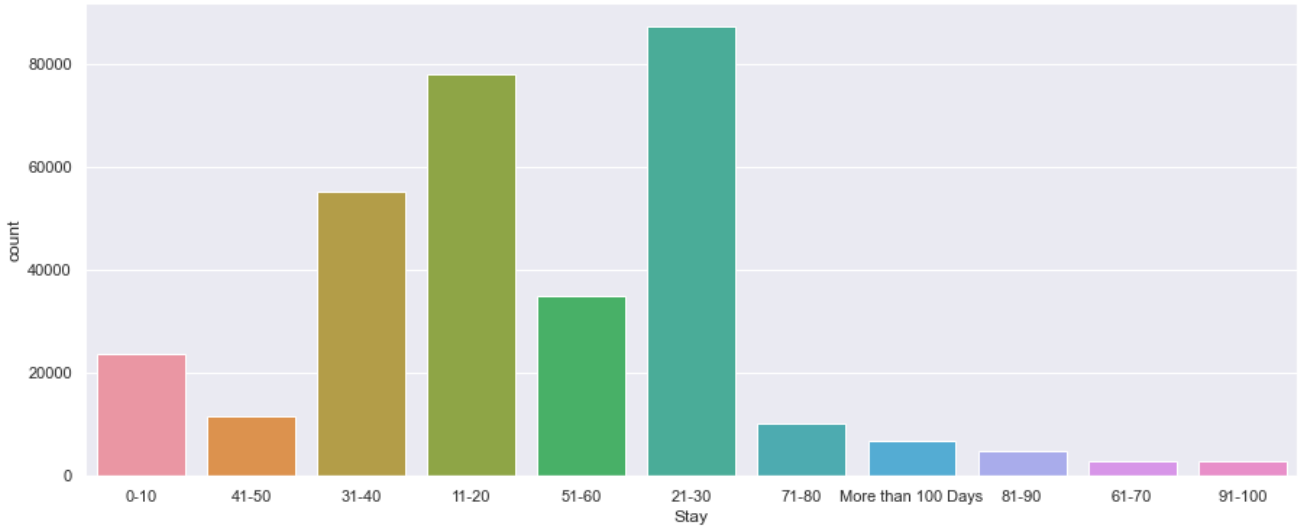
I replaced the converted data into new columns “new_stay” and “new_age”. The updated data is shown below

```
data.head()
```

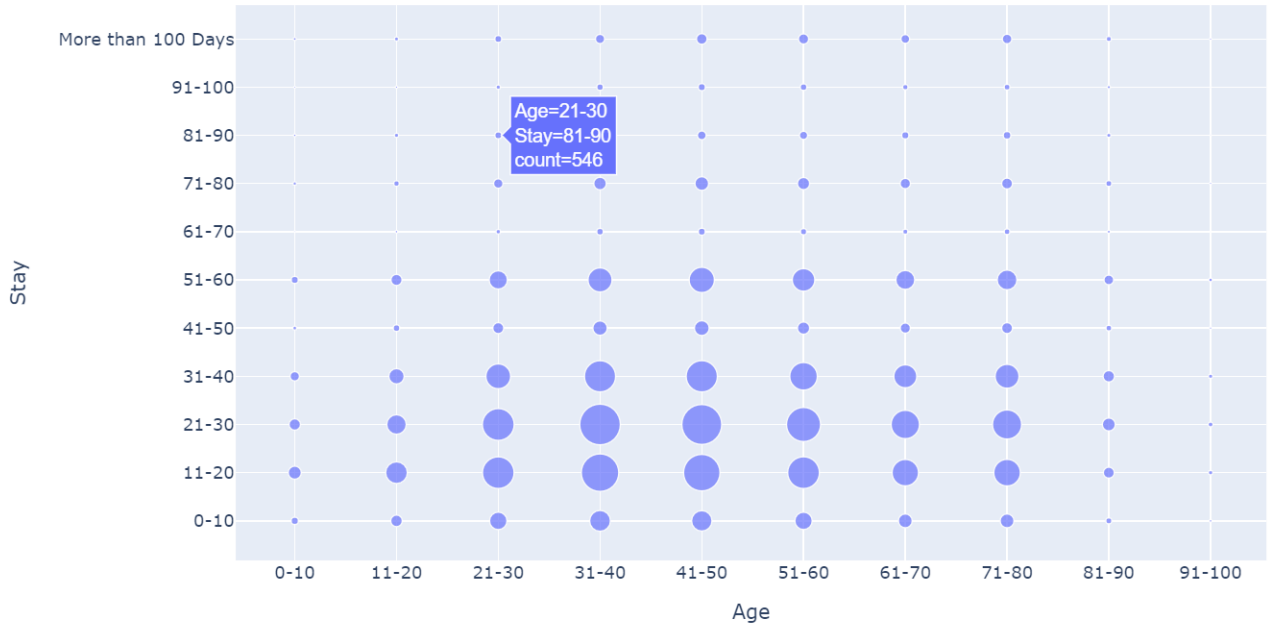
nt	Ward_Type	Ward_Facility_Code	Bed Grade	patientid	City_Code_Patient	Type of Admission	Severity of Illness	Visitors with Patient	Age	Admission_Deposit	Stay	new_age	new_stay
py	R	F	2.0	31397	7.0	Emergency	Extreme	2	51-60	4911.0	0-10	5	0
py	S	F	2.0	31397	7.0	Trauma	Extreme	2	51-60	5954.0	41-50	5	4
ia	S	E	2.0	31397	7.0	Trauma	Extreme	2	51-60	4745.0	31-40	5	3
py	R	D	2.0	31397	7.0	Trauma	Extreme	2	51-60	7272.0	41-50	5	4

Exploratory Data Analysis:

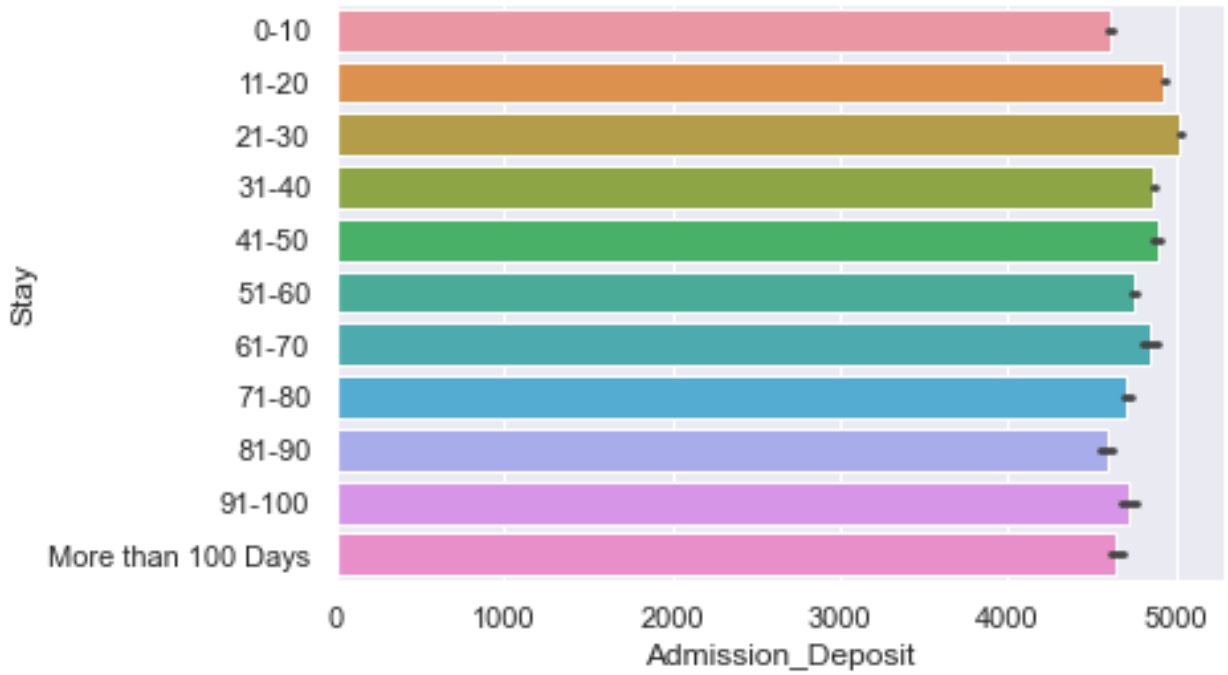
I have plotted many graphs to understand the trend of the data. These visualizations are shown below



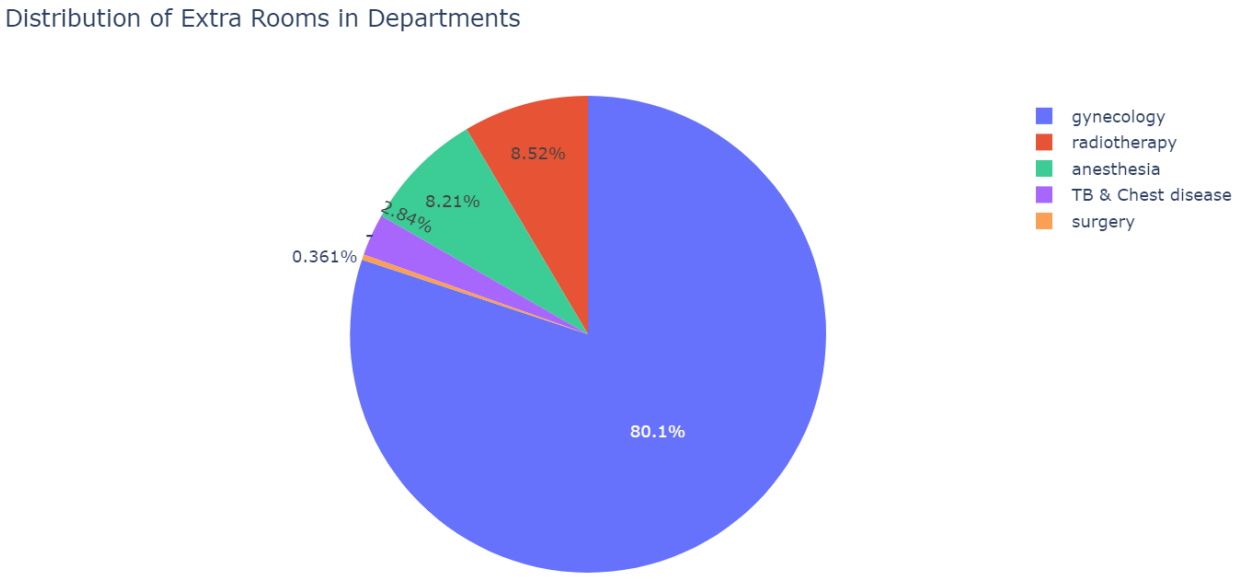
By looking at the above bar graph it is clear that patients usually stay till 20-30 days for their treatment and this bar also gives a count of how many patients stayed for a particular period of time.



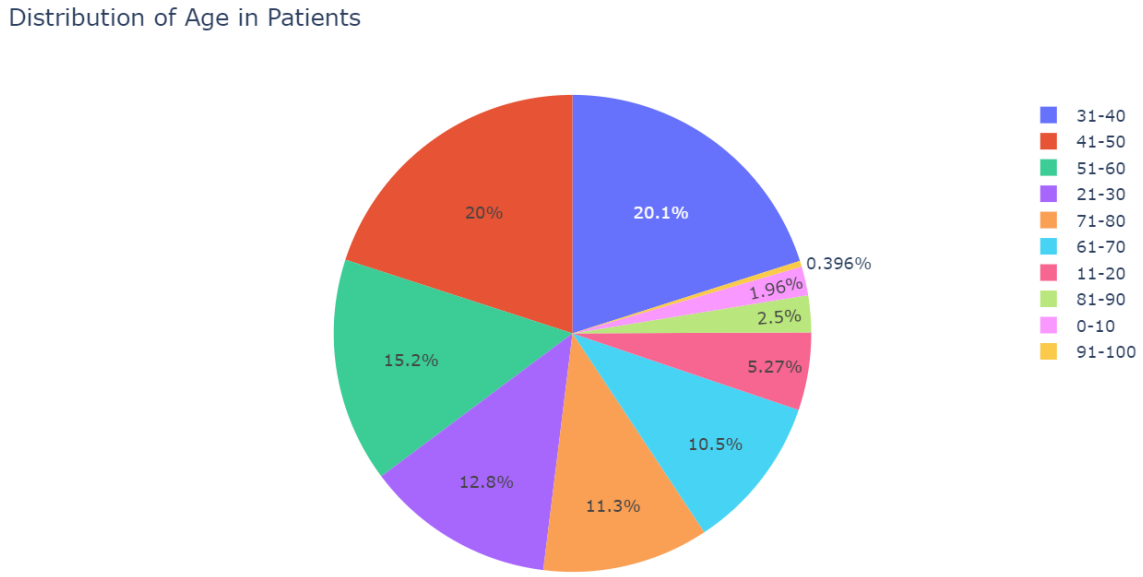
The above scatter plot shows the count and duration of patients belonging to different age groups. By looking at the size of different dots, we can say that patients with age group of 41-50 usually stays for about 21-30 days for their treatment.



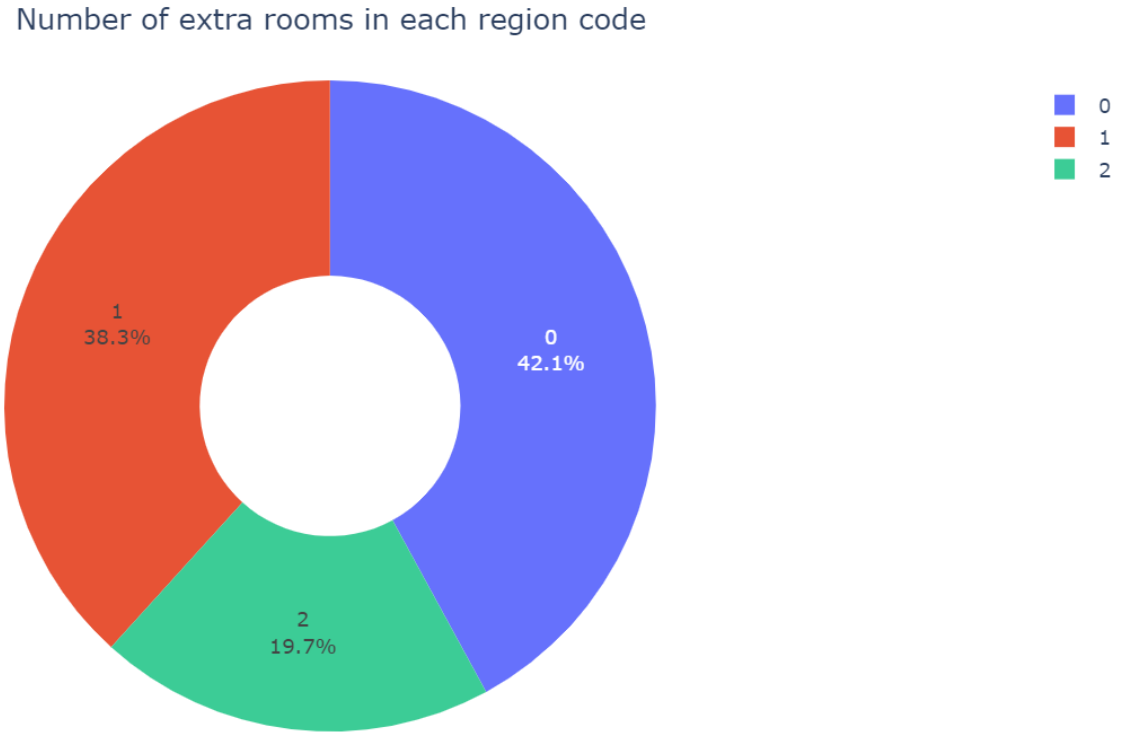
The above graph gives an idea about the deposit given by the patients during their admission and their usual length of stay. It can be inferred from the above visualization that the longer a patient plans to stay for his treatment the lesser he pays as admission deposit. This trend is not strongly correlated but we can draw some insights form this visualization.



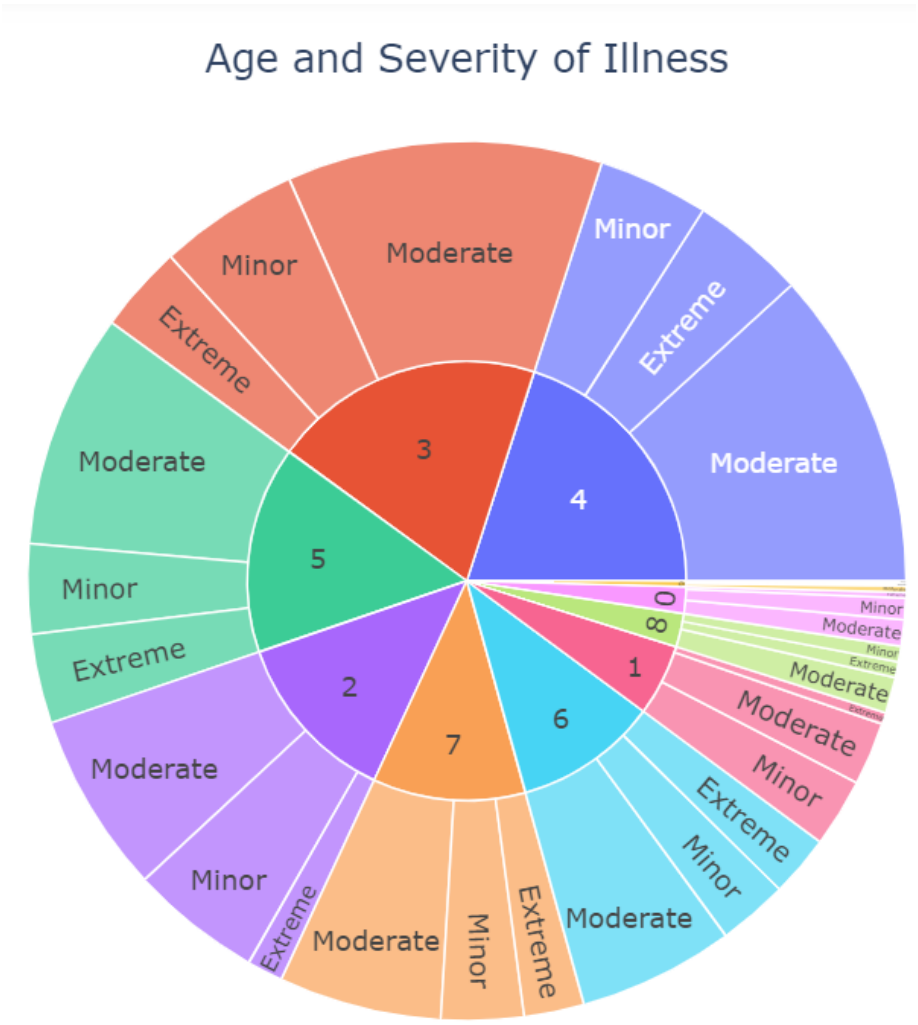
The above pie chart shows the distribution of patients among each department and clearly the Gynecology department has the highest number of patients with 80.1% of occupancy. This chart helps to show the distribution of patients among different departments.



The above figure shows the distribution of age among the all the patients records present in the data set. By looking at the above pie chart it is clear that the majority of the patients admitted are from the age group of 31-40 followed by 41-50.

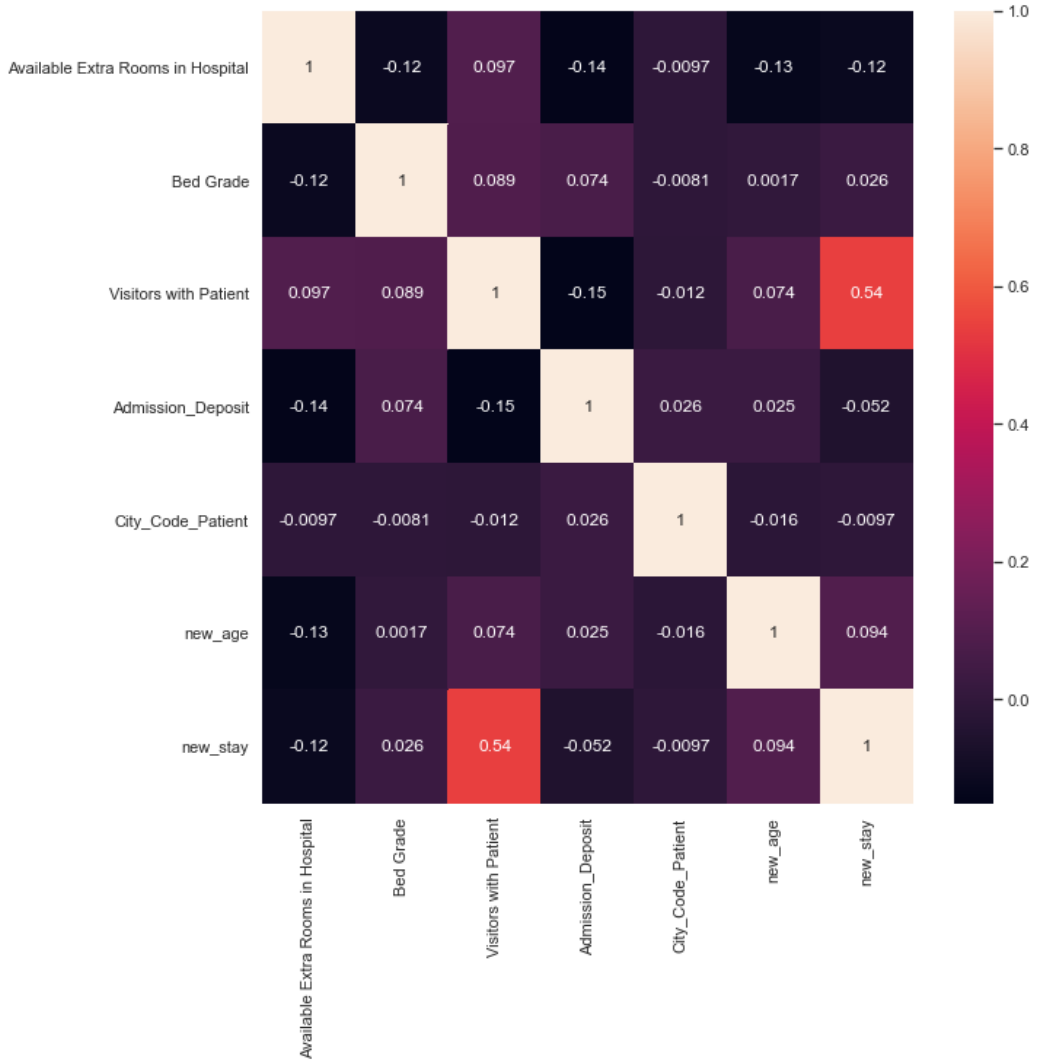


The above pie chart shows the distribution of extra rooms in each region. Since the data set has the information of three different regions, this chart helps to find the percentage of extra rooms available in each region and clearly the region a:0 has the highest number of extra rooms.



The above pie chart shows the information about different age groups with the corresponding severity of illness. The above visualization shows the percentage of people form each group and the corresponding level of severity. This is an important as the length of stay depends upon the severity of the patient.

CORRELATION:



The above heatmap helps to understand the correlation between different variables. By looking at this heat map, it is clear that the data is not closely correlated, and some variables are highly uncorrelated.

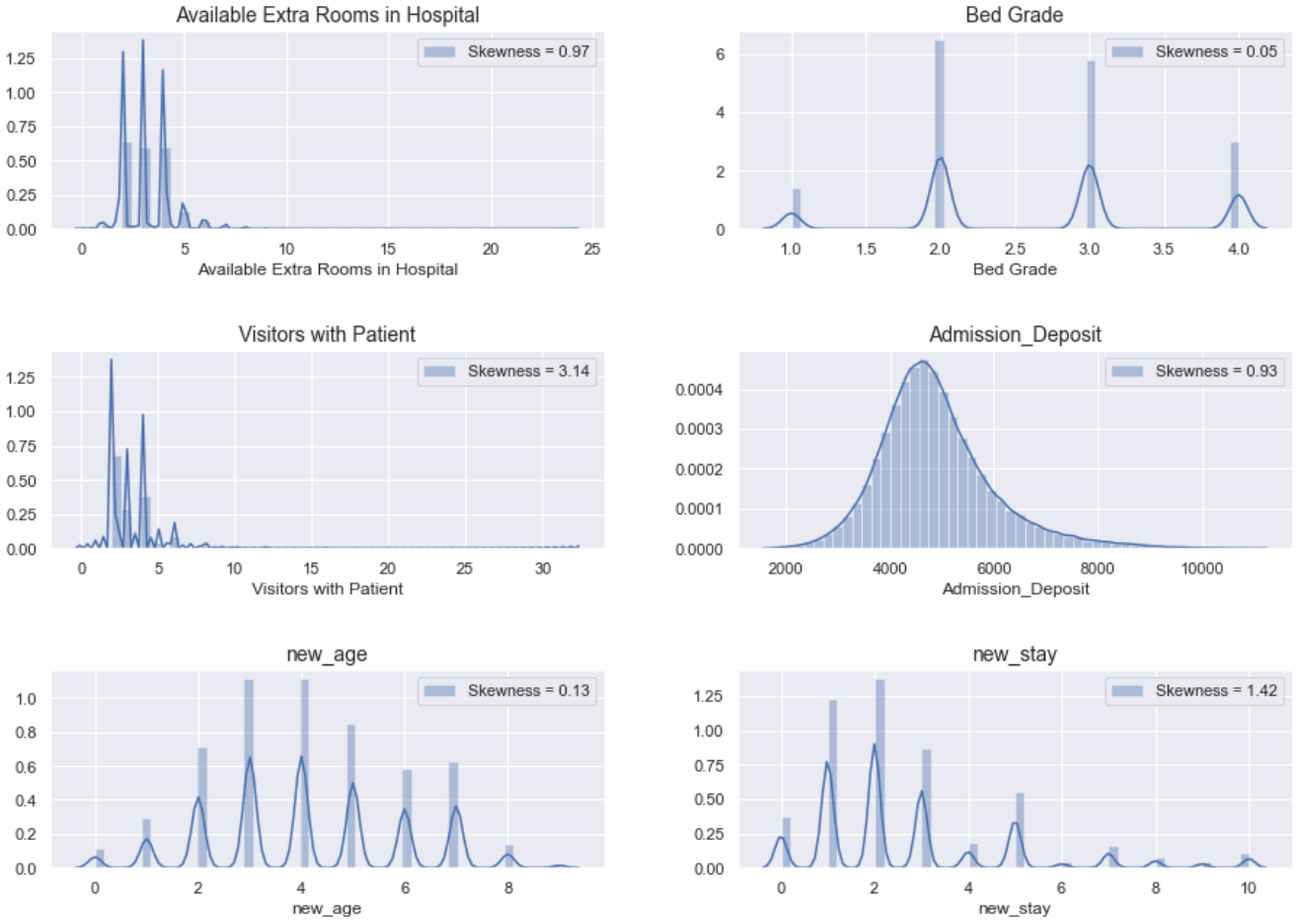
	Available Extra Rooms in Hospital	Bed Grade	Visitors with Patient	Admission_Deposit	City_Code_Patient	new_age	new_stay
Available Extra Rooms in Hospital	1.000000	-0.115868	0.096714	-0.143739	-0.009681	-0.133491	-0.121120
Bed Grade	-0.115868	1.000000	0.088945	0.073833	-0.008105	0.001732	0.025741
Visitors with Patient	0.096714	0.088945	1.000000	-0.150358	-0.012074	0.073795	0.537537
Admission_Deposit	-0.143739	0.073833	-0.150358	1.000000	0.025837	0.025182	-0.052077
City_Code_Patient	-0.009681	-0.008105	-0.012074	0.025837	1.000000	-0.016406	-0.009704
new_age	-0.133491	0.001732	0.073795	0.025182	-0.016406	1.000000	0.094163
new_stay	-0.121120	0.025741	0.537537	-0.052077	-0.009704	0.094163	1.000000

The above table shows the same correlation data with its related values.

CHECKING SKEWNESS AND PERFORMING NORMALIZATION:

By using the below code, I have checked the skewness of the important variables.

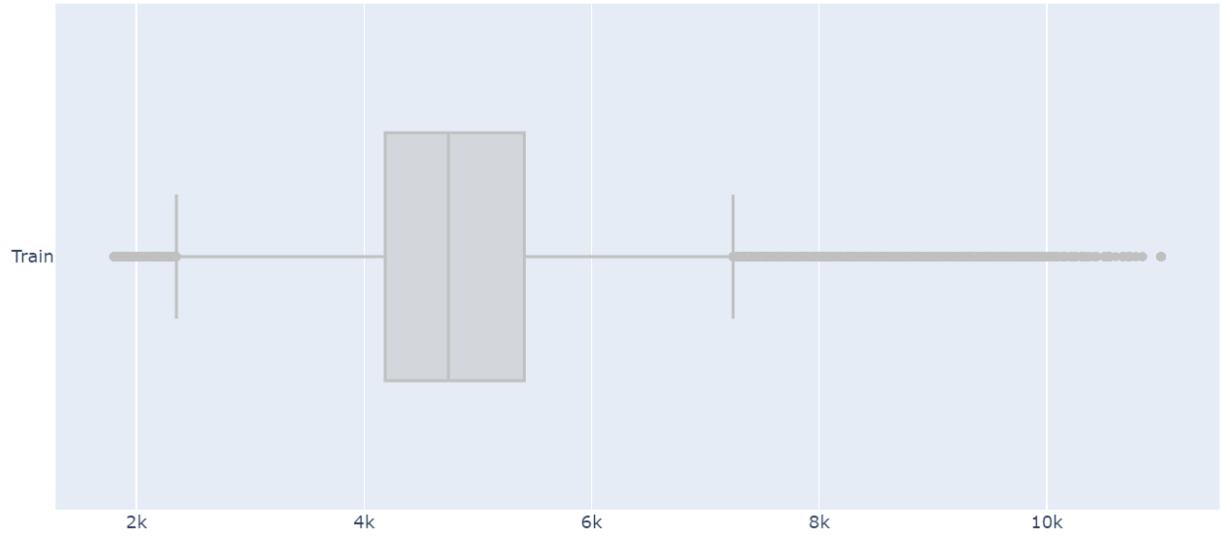
```
fig, new_plot =plt.subplots(3,2, figsize=(14,10))
fig.tight_layout(pad=5.0)

for new_plot, n in zip(new_plot.flatten(), numerical_data.columns.tolist()):
    sns.distplot(ax=new_plot, a=numerical_data[n], label="Skewness = %.2f"%(numerical_data[n].skew()))
    new_plot.set_title(n, fontsize = 14)
    new_plot.legend(loc = 'best')
```



If a value is higher than the 1.5*IQR above the upper quartile (Q3), the value will be considered as outlier. The higher outliers will cause the skewness and kurtosis of the distributions to become larger and more positive. The number of outliers will greatly affect the values. Outliers effect the skewness directly or indirectly, so I have plotted various boxplots to look for outliers.

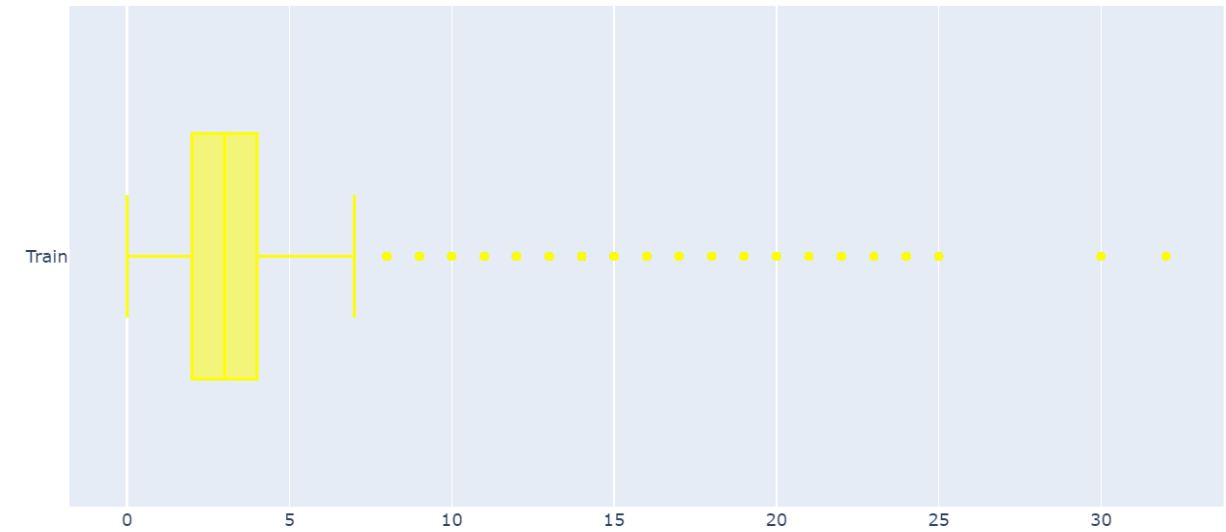
Looking for Outliers,

ADMISSION DEPOSIT:



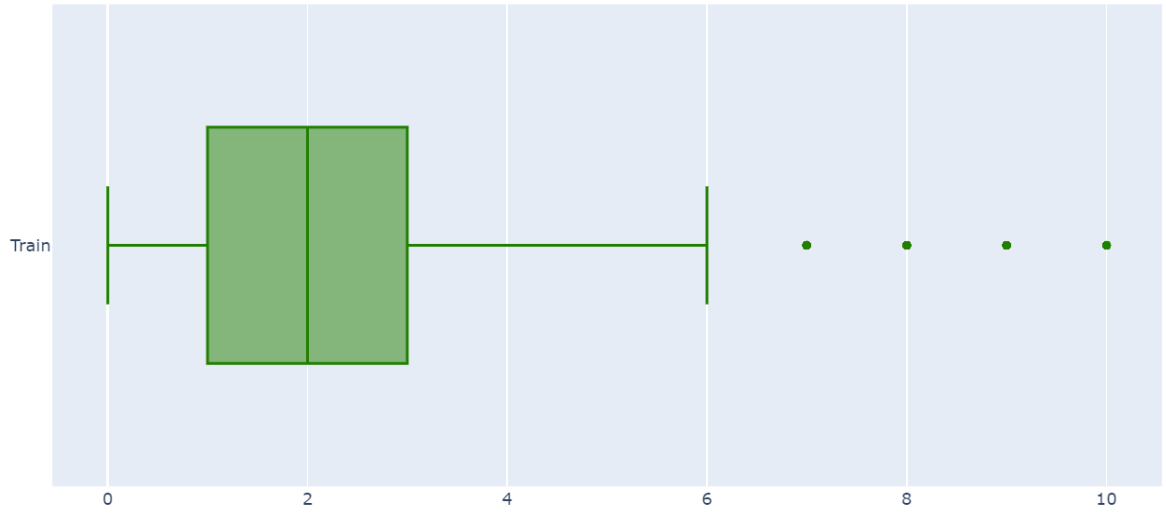
This the box plot of the variable “Admission Deposit”, as we can see from the above box plot there are many outliers present.

VISITORS WITH PATIENT:



This is the box plot of “Visitors with Patient”, it is clear from the above boxplot, the variable “visitors with Patient” has many outliers and we need to remove the outliers.

NEW_STAY:



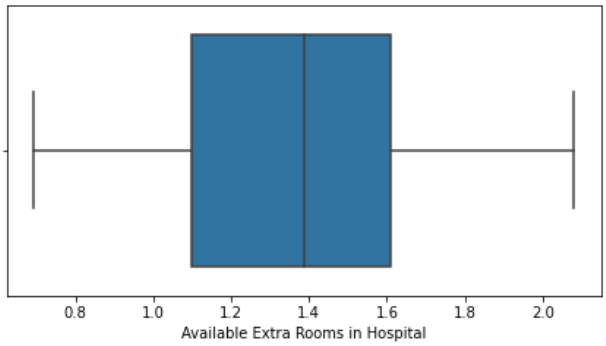
The variable “new_stay” also has many outliers and we can clearly see that by looking at the above boxplot.

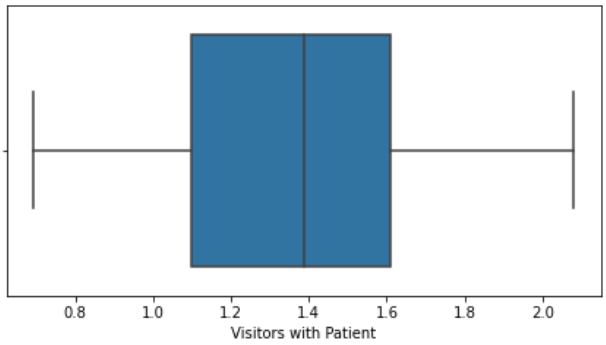
By looking at the above plots, the variables “visitors with patients” and “new_Stay” look highly skewed and I performed the following analysis, I calculated the Inter Quartile Range and if a data point is below Q1 – 1.5×IQR or above Q3 + 1.5×IQR, it is viewed as being too far from the central values to be reasonable. I have also compared it with the Z score and then I removed the values which are highly distorted.

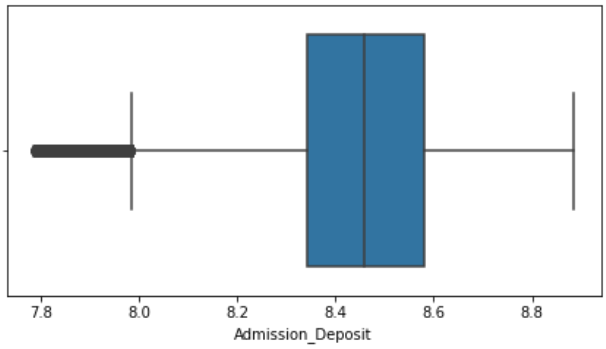
```
q1=data['Visitors with Patient'].quantile(0.25)
q3 = data['Visitors with Patient'].quantile(0.75)
iqr = q3-q1
data = data[~ ((data['Visitors with Patient'] < q1 - 1.5 * iqr) | (data['Visitors with Patient'] > (q3 + 1.5 * iqr)))]

q1=data['Admission_Deposit'].quantile(0.25)
q3 = data['Admission_Deposit'].quantile(0.75)
iqr = q3-q1
data = data[~ ((train['Admission_Deposit'] < q1 - 1.5 * iqr) | (data['Admission_Deposit'] > (q3 + 1.5 * iqr)))]
```

After performing the above calculation, I performed the logarithmic transformation, as log(0) is infinite , I have added plus one to the transformation so that the results are not distorted.

```
data['Available Extra Rooms in Hospital'] = np.log(train['Available Extra Rooms in Hospital'] + 1)
data['Visitors with Patient'] = np.log(train['Visitors with Patient'] + 1)
data['Admission_Deposit'] = np.log(train['Admission_Deposit'] + 1)


# Remove outliers after log transform on data train
data = data[data['Available Extra Rooms in Hospital'] > 0]
data = data[data['Visitors with Patient'] > 0]
data = data[data['Admission_Deposit'] > 0]
```







After performing the above analysis, the variables have normalized, and the outliers are removed. By looking at the above box plot picture it is clear. By using Inter Quartile Range and by comparing it with both the lower quartile value and upper quartile value and by performing logarithmic transformations, outliers are successfully removed and now the data is normalized.

CREATING MODELS:

RANDOM FOREST:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks 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/average prediction of the individual trees.

Random forest classifier is collection of decision trees from a randomly selected subsets of training sample. The algorithm will aggregate the votes from the decision trees and approves the final class object. Below is the implementation of Random Forest Classifier. I have imported the Random Forest classifier from the Sklearn package and implement the model.

Here I have created a model before removing the skewness and after removing the skewness of the data.

Random Forest model before removing the skewness:

```
clf_rf = RandomForestClassifier(n_estimators=1000, max_depth=15)

clf_rf.fit(X_train, Y_train)

Y_pred_rf = clf_rf.predict(X_val)
# get the accuracy score
acc_rf = accuracy_score(Y_pred_rf, Y_val)
print(acc_rf)
```

```
0.31042128603104213
```

The accuracy is very low here and it can be improved.

Random forest model after normalizing the data:

```
randfor = RandomForestClassifier(n_estimators=200, max_depth=15)

randfor.fit(X_train, Y_train)

pred_randfor = randfor.predict(X_val)
# get the accuracy score
accuracy = accuracy_score(Y_pred_rf, Y_val)
print(accuracy)
```

```
0.4003736967717623
```

As we can see that the accuracy has been improved by almost 10%.

Since the data is highly uncorrelated, the accuracy scores are very low.

The other models performed are

LINEAR REGRESSION:

linear regression is a linear approach to modelling the relationship between a scalar response and one or more explanatory variables. The case of one explanatory variable is called simple linear regression; for more than one, the process is called multiple linear regression

LINEAR REGRESSION

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
linear_reg = LinearRegression()

linear_reg.fit(X_train, Y_train)

pred_linear_reg = linear_reg.predict(X_val)

accuracy = linear_reg.score(X_val,Y_val)
print(accuracy)
```

```
0.361929490302932
```

The accuracy score of Linear regression is 36.1% and it is low when compared to random forest model.

KNN MODEL:

the k-nearest neighbors' algorithm is a non-parametric method proposed by Thomas Cover used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space.

The K Nearest Neighbors is a Supervised Machine Learning algorithm. This algorithm can be used for both Regression and Classification problems. The k indicates the number of nearest neighbors for the new data points. Choosing the k values has a drastic impact on the KNN results. I have imported KNeighborsClassifier from the Sklearn Library. The decision is dependent on the similarity of data points.

```
X_train, X_val, Y_train, Y_val = train_test_split(x, y, test_size = 0.2, random_state=0)
```

```
neighbors = KNeighborsClassifier(n_neighbors=11) # 11 different values of Stay
neighbors.fit(X_train, Y_train)
new_Y_pred= neighbors.predict(X_val)
# get the accuracy score
acc_neigh = accuracy_score(new_Y_pred, Y_val)
print(acc_neigh)
```

```
0.29609973621404345
```

The accuracy obtained by this model is much lower compared to other models, here we need to classify the number of neighbors we need. I have given 11 different values for length of stay, as the target variable is Length of stay, and we have 11 different lengths of stay. This model gave the lowest accuracy.

CATBOOST Model:

CatBoost is a depth-wise gradient boosting library developed by Yandex. It uses oblivious decision trees to grow a balanced tree. The same features are used to make left and right splits for each level of the tree. As compared to classic trees, the oblivious trees are more efficient to implement on CPU and are simple to fit.

```
X_train , X_test , Y_train , Y_test = train_test_split(X,Y,test_size = 0.2 , random_state = 0)
```

```
train_dataset = Pool(data=X_train, label=Y_train)

eval_dataset = Pool(data=X_test, label=Y_test)
```

```
model = CatBoostClassifier(iterations=750,
                           learning_rate=0.08,
                           depth=7,
                           loss_function='MultiClass',
                           eval_metric='Accuracy')
```

```
model.fit(train_dataset)

# validation

eval_pred = model.predict(eval_dataset)
```

```
0:      learn: 0.3810559        total: 734ms    remaining: 9m 9s
1:      learn: 0.3877331        total: 1.35s    remaining: 8m 27s
2:      learn: 0.3893111        total: 1.94s    remaining: 8m 4s
```

```
model.get_best_score()
```

```
: {'learn': {'Accuracy': 0.4592423945044161, 'MultiClass': 1.4052112296459598}}
```

This model has given the highest accuarcy of about 46%. Since the data is highly uncorrelated an accuracy of 46% is good fit for this model.

PREDICTION:

By using the tesing data, the values are predicted by using the following code,

```
new_dataset = Pool(test_X)

y_pred = model.predict(new_dataset)
```

```
output = pd.DataFrame(test_data['case_id'].values,columns=['case_id'])
output['Stay'] = y_pred
swap_dict_stay = dict([(value, key) for key, value in dept_Stay.items()])
output['Stay'].replace(swap_dict_stay, inplace=True)
```

```
output.head(5)
```

	case_id	Stay
0	318439	0-10
1	318440	51-60
2	318441	21-30
3	318442	21-30
4	318443	51-60

CONCLUSION:

Recently due to the Corona virus pandemic, many lives were lost and the major reason for that was lack of medical equipment like Oxygen Ventilators, surgical masks, PPE kits and many more. One more reason was the availability of beds was limited in regions like New York. Poor management of Hospital resources was the major reason for the failure of medical system during pandemics. This can be avoided by knowing exactly what a patient needs and by accurately predicting the length of stay of patient and the resources used by the patient, so that the hospital management can allocate the resources accordingly.

I have successfully created Machine Learning models, and I have successfully predicted the Length of Stay of patients. Since the data was highly skewed and highly uncorrelated the obtained accuracy was lower than ideal Machine Learning models.

RESEARCH WORK:

https://arxiv.org/ftp/arxiv/papers/1606/1606.01354.pdf

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6023432/

https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0217-0

https://healthitanalytics.com/features/what-is-deep-learning-and-how-will-it-change-healthcare

https://healthitanalytics.com/features/how-machine-learning-is-transforming-clinical-decision-support-tools

https://catboost.ai/

https://scikit-learn.org/stable/