Project :- Healthcare Analytics to Optimize Hospital Resource Management.

Final Project Report

Gavin Gunawardena, Divya Singh Sankhla

gavin.gunawardena@trojans.dsu.edu, divyasingh.sankhla@trojans.dsu.edu

1. Task

The task is to accurately predict the Length of Stay for each patient on a case-by-case basis so that the Hospitals can use this information for optimal resource allocation and better functionality. The length of stay is divided into 11 different classes ranging from 0-10 days to more than 100 days. The final goal of this project would be to help management to optimize staff allocation and functionality of hospitals by analyzing what patient details tend to minimize patient stay duration. This would in turn allow the hospital to efficiently accommodate the incoming patients, maximizing the benefit it provides to its community as well as the profits it generates to sustain itself.

2. Approach

For the project implementation we plan to fit a "Classification Model" to accurately predict the length of stay for patients at the hospital based on multiple independent parameters. Our approach towards model building proceeds with the following steps of implementation.

A. Data Formatting and Cleaning
    a. Identifying outliers and missing values.
    b. Removing outliers and imputing missing values
B. Libraries: - Numpy , Pandas, Seaborn, Sklearn , Matplotlib
C. Platforms Used :- Python , Tableau
D. Exploration and Data Analysis: - Explore data analysis and identify the correlation between variables and those variables impacting the Predictor significantly.
    a. Dummy encoding of the variables.
    b. Plotting visuals to interpret the relationship between the variables.
    c. Displaying meaningful insights based on the data.
E. Feature Engineering
    a. Optimizing the dataset for model buildings via actions such as:
        i. Dummy Coding ii. Data Scaling
F. Model Building
    a. Splitting dataset into test and train dataset: -
    b. Model Selection: -
        i. Comparing and Implementing different classification models. ii. Significance and description of models that we shall use
G. Defining Accuracy: - Metric to score the accuracy of the model
    a. 100*Accuracy Score.
H. Model Validation: - We will validate our data model by predicting the length of stay for the test dataset.
    a. Generate Confusion Matrix to validate the results of the model.

3. Dataset and Metric

For our project we are using the Healthcare Analytics Dataset retrieved from the Kaggle website.

Link to dataset:

- https://www.kaggle.com/nehaprabhavalkar/av-healthcare-analytics-ii?select=healthcare

This dataset includes around 137,000 test examples and around 318,000 training examples. Data preprocessing that we decided that this dataset needed includes: missing value imputation, dummy coding for categorical variables, and grouping/changing of bucket ranges of categorical variables (Age). Other than those that we have already addressed, we do not expect any issues with this dataset. Data Categorization of variables in the dataset:

Ordinal/Discrete	Nominal/discrete	Categorical
Case_id	Hospital_Code	Hospital_type_code
Available Extra Rooms in Hospital	City_Code_Hospital	Hospital_region_code
Bed Grade	Patientid	Department
	City_Code_Patient	Ward_Type
		Ward_Facility_Code
		Type of Admission
	Visitors with Patient	Severity of Illness
	Admission_Deposit	Age
		Stay

4.Results:

Results based on the preliminary work.

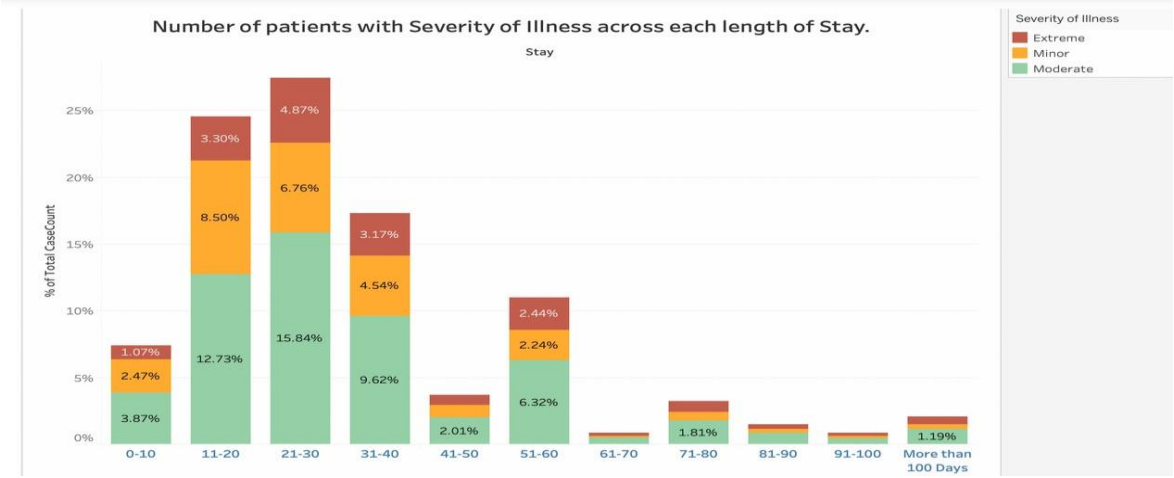
1. Missing Values identified in the dataset.

   a. For the attribute “Bed Grade” that represents the condition of the beds available there are 35 records missing in the test data and 113 records missing in the training data.
   b. There are a total of 2192 records missing in the “City_Code_Patient” field of the test data and 4,532 in the training data, representing the zip code for the patient's city.

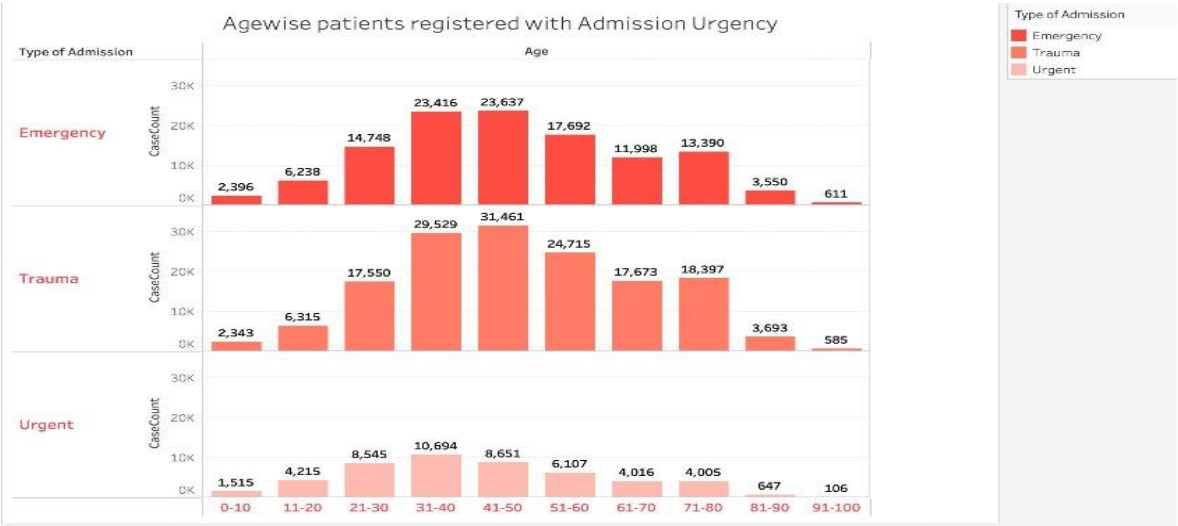
2. Dummy Encoding of the variables.

Categorical variables we lowered the amount of categories in include “Age” and “Stay”. The categorical variable we changed to a numeric was “Severity of Illness”. The variables we utilized dummy coding on were “Severity of Illness” and “Age Range”. All of this is subject to change as we continue the project.

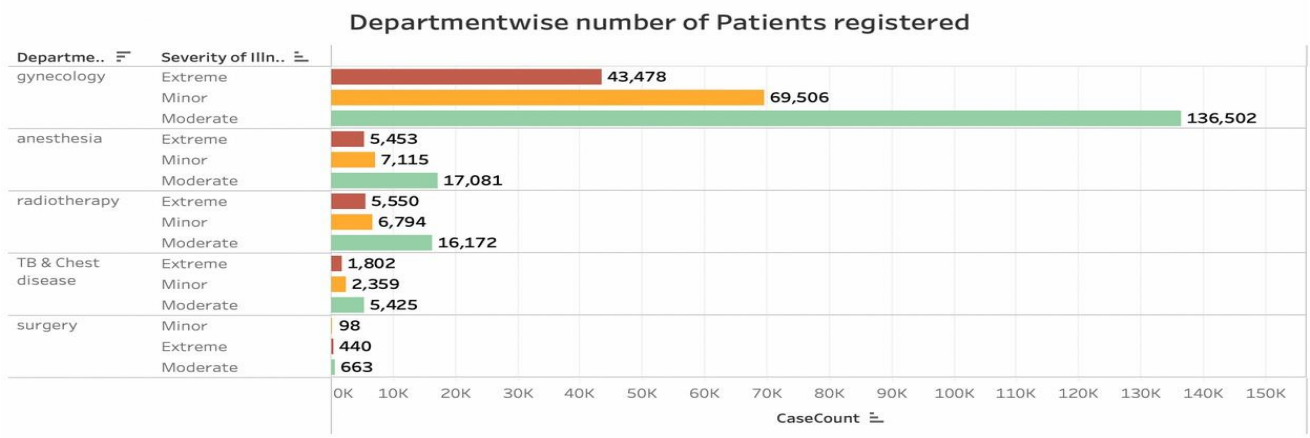
3. Results from Exploratory Data Analysis.



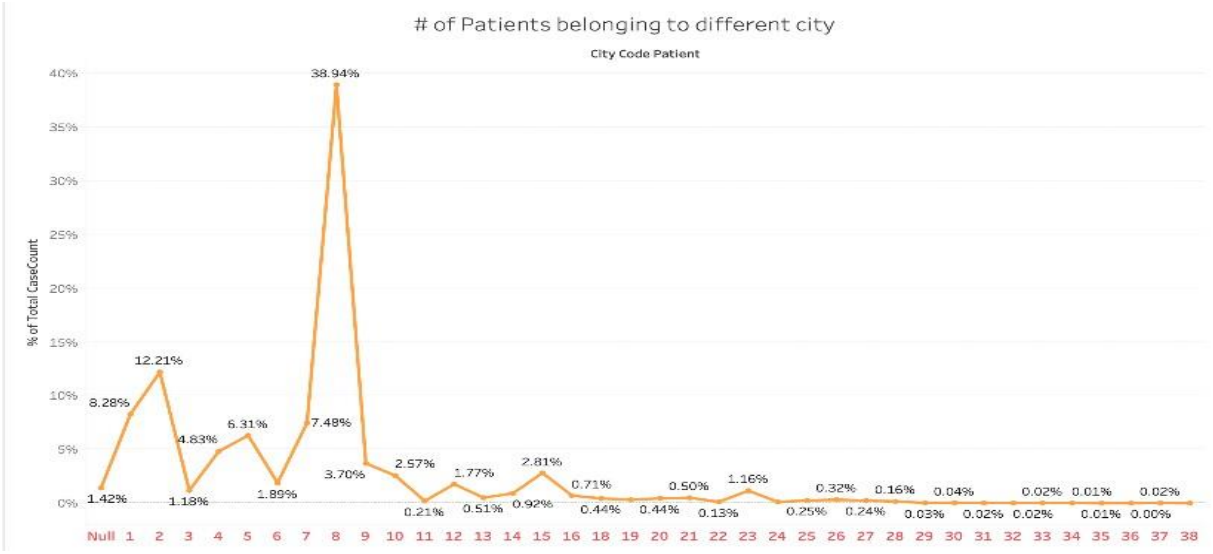
From the above chart, it appears that a large percentage of patients staying over a month are affected with Minor and Moderate illnesses. Based on this trend we suggest treating patients with minor illnesses early so that patients coming up with extreme illnesses can be allotted more resources.



The above chart shows how the age brackets "31-40", "41-50","51-60" have the highest density of patients corresponding to all of the types of admissions to the hospital.



A large fraction of Extreme cases for patients are registered in the Gynecology, Radiotherapy and Anesthesia departments. We may optimize the staff allocation by increasing the number of staff for these 3 departments, which may help the hospital to discharge its patients earlier.

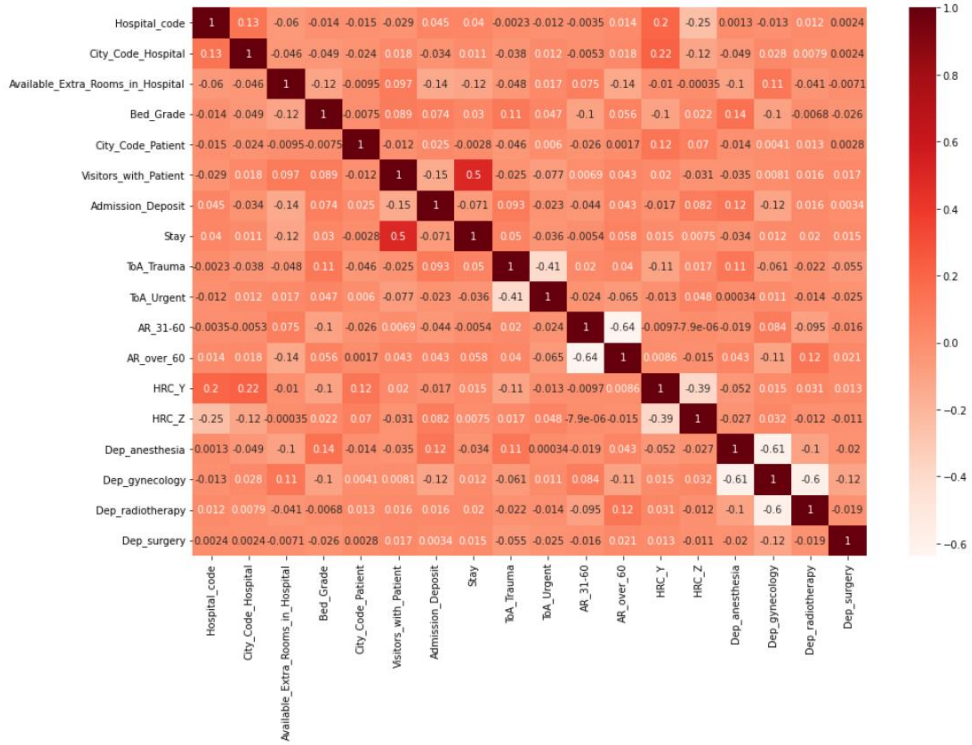


From the above charts it appears City Code 8 has the maximum density of patients coming into the Hospital. However, as per data, it appears there is no hospital established in city code 8 and we suggest building a hospital there to divert patient traffic from city codes 6 and 7, thereby helping patients receive healthcare.

Results based on the intermediary tasks.

1. Variable Selection

The criteria for selecting variables for the model input is based on the Pearson's coefficient which is demonstrated by the heatmap below.



For the selection, we used a threshold of > 0.02 of the Pearson r values for the variables to be considered as significant and valid for model input. The table below displays the selected variables and their r values.

Stay	1.000000
Visitors_with_Patient	0.497877
Available_Extra_Rooms_in_Hospital	0.117671
Admission_Deposit	0.071185
AR_over_60	0.058358
ToA_Trauma	0.049531
Hospital_code	0.039905
ToA_Urgent	0.036015
Dep_anesthesia	0.034424
Bed_Grade	0.030108
Dep_radiotherapy	0.020286

2. Feature Engineering:- We performed changes in the existing variables to tune our dataset and make it more explainable to the models. Categorical ordinal variables were converted to numerics based on their ordinality.

   - Severity of Illness Numeric:-(Key: 0=Minor, 1=Moderate, 2=Severe)
     Department:(Key: 0=TB & Chest disease, 1=anesthesia,
     2=radiotherapy, 3=gynecology)
   - Age:-(#Key: '0-30'=['0-10', '11-20', '21-30'] '31-60'=['31-40', '41-50', '51-60'], 'over 60'= rest of the data)
   - Stay:-(#Key: 1:['0-10','11-20,'21-30','31-40'], 2:['41-50','51-60','61-70'] 3:['71-80','81-90','91-100','More than 100 Days'])

3. Data Standardization

We performed data standardization to scale our data into the same range of values, since our dataset consisted of several different parameters. We utilized Sci-Kit Learn's Standard Scaler and fit it on the training dataset. Then we used the same object to transform Test and Validation datasets so that features of the dataset have the same weights in order to improve model accuracy and performance.

4. Data Split

This step consisted of a split of the training dataset into training and validation datasets. The split ratio we decided on was 70% for training and 30% for validation.

5. Model Testing

We decided to implement 3 models, Decision Tree, Random Forest, and Stochastic Gradient Descent. Models are selected based on their popularity and accuracy with classification problems, and also the ability of decision tree and random forest models to provide visuals of their decision process which can be used for convincing stakeholders and helping them understand the logic behind model decisions.

6. Model Tuning

Experiments we made to evaluate our approach were all based on trying to maximize the accuracy rates of our models. When first running our models, we were surprised to find very low accuracy rates on our test datasets. These accuracy rates were between 40 and 50 percent and thus we tried various methods to improve them.

At first we tried hyperparameter tuning, but via this, we were only able to improve accuracy rates by around 5% on average. For this project, we decided to, as mentioned earlier, utilize decision tree, random forest, and stochastic gradient descent. Due to the nature of these models, we had some flexibility in regard to hyperparameters to possibly solve the accuracy rate issues we were facing. With the decision tree, we were able to make a gain of about 9% in accuracy on the test dataset via lowering the max depth, increasing the minimum samples split, and adjusting the criterion of the splits from gini to entropy. With the random forest model, we were able to improve accuracy by about 5% via similar adjustments as the ones we made for the decision tree model. Finally for stochastic gradient descent, we were not able to make more than a decimal value impact on the accuracy rate by adjusting the hyperparameters and thus gave up after cycling through many of the different options it allowed. These adjustments included adjusting the loss function and penalty options. We did find that changing the loss function for SGD from hinge to log and penalty from L2 to L1 allowed us some accuracy gains, but as mentioned earlier, they were only improvements by about three tenths. This hyperparameter testing was automated via grid search in scikit learn.

Eventually we realized that we could make great gains on our accuracy rates by "moving the goalposts", or changing our end goal to something that was much easier to attain. Originally, our dependent variable had 11 different possible categories. Adjusting this to 3 increased the accuracy rates of our models by about 30%. We settled on this method and saw it as a success as it allowed us to obtain satisfactory accuracy rates (near 80%) for a machine learning model while not further complicating our processes.

Results Based on the Final Work.

1. Final Model Selected

We chose the Decision Tree as our final model as it gives us a very high accuracy rate on the test dataset compared to the other models while providing the fastest processing speed between the models.

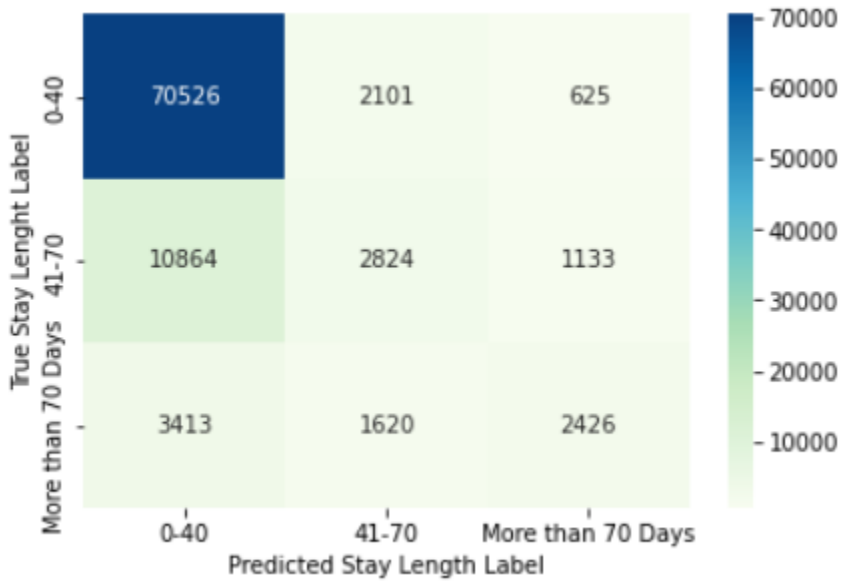
2. Displaying Model Accuracy Score Results

	Model	Prediction Accuracy - Training	Prediction Accuracy - Test
0	Decision Tree	79.64	79.32
1	Random Forest	85.61	79.30
2	Stochastic Gradient Descent	78.75	78.63

3. Displaying Confusion Matrix of our chosen model

A confusion matrix was made via the chosen model to compare the true values to the predicted values for the decision tree.

Decision Tree Confusion Matrix:



This confusion matrix shows how the decision tree was very accurate with the majority of the data, which fell into the 0-40 category, but still made many misclassifications, especially for the 41-70 category which it often incorrectly predicted to be part of the 0-40 category.

4. Displaying End Results on the Validation Dataset Predictions

tra_Rooms_in_Hospital	Admission_Deposit	AR_over_60	ToA_Trauma	Hospital_code	ToA_Urgent	Dep_anesthesia	Bed_Grade	Dep_radiotherapy	Stay
3	3095.0	1	0	21	0	0	2.0	0	0-40
2	4018.0	1	1	29	0	0	2.0	0	41-70
3	4492.0	1	0	26	0	0	4.0	0	0-40
3	4173.0	1	1	6	0	0	2.0	0	0-40
2	4161.0	1	1	28	0	0	2.0	0	41-70
3	4659.0	1	1	23	0	0	2.0	0	0-40
2	4167.0	1	1	26	0	0	2.0	0	0-40
4	4396.0	0	0	25	0	0	3.0	0	0-40
4	4088.0	0	1	23	0	0	3.0	0	0-40
3	3925.0	0	0	23	1	0	4.0	0	0-40

	Count
Stay	
0-40	121909
41-70	9264
More than 70 Days	5884

5. Detailed Timeline and Roles: -

Task	Deadline	Lead
Dummy Coding Variable Adjustments	11/14/21	Gavin Gunawardena
Missing Data Imputation Outlier Analysis	11/14/21	Divya Singh Sankhla
Data Visualization Analysis of Dataset	11/14/21	Divya Singh Sankhla
Variable Selection and Feature Engineering	11/28/21	Divya Singh Sankhla
Model Building	11/28/21	Gavin Gunawardena
Model Testing and Obtaining Accuracy	11/28/21	Gavin Gunawardena
Prepare report	12/12/21	all