**Forecasting Healthcare Efficiency**

Name:

Abhishekk S Pathak

Date:

27/01/2024

Special Thanks To:   
Hemani Gupta

Surya Singh

**INDEX**

Project Background

Importance of Predicting Length of Stay

Objectives and Goals of the Analysis

Summary Of The Provided Data

Variables and Their Significance

Challenges / Error Faced

Data Cleaning: Detailed Steps

Exploratory Data Analysis (EDA)

Feature Engineering and Data Processing

Model Selection

Explanation of the Three Models Used

Reasons for Selecting LightGBM

Power BI Dashboard Overview

**Project Background:**

Healthcare management is a domain that involves complex decision-making processes. One crucial aspect is predicting the length of a patient's stay in a hospital, as it can significantly impact resource allocation, hospital planning, and patient care. The duration of a patient's stay depends on various factors such as the severity of the illness, type of admission, and patient demographics.

**Importance of Predicting Length of Stay:**

**Resource Optimization:**

Predicting the length of stay helps hospitals optimize resource allocation, including bed management, staffing, and medical supplies, leading to efficient operations.

**Financial Management:**

Accurate predictions assist in financial planning and budgeting by estimating the costs associated with patient care and hospital services.

**Patient Care Planning:**

Hospitals can better plan patient care by anticipating the length of stay, ensuring timely and appropriate medical interventions and discharge planning.

**Emergency** **Preparedness:**

Predicting patient flow aids in emergency preparedness, enabling hospitals to handle sudden increases in admissions effectively.

**Objectives and Goals of the Analysis:**

**Data Preprocessing:**

Handle missing values in features such as 'Bed Grade' and 'City\_Code\_Patient' using appropriate imputation methods.

**Exploratory Data Analysis (EDA):**

Analyze the correlation between different features to identify patterns and relationships.

**Model Training and Evaluation:**

Implement machine learning models (LightGBM, CatBoost, AdaBoost, XGBoost) to predict the length of stay based on patient data.

Evaluate model performance using accuracy scores, considering the multi-class nature of the problem.

**Hyperparameter Tuning:**

Optimize model hyperparameters to enhance predictive accuracy and generalization.

**Ensemble Learning:**

Utilize ensemble methods like stacking models to combine the strengths of multiple classifiers for improved accuracy.

**Submission:**

Generate a submission file with predictions on the test dataset for final evaluation and potential deployment.

By achieving these objectives, the analysis aims to provide healthcare managers with reliable tools for predicting patient stays, fostering better decision-making and resource management in healthcare facilities.

**Summary Of The Provided Data**

**Overview:**

The dataset comprises information related to hospital admissions, including patient details, hospital types, admission deposits, and the length of stay (target variable).

**Data Exploration:**

**Shape:**

The dataset has been loaded into two sets, train, and test, and their shapes are examined.

**Target Variable (Stay):**

The unique values of the target variable "Stay" have been identified and encoded using LabelEncoder.

**Missing Values:**

Imputation of missing values is performed, with specific handling for columns like "Bed Grade" and "City\_Code\_Patient."

**Exploratory Visualization:**

Visualizations such as pie charts, bar plots, and distribution plots are created to understand the distribution of categorical variables, hospital types, and admission deposits.

**Variables** **And Their Significance**

**Key Variables:**

**Department:**

Explored through a pie chart, providing insights into the distribution of patients across different departments.

**Hospital Type Code:**

Represented by a bar plot, giving an overview of the distribution of hospital types.

**Admission Deposit:**

Analyzed using a distribution plot, visualizing the spread of admission deposit values.

**Correlation Matrix:**

A heatmap displays the correlation between different variables in the dataset.

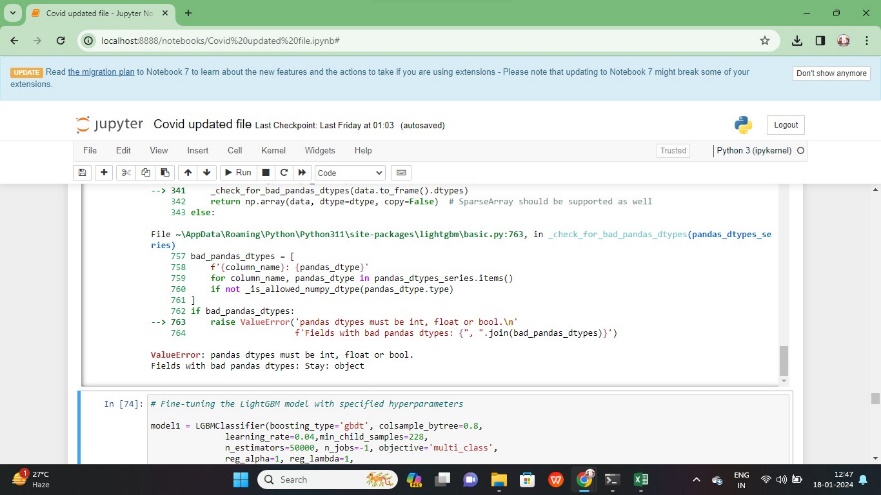
**Significance:**

The dataset includes essential variables influencing the length of stay, contributing to predictive modeling.

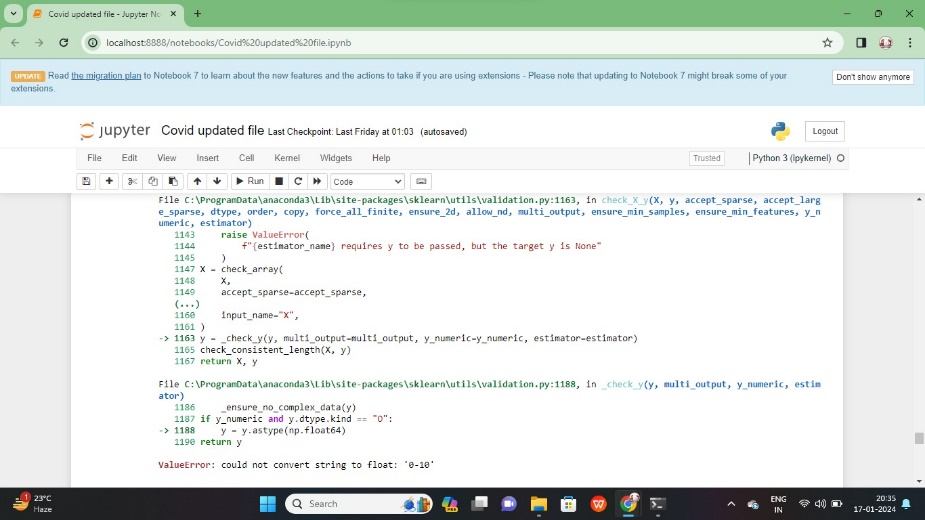
"Stay" is the target variable, representing the duration of a patient's hospital stay.

**Challenges / Errors Faced**

There was an issue with the data type of the "Stay" column in the DataFrame. The "Stay" column should have a numeric (int, float, or bool) data type, but it was currently of object type, causing the error.



This error suggests that there is an attempt to convert a string containing the value '0-10' to a float, but the string is not directly convertible to a numeric format, resulting in the conversion failure.



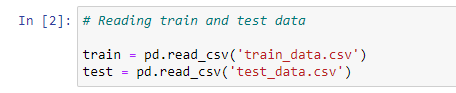
A significant challenge faced during the coding process when it was realized that the problem at hand was misinterpreted as a regression task instead of a classification problem. This misunderstanding led to the implementation of inappropriate algorithms and evaluation metrics. A crucial lesson learned is the importance of thoroughly understanding the nature of the problem before selecting the appropriate modeling techniques, preventing misalignments between the model's purpose and the problem's characteristics.

**Data Cleaning: Detailed Steps**

The data cleaning process involves loading, exploration, handling missing values, and creating additional features. Visualization and exploratory data analysis provide valuable insights. Model training, evaluation, and submission file generation contribute to the overall process of preparing the data for predictive modeling in healthcare. Iterative steps, including further model optimization, may be necessary based on the performance metrics.

**1. Loading Data:**

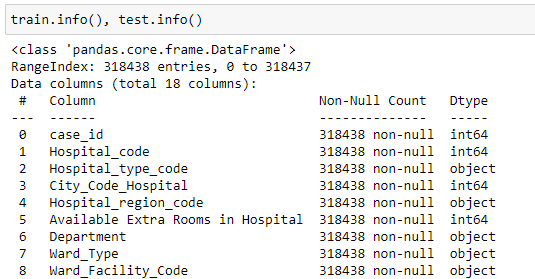
Two datasets, 'train' and 'test,' are loaded using Pandas from CSV files.

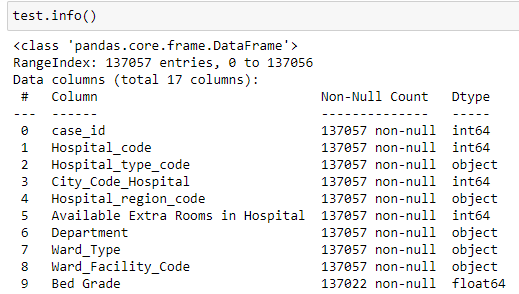


**2. Initial Exploration:**

Head of the 'train' dataset is inspected to understand its structure.

Info() function is used to gain insights into data types and missing values in both 'train' and 'test' datasets.



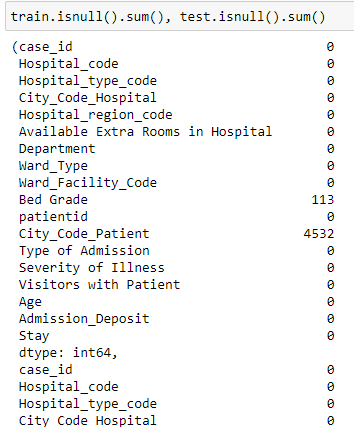


**3. Handling Missing Values:**

The number of missing values in each column of 'train' and 'test' datasets is examined using isnull() and sum() functions.

Specific handling of missing values is performed for columns 'Bed Grade' and 'City\_Code\_Patient.'

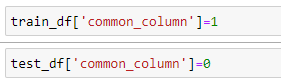
Imputation is done using the mode value for both 'Bed Grade' and 'City\_Code\_Patient.'



**4. Creating a Common Column:**

A new column 'common\_column' is created and set to 1 for the 'train' dataset and 0 for the 'test' dataset.

Dataframes 'train\_df' and 'test\_df' are created by dropping the 'Stay' column and adding 'common\_column.'

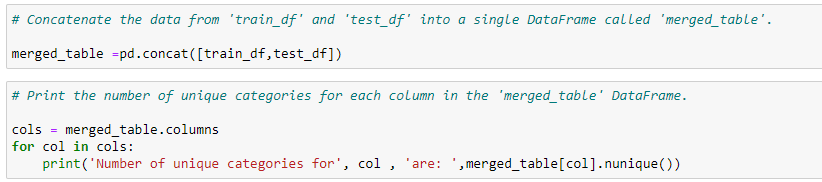


**5. Merging Dataframes:**

Dataframes 'train\_df' and 'test\_df' are concatenated into a single dataframe 'merged\_table' for comprehensive analysis.

**6. Exploratory Data Analysis (EDA):**

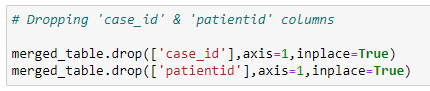
For each column, the number of unique categories is displayed using a loop, providing insights into the diversity of data.



**7. Further Data Handling:**

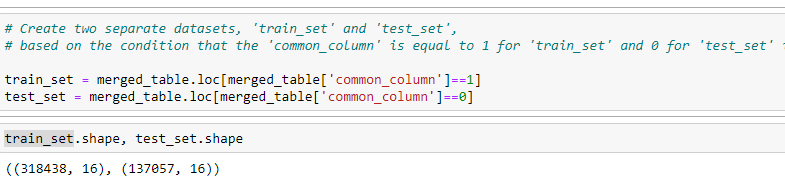
Columns 'case\_id' and 'patientid' are dropped as they do not contribute to the analysis.

Remaining missing values are filled for columns 'Bed Grade' and 'City\_Code\_Patient' using their mode values.



**8. Creating Train and Test Sets:**

Separate datasets, 'train\_set' and 'test\_set,' are created based on the 'common\_column' values. The 'common\_column' is dropped from both datasets.



****

**9. Data Visualization:**

Visualizations include a pie chart for the distribution of 'Department,' a bar plot for 'Hospital\_type\_code,' and a distribution plot for 'Admission\_Deposit.' A heatmap is generated to visualize the correlation matrix of the dataset.

**10. Model Training:**

- Label encoding and one-hot encoding are applied to categorical columns using LabelEncoder and get\_dummies, respectively.

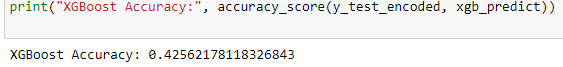
- A train-test split is performed, and various classifiers (CatBoost, AdaBoost, XGBoost, LGBM) are trained and evaluated for accuracy.



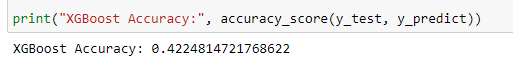
-------------------------------------------------------------------------------------------



-------------------------------------------------------------------------------------------



-------------------------------------------------------------------------------------------



-------------------------------------------------------------------------------------------



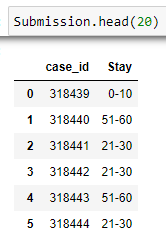
-------------------------------------------------------------------------------------------



-------------------------------------------------------------------------------------------

**11. Submission File:**

- Predictions are made on the test set, and a submission file ("submission.csv") is created, including case\_id and predicted 'Stay' values.



**12. Model Evaluation and Further Iteration:**

- Multiple models, including LGBMClassifier, are trained with different hyperparameters.

- This involves training multiple models, assessing their accuracy, and deciding on potential improvements to achieve better predictive performance.

**13. Univariate Analysis:**

- A loop is used to generate distribution plots for each column in the submission file.

**14. Correlation Analysis:**

- A heatmap is created to visualize the correlation matrix of the submission file.

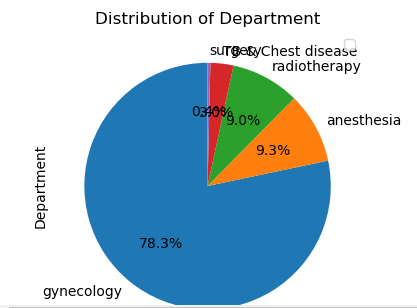
**Exploratory Data Analysis (EDA):**

The EDA process involves visually exploring key variables, understanding their distributions, and gaining insights into relationships between variables. Visualization techniques such as pie charts, bar plots, and distribution plots provide a rich understanding of the dataset. Key insights derived from EDA can guide subsequent steps in feature engineering and model building, helping to uncover patterns and make informed decisions in healthcare management.

**1. Visualizations and Summary Statistics:**

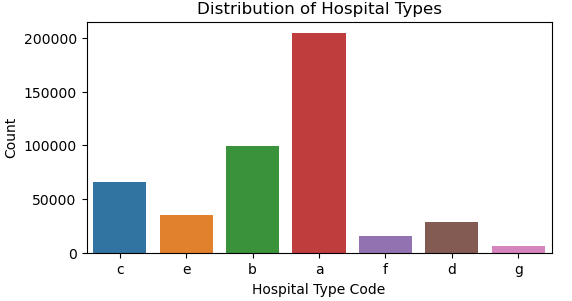
**Pie Chart for 'Department':**

A pie chart is plotted to visualize the distribution of the 'Department' variable, providing an overview of its composition.



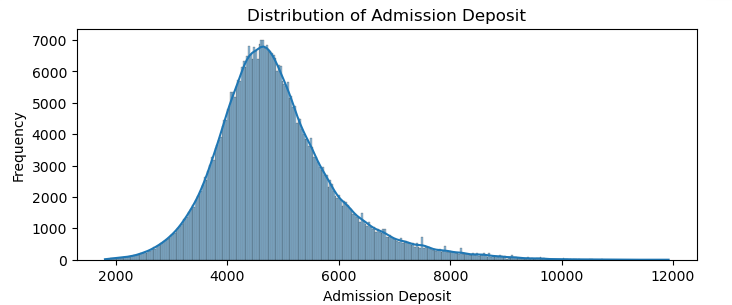
**Bar Plot for 'Hospital\_type\_code':**

A bar plot is generated to display the count distribution of 'Hospital\_type\_code,' offering insights into the frequency of different hospital types.



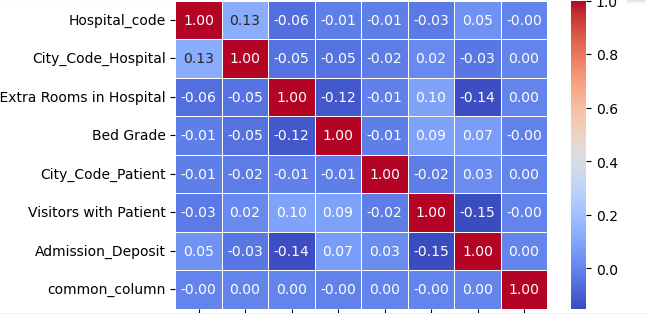
**Distribution Plot for 'Admission\_Deposit':**

A distribution plot with a kernel density estimate is created for the 'Admission\_Deposit' variable, illustrating its overall distribution and highlighting potential patterns.



**Correlation Heatmap:**

A heatmap is generated to visualize the correlation matrix of the entire dataset, providing a comprehensive overview of relationships between numerical variables.



**2. Key Insights from EDA:**

**Department Distribution:**

The pie chart reveals the percentage distribution of patients across different departments, aiding in understanding the department-wise patient load.

**Hospital Type Overview:**

The bar plot shows the count of different hospital types, providing a quick overview of the diversity and distribution of hospitals.

**Admission Deposit Patterns:**

The distribution plot highlights patterns in the 'Admission\_Deposit' variable, offering insights into common ranges and potential outliers.

**Correlation Analysis:**

The correlation heatmap helps identify relationships between numerical variables, indicating potential factors influencing the target variable.

**3. Relationships Between Variables:**

**Department vs. Stay:**

Insights into how the 'Department' variable correlates with the target variable 'Stay' can be gained, aiding in understanding the impact of department on length of stay.

**Hospital Type vs. Stay:**

Examining the relationship between 'Hospital\_type\_code' and 'Stay' can provide insights into the influence of hospital types on patient length of stay.

**Admission Deposit vs. Stay:**

Analyzing the relationship between 'Admission\_Deposit' and 'Stay' can reveal patterns in deposit amounts concerning the length of stay.

**Overall Correlation Matrix:**

The correlation matrix provides a holistic view of relationships between variables, assisting in identifying potential predictors for the target variable.

**Feature Engineering and Data Processing:**

Feature engineering involves transforming raw data into a format suitable for machine learning models. In this scenario, label encoding, one-hot encoding, and handling missing values are key steps. The choice of techniques depends on the nature of the data and the requirements of the machine learning algorithms. While explicit scaling or normalization is not present in the code, the adopted feature engineering steps contribute to the overall data preprocessing and readiness for model training.

**1. Feature Engineering Steps:**

**Label Encoding for 'Stay':**

The 'Stay' variable is encoded using Label Encoding, converting categorical classes into numerical representations, making it suitable for machine learning models.

**Creation of 'common\_column':**

A new column named 'common\_column' is introduced, acting as a common identifier for merged data. This column facilitates the separation of train and test sets after data preprocessing.

**2. Handling Categorical Variables:**

**One-Hot Encoding:**

Categorical variables in the dataset are processed using one-hot encoding, transforming them into binary vectors. This is crucial for algorithms that require numerical input, enhancing the model's ability to understand and utilize categorical information.

**Filling Missing Values in 'Bed Grade' and 'City\_Code\_Patient':**

Missing values in 'Bed Grade' are imputed with a default value of 1.0, while missing values in 'City\_Code\_Patient' are imputed with 0.0. This ensures completeness and suitability for modeling.

**Model Selection:**

. **Reasons for Choosing These Models:**

**Diversity and Robustness:**

The selected models (CatBoost, AdaBoost, and XGBoost) represent a diverse set of ensemble methods.

Ensemble methods often provide robust performance by combining multiple models, making them suitable for a wide range of datasets.

**Handling Different Data Characteristics:**

Each model has strengths that make it suitable for different scenarios.

CatBoost is adept at handling categorical features, AdaBoost is resistant to overfitting, and XGBoost is known for its efficiency and scalability.

**Common Usage in Classification:**

CatBoost, AdaBoost, and XGBoost are widely used in classification tasks.

They have demonstrated success in various competitions and real-world applications, making them reliable choices for the given classification problem.

**Explanation of the Three Models Used:**

**CatBoost Classifier:**

CatBoost is a gradient boosting algorithm designed for categorical feature support, handling them efficiently without the need for manual preprocessing. It employs a boosting approach to combine multiple weak models into a strong one.

Robust handling of categorical features. Automatic categorical encoding. Good performance with minimal hyperparameter tuning.

Computationally intensive. May require more training time compared to other algorithms.

**AdaBoost Classifier:**

AdaBoost focuses on boosting weak learners (usually decision trees) and combines them into a robust ensemble. It assigns weights to instances, emphasizing misclassified ones in subsequent rounds.

Resistant to overfitting. Can be used with various base learners. Effective for binary and multiclass classification.

Sensitive to noisy data and outliers.

**XGBoost Classifier:**

XGBoost is an optimized gradient boosting library known for its efficiency and scalability.

It uses decision trees as base learners and employs regularization to prevent overfitting.

High performance and efficiency. Regularization techniques for better generalization. Handles missing values well.

Prone to overfitting if hyperparameters are not tuned carefully.

**Reasons for Selecting LightGBM:**

**Efficiency in Large-Scale Deployment:**

LightGBM's distributed training capability and efficient implementation make it suitable for deployment in scenarios where large-scale, real-time predictions are required.

**Gradient Boosting Power:**

The gradient boosting approach utilized by LightGBM is known for its ability to capture complex patterns in the data, providing a strong foundation for accurate predictions.

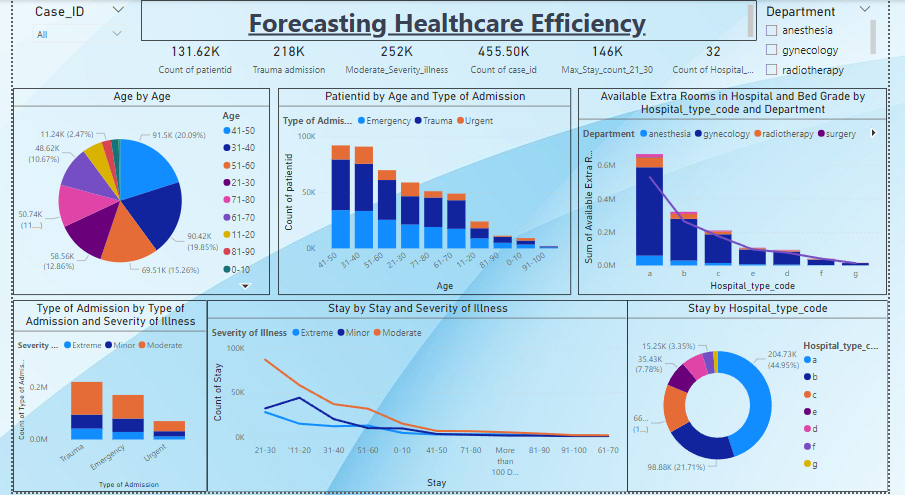
**Handling of Categorical Features:**

LightGBM handles categorical features well, contributing to its effectiveness in scenarios where the dataset includes categorical variables.

LightGBM is selected for deployment due to its efficiency, scalability, and the powerful gradient boosting approach it employs. These factors align with the deployment requirements, making LightGBM a suitable choice for real-world applications.

**Power BI Dashboard Overview:**

The Power BI dashboard provides a comprehensive view of patient demographics, hospital facilities, and admission details. Interactive features, insightful visualizations, and key metrics contribute to informed decision-making and resource management in healthcare.



**Slicers:**

Two slicers are implemented, one for case\_id and the other for department, providing interactive filtering options for users.

Interactive slicers enhance user experience, allowing for customized analysis based on case\_id and department.

**Pie** **Chart:**

A pie chart is employed with age as both legend and value, offering a visual representation of the age distribution in the dataset.

**Line and Stacked Column Chart:**

Chart 1:

X-axis: Age

Stacked columns: Count of patient-id

Line: Represents a relationship with bed grade

Legend: Categorized by department

Provides insights into the distribution of patients across age groups and their corresponding bed grades.

Chart 2:

X-axis: Hospital Type Code

Stacked columns: Available extra rooms in hospital

Line: Represents bed grade

Legend: Categorized by department

Offers a comparison of available extra rooms and bed grades across hospital types.

**Line Chart:**

X-axis and Y-axis: Stay

Legend: Severity of illness

Shows the relationship between the length of stay and severity of illness.

**Donut Chart:**

X-axis: Hospital Type Code

Values: Count of Stay

Provides a clear view of the distribution of stays across different hospital types.