Health Care Analytics

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Synopsis

1. **Title of Project:** Health Care Analytics

2. **Problems with the existing system:**
The pressure on healthcare institutions to enhance patient outcomes and provide quality treatment is growing. Even while this situation is difficult, it also gives enterprises a chance to significantly raise the standard of care by utilising additional information and insights from their data.

3. **Description of the proposed system:**
Health care analytics is the study of trends and patterns in acquired data using quantitative and qualitative techniques. While various performance metrics are used in healthcare management, a patient's length of stay is an important one.
Predicting the length of stay (LOS) enables hospitals to optimise their treatment plans in order to reduce LOS and infection rates among patients, staff, and visitors.

4. **Tools/Platforms:**

The Hardware requirement specifications:

Processor	11th Gen Intel(R) Core(TM) i5-1135G7
RAM	8.00 GB
Memory	512 GB
System type	64-bit operating system, x64-based processor

Hardware Requirements

The Software requirement specifications:

OS	Windows 11
Front End	Python, Jupiter notebook
Development Tool	IDLE

Software Requirements

5. **Methodology:**

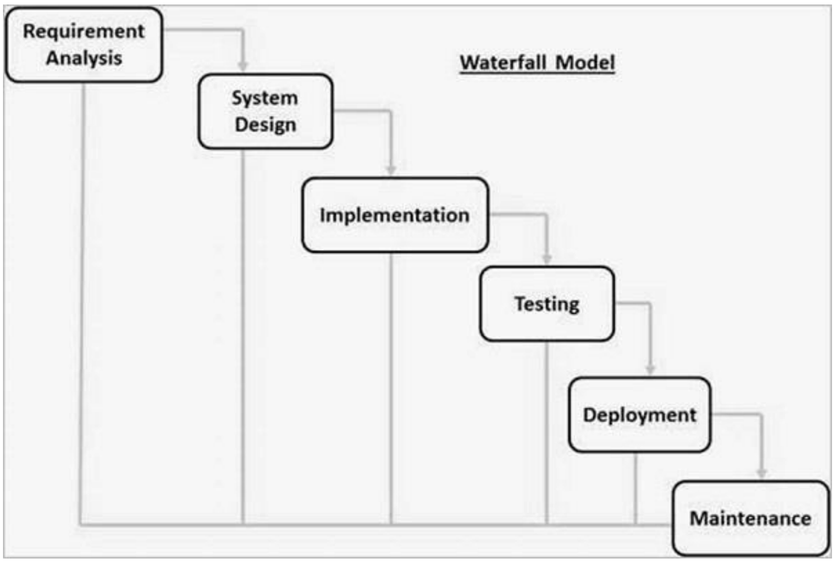
5.1._SDLC Model used

The SDLC model used in this project is WATERFALL MODEL.
A waterfall model is an example of a sequential model. The software development activity is divided into phases in this model, and each phase consists of a series of tasks with different objectives.

The SDLC processes were pioneered by the Waterfall model. In fact, it was the first widely used model in the software industry. It is divided into phases, with the output of one becoming the input of the next. A phase must be completed before proceeding to the next phase. In short, the Waterfall model has no overlap. The development of one phase in waterfall begins only after the previous phase is completed. As a result of this, each phase of the waterfall model is very precise and well-defined. The waterfall model is so named because the phases fall from a higher level to a lower level, much like a waterfall.

Phases of Waterfall Model:

a. **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
b. **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
c. **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
d. **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
e. **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
f. **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.



5.2. Justification for selection of model

a. This model is simple and easy to understand and use.

b. It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.

c. In this model phases are processed and completed one at a time. Phases do not overlap.

d. Waterfall model works well for smaller projects where requirements are clearly defined and very well understood.

6. Future Scope:

- Predict the Length of Stay of patients
- Help hospitals manage resources in development of new treatment plans
- Reducing the length of stay can reduce overall national medical expenses

Chapter-1

Software Project Planning

1.1. Description of the Software System under Study

Analyzing Hospital admission data to accurately predict the patient's Length of Stay at the time of admit so that the hospitals can optimize resources and function better. Understanding the problem in detail by Hypothesis generation: Assuming different factors that impact the outcomes of Length of Stay before any data exploration or analysis like dividing the variables into two levels, that is, Patient-Level and Hospital-Level.

1.2. Data Collection

The reference websites that are considered for creation of the project are:
GitHub – for dataset
Naïve Bayes for Machine Learning – From Zero to Hero (floydhub.com)
XGBoost Parameters — xgboost 2.0.0-dev documentation
21 Examples of Big Data In Healthcare With Powerful Analytics (datapine.com)

Modules selected for study:
Model 1 - Naïve Bayes
Model 2 – XGBoost
Model 3 – Neural Networks

1.3. Tools/Platforms

1.3.1. Hardware Specifications

Processor	11th Gen Intel(R) Core (TM) i5-1135G7
RAM	8.00 GB
Memory	512 GB
System type	64-bit operating system, x64-based processor

Table No 1.1: Hardware Specifications

1.3.2. Software Specifications

OS	Windows 11
Front End	Python, Jupiter notebook
Development Tool	IDLE

Table No 1.2: Software Specifications

1.4. Project Planning

Steps involved in project planning are:

a. **Requirement Gathering and analysis** – Gathering of dataset
b. **System Design** – Analyzing the variables and observations of dataset and hypothesis generation
c. **Implementation** – Cleaning and preparation of data
d. **Integration and Testing** – Feature Engineering
e. **Deployment of system** – Training of Model and checking the accuracy score and plotting the results in TensorBoard
f. **Maintenance** – Checking the code for any errors or fault in analysis

1.5. Methodology

1.5.1. SDLC Model to be used

The SDLC model used in this project is WATERFALL MODEL.

A waterfall model is an example of a sequential model. The software development activity is divided into phases in this model, and each phase consists of a series of tasks with different objectives.

The SDLC processes were pioneered by the Waterfall model. In fact, it was the first widely used model in the software industry. It is divided into phases, with the output of one becoming the input of the next. A phase must be completed before proceeding to the next phase. In short, the Waterfall model has no overlap. The development of one phase in waterfall begins only after the previous phase is completed. As a result of this, each phase of the waterfall model is very precise and well-defined. The waterfall model is so named because the phases fall from a higher level to a lower level, much like a waterfall.

1.5.2. Justification for the Selection of Model

- This model is simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model phases are processed and completed one at a time. Phases do not overlap.
- Waterfall model works well for smaller projects where requirements are clearly defined and very well understood.

Chapter-2:
Software Requirement Specification (SRS)

2.1. Description of Information System (Block Diagram):

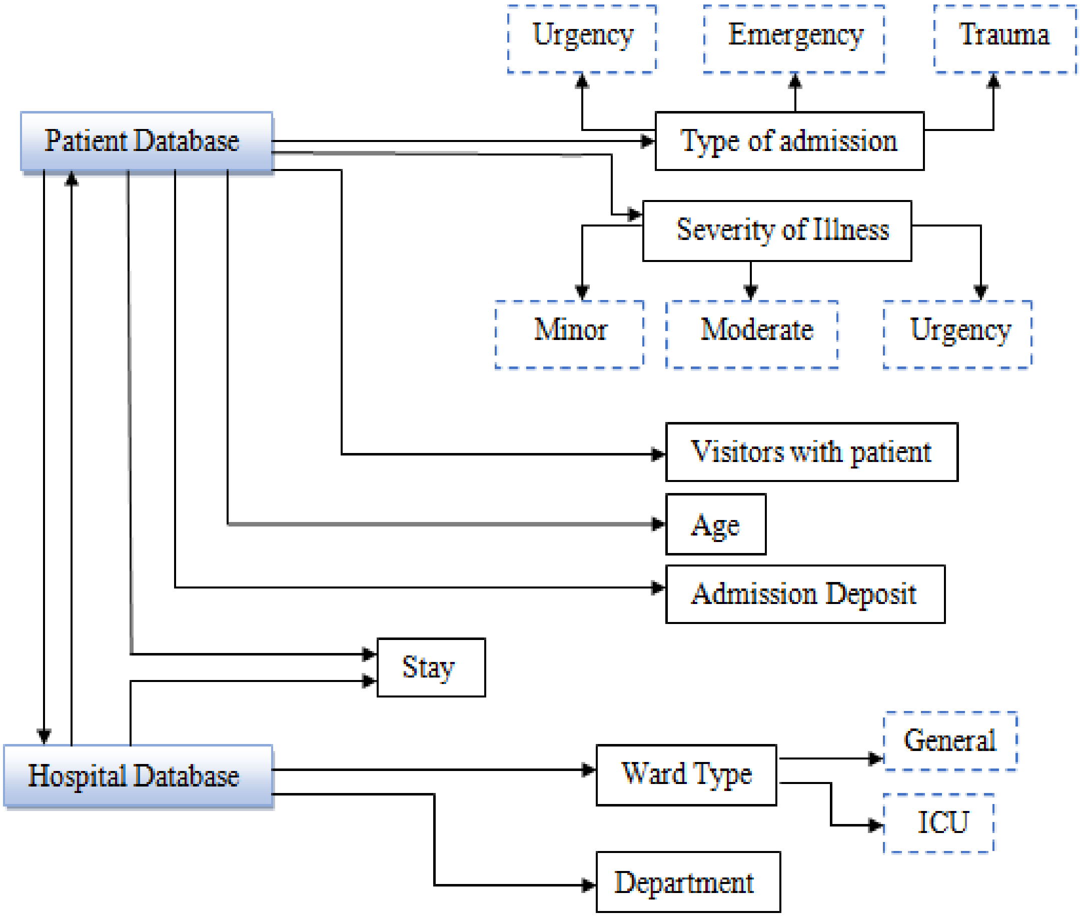


Fig 2.1: Block Diagram

2.1.1 Product Features:

This activity involves identifying process such as finding out that how long a patient stays in the healthcare. The process is called as **"Length of stay (LOS)".**

2.1.2 Input/ Output Data elements:

Input/Output name	Data Elements
Stay	Type of admission, Severity of illness, Visitors with Patient, Age, Admission_Deposit, Ward_Type, Department

Table 2.1: Input/Output Elements

2.1.3 Procedures/ rules/ mathematical relationships:

1. **Naïve Bayes theorem**

- P€ is the prior probability of a patient’s length of stay (LOS).
- P€ is the probability of a feature variable.
- P(E|H) is the probability of patient’s LOS given that the features are true.
- P(H|E) is the probability of the features given that patient’s LOS is true.

$$\mathbf{P\ (H\mid E) = [\ \frac{\mathit{P\ (E\mid H)}}{\mathit{P(E)}}]\ P(H)}$$

2. **Boosting –** is a sequence technique that works on the principle of an ensemble.
3. **Neural network-**Neural Networks are built of simple elements called neurons, which take in a real value. Multiply it by weight, and fan it through a non-linear activation function. The process records one at a time and learns by comparing their classification of the record with the known actual classification of the record. The errors from the initial classification of the first record are fed back into the network and used to modify the network’s algorithm for further iterations
4. **Softmax –** Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector

2.2. Use case diagram:

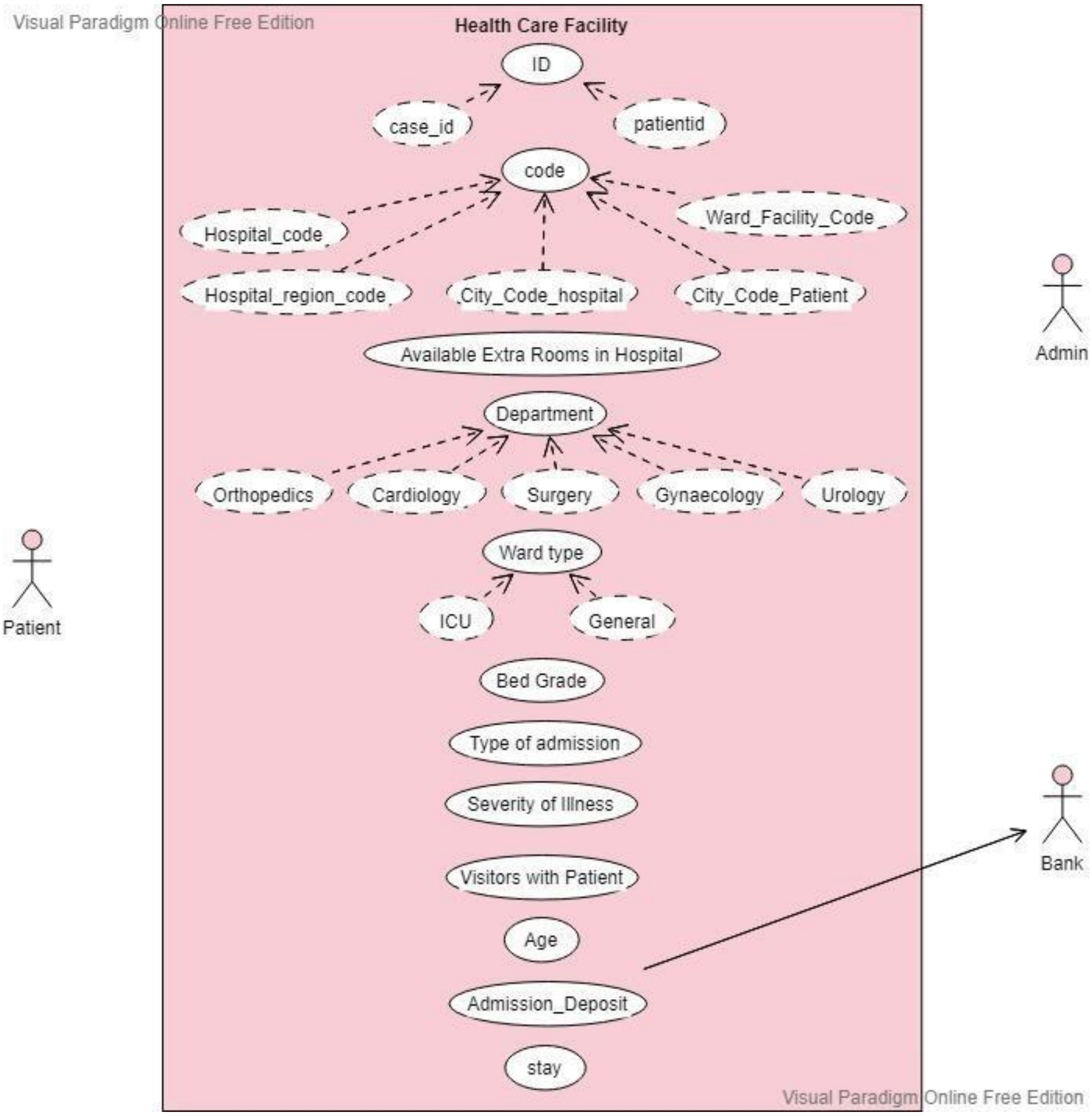


Fig 2.2: Use-Case Diagram

2.3. Software Product constraints:

In this project “City_code_patient” and “Bed Grade” are the constraints as they have missing values. These missing values must be treated before feeding to the algorithm as they distort the model performance.

“Mode” imputation technique can be used to replace these missing values.

Chapter-3

Software Project Analysis

3.1. Data Flow Diagram

Data-flow diagram is a way of representing a flow of data through a process or a system.

The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops.

DFD Level 1:

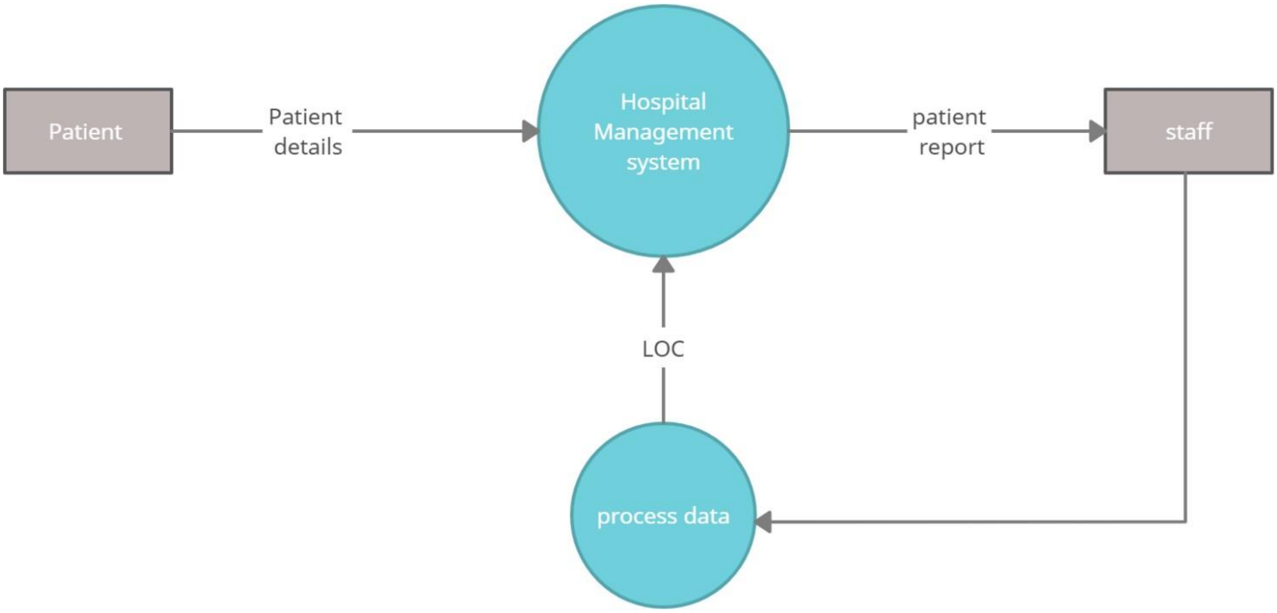


Fig 3.1a: DFD Level 1

The above data flow diagram shows the process of Hospital Management System where the patient details are taken from the patient and then the patient report is forwarded to staff and the staff hence process the data, calculate the LOS, and give it back to the Hospital Management.

DFD Level 2:

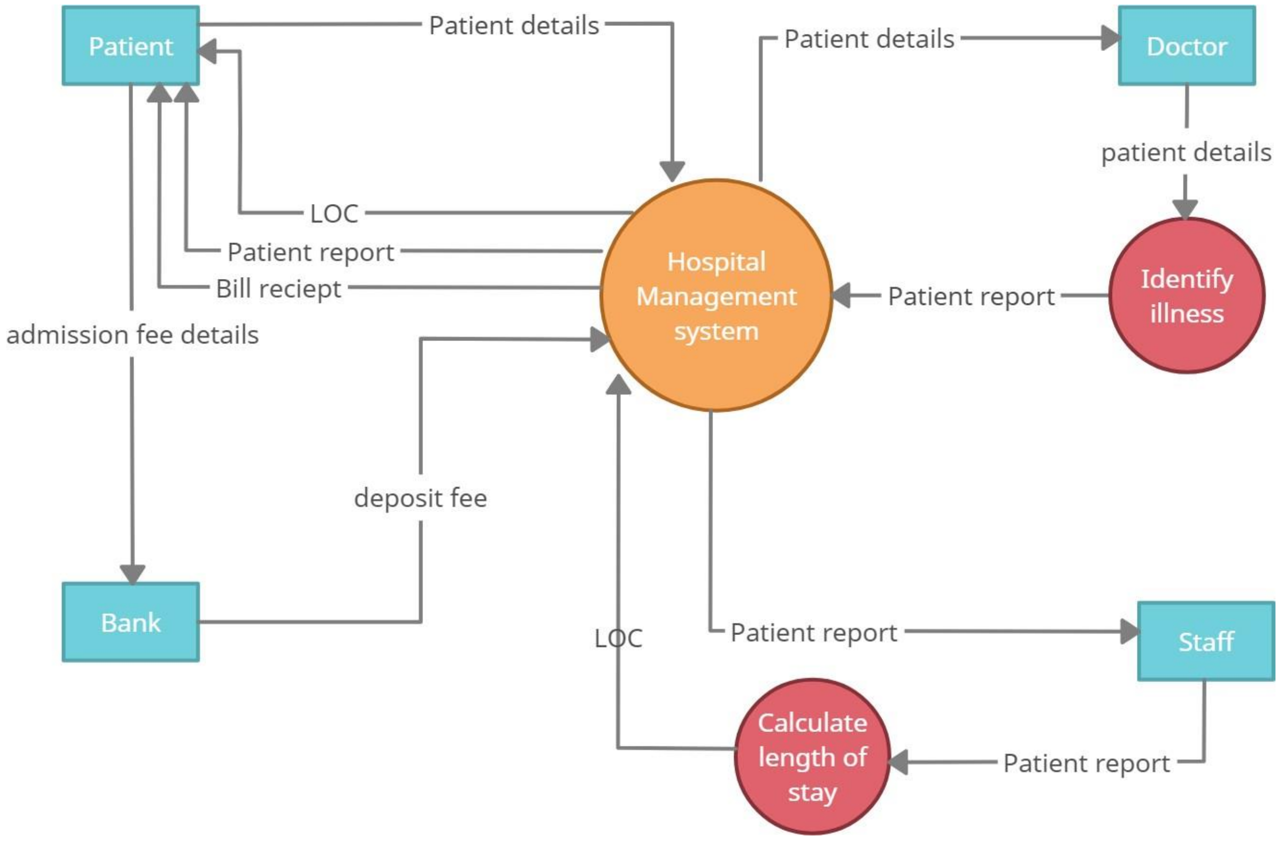


Fig 3.1b: DFD Level 2

The above diagram shows the Level 2 of the data flow diagram for the Hospital Management System. The Patient, Doctors, Staff and Bank are directly related to the system. The Doctors identify illness and give a patient report to the Hospital, and it is further sent to the staff who calculates the LOS and send it the Hospital Management System which can henceforth help the Management to optimise their care and resources better.

3.2. Entity – Relationship Diagram

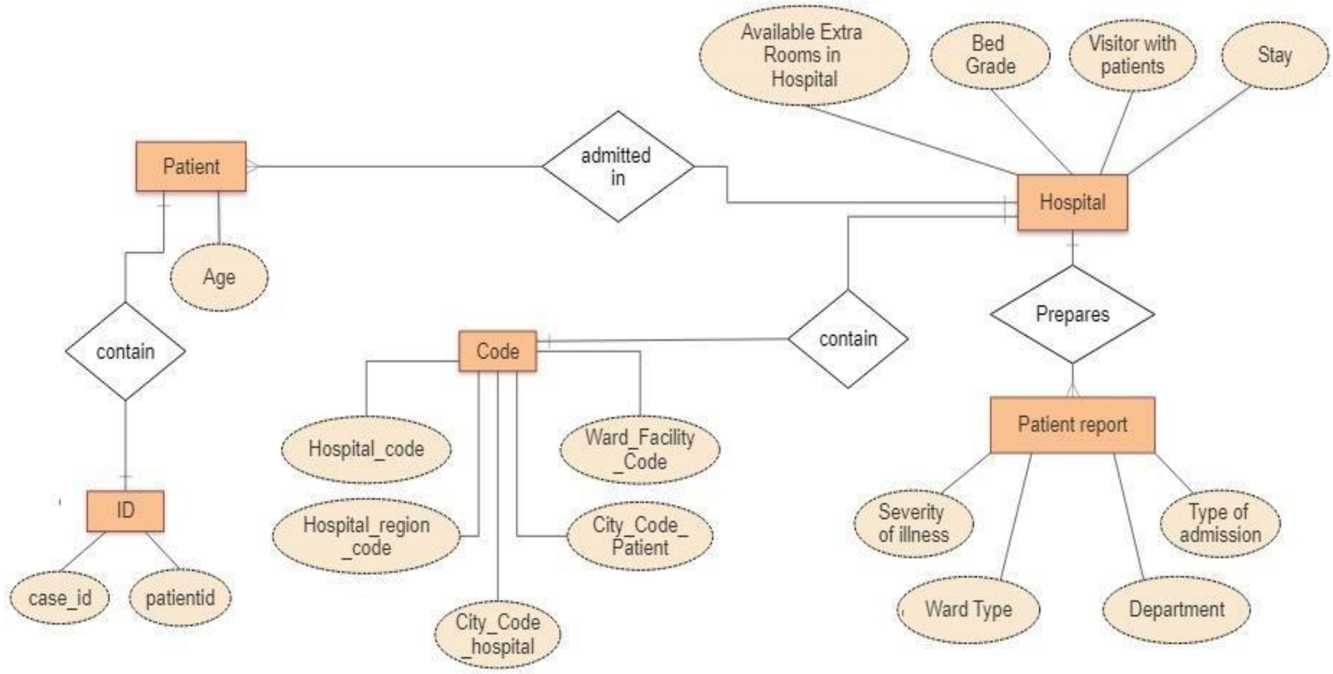


Fig 3.2: ER Diagram

An entity-relationship diagram model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities. So here, Patient, Hospital, ID, Code and Patient Report are all entities which have their different attributes. Hence, they both, the entities, and their attributes with the dataset, are related to each other and form an association which bolsters the whole project and its operations. In a way through this model, it clearly visualizes the whole project functioning and modelling.

3.3. Database Specifications

FILE NAME: HEALTHCARE ANALYTICS			
Field Name	**Field Type**	**Size**	**Description**
case_id	Numeric	6	Case_ID registered in Hospital
Hospital_code	Numeric	2	Unique code for the Hospital
Hospital_type_code	Character	1	Unique code for the type of Hospital
City_Code_Hospital	Numeric	2	City Code of the Hospital
Hospital_region_code	Character	1	Region Code of the Hospital
Available Extra Rooms in Hospital	Numeric	2	Number of Extra rooms available in the Hospital
Department	Character	18	Department overlooking the case
Ward_Type	Character	1	Code for the Ward type
Ward_Facility_Code	Character	1	Code for the Ward Facility
Bed Grade	Numeric	1	Condition of Bed in the Ward
patientid	Numeric	6	Unique Patient Id
City_Code_Patient	Numeric	2	City Code for the patient
Type of Admission	Character	9	Admission Type registered by the Hospital
Severity of Illness	Character	8	Severity of the illness recorded at the time of admission
Visitors with Patient	Numeric	2	Number of Visitors with the patient
Age	Character	6	Age of the patient
Admission_Deposit	Numeric	5	Deposit at the time of Admission
Stay	Character	18	Patient Length of Stay

Table 3.1: Database Specification

3.4. Validation Specifications

Validation rules include:

- Cleaning the dataset and working with NA or missing values.
- Combining columns for analysis
- Dividing the train and test dataset
- Ensuring the working of models

Hypothesis Generation:

Understanding the problem in detail by assuming different factors that impact the outcomes of Length of Stay before any data exploration or analysis. Here the variables can be divided into two levels: Patient-Level and Hospital-Level.

Patient-Level:

- Type of Admission – Patients can be admitted in three levels Urgent, Emergency, and Trauma.

Patients admitted to urgent care are likely to stay fewer days. Whereas Trauma patients usually stay longer because they must be monitored until they are qualified to be discharged.

- Severity of Illness – Severity can be classified as Minor, Moderate, and Extreme.

A patient recorded as minor will stay fewer days than a patient recorded as extreme.

- Visitors with Patient – Patients with more visitors are like to stay longer in the hospital.
- Age – Infants and older Patients usually take a longer time to recover so they stay longer than younger Patients.
- Admission Deposit – Patients who are likely to deposit a high amount of money at the time of admission might have severe conditions and stay longer.

Hospital-Level:

- Ward Type – Patients allocated in ICU might stay longer than the general ward as their condition is more severe.
- Department – Patients under surgery are likely to stay longer than gynaecology as their recovery time is longer.

Data Exploration

Overview of Data: The train data consist of 318438 observations for which patient length of stay can be predicted from 17 variables.

In the data, the target variable “stay” is divided into 11 different classes ranging from 0 days to more than 100 days.

```
array (['0-10', '41-50', '31-40', 'Nov-20', '51-60', '21-30', '71-80',
       'More than 100 Days', '81-90', '61-70', '91-100'], dtype=object)
```

Data Cleaning and Preparation:

In the data set, variables “City_code_patient” and “Bed Grade” have missing values. These missing values must be treated before feeding to the algorithm as they distort the model performance. So, the missing values are replaced using the “mode” imputation technique. Since most of the variables in the dataset have ordinal data, we transformed them into levels by using a label encoder to perform further analysis on the data.

Variables	Number of Distinct Observations
Hospital_type_code	7
Hospital_region_code	3
Department	5
Ward_Type	6
Ward_Facility_Code	6
Type of Admission	3
Severity of Illness	3
Age	10
Stay	11

Table 3.2: Distinct Observations of Ordinal Data

Feature Engineering

Once the data is cleaned and prepared, we grouped patientid and case_id to extract the new column "count_id_patient". This variable contains the count of multiple admits of a patient under different case_id.

Further two more columns "Hospital_region_code" and "ward_facility_code" were grouped to patientid and case_id.

These two new variables "count_id_patient_hospitalCode" and "count_id_patient_wardfacilityCode" contain the count of multiple admissions in a hospital region and the count of multiple wards allocated to a patient.

Before getting into analysis, the train data must be split into two parts, the first part with all the feature variables and the second part with a target variable ("Stay"). Then pre-processed into train and validation sets. So, here we portion the train set with 80% and validation set with 20% of the data for Naïve Bayes and XGBoost models.

Models Used:

Model 1 - Naïve Bayes

Naïve Bayes is a classification technique that works on the principle of Bayes theorem with an assumption of independence among the variables. Here the goal is to predict Length of Stay i.e., "Stay" column (Target Variable) and it is classified into 11 levels. We must find the probability of each patient's length of stay using feature variables, which contain the patient's condition and hospital-level information. These feature variables are ordinal and naïve Bayes is a perfect multilevel classifier.

In Bayes theorem, given a Hypothesis H and Evidence E, it states that the relation between the probability of Hypothesis P(H) before getting Evidence and probability of hypothesis after getting Evidence P(H|E)

$$P\,(H \mid E) = [\, \boldsymbol{P}\,(\boldsymbol{E} \mid \boldsymbol{H}) \,/\, \boldsymbol{P}(\boldsymbol{E})]\, P(H)$$

When we apply Bayes Theorem to our data it represents as follows.

- P(H) is the prior probability of a patient's length of stay (LOS).
- P(E) is the probability of a feature variable.
- P(E|H) is the probability of a patient's LOS given that the features are true.
- P(H|E) is the probability of the features given that patient's LOS is true.

Model is trained using Gaussian Naïve Bayes classifier, partitioned train data is fed to the model in array format then the trained model is validated using validation data. This model gives an accuracy score of 34.55% after validating.

Model 2 – XGBoost

Boosting is a sequential technique that works on the principle of an ensemble. At any instant T, the model outcomes are weighed based on the outcomes of the previous instant (T -1). It combines the set of weak learners and improves prediction accuracy. Tree ensemble is a set of classification and regression trees. Trees are grown one after another, and they try to reduce the misclassification rate. The final prediction score of the model is calculated by summing up each and individual score.

Before feeding train data to the XGB Classifier model, booster parameters must be tuned. Tunning the model can prevent overfitting and can yield higher accuracy.

In this XGBoost model, we have used the following parameters for tunning,

- **learning_rate = 0.1** - step size shrinkage used to prevent overfitting. After each boosting step, we can directly get the weights of new features, and eta shrinks the feature weights to make the boosting process more conservative.
- **max_depth = 4** – Maximum depth of the tree. This value describes the complexity of the model. Increasing its value results in overfitting.
- **n_estimators = 800** – Number of gradient boosting trees or rounds. Each new tree attempts to model and correct for the errors made by the sequence of previous trees. Increasing the number of trees can yield higher accuracy but the model reaches a point of diminishing returns quickly.
- **objective = 'multi: softmax'** – this parameter sets XGBoost to do multiclass classification using the softmax objective because the target variable has 11 Levels.
- **reg_alpha = 0.5** - L1 regularization term on weights. Increasing this value will make the model more conservative.
- **reg_lambda = 1.5** - L2 regularization term on weights and is smoother than L1 regularization. Increasing this value will model more conservative.
- **min_child_weight = 2** - Minimum sum of instance weight needed in a child.

Once the model was trained and validated, it yields an accuracy score of 43.04%. When compared to the Naïve Bayes model that is an 8.5% improvement.

Model 3 – Neural Networks

Neural Networks are built of simple elements called neurons, which take in a real value, multiply it by weight, and run it through a non-linear activation function. The process records one at a time and learns by comparing their classification of the record with the known actual classification of the record. The errors from the initial classification of the first record are fed back into the network and used to modify the network's algorithm for further iterations.

In this neural network model, there are six dense layers, the final layer is an output layer with an activation function "SoftMax."

SoftMax is used here because each patient must be classified in one of the 11 levels in the Stay variable. In this model, increasing the number of neurons from each layer to the other layer, will increase the hypothetical space of the model and try to learn more patterns from the data. There are a total of 442,571 trainable parameters. Every layer is activated using "relu" activation function because it overcomes the vanishing gradient problem, allowing models to learn faster and perform better.

Before training the model, data were scaled, converted into a sparse matrix, and portioned into 80% as a train set and 20% as a test set. This neural network model was compiled using "categorical_crossentropy" as a function of loss because the target variable is categorical and "SGD" as an optimizer argument. Initially, the model was trained using portioned train data with 20 epochs and validation set argument set at 20%.

Finally, evaluating the model with a test set yields an accuracy score of 42.05%. Neural Networks supposedly performs better than any other models. But because of the smaller dataset, it was not able to learn more accurately than the XGBoost model.

Prediction and Result:

In the Naïve Bayes model, patients are more likely to be misclassified. This model is biased towards the duration of 21-30 days, it has classified 72,206 patients for this level.

Length of Stay	Predicted Observations from Naïve Bayes	Predicted Observations from XGBoost	Predicted Observations from Neural Network
0-10 Days	2598	4373	4517
11-20 Days	26827	39337	35982
21-30 Days	**72206**	58261	61911
31-40 Days	15639	12100	8678
41-50 Days	469	61	26
51-60 Days	13651	19217	21709
61-70 Days	92	16	1
71-80 Days	955	302	248
81-90 Days	296	1099	1165
91-100 Days	2	78	21
More than 100 Days	4322	2213	2799

Table 3.3: Number of observations from all models

Whereas the other two models XGBoost and Neural Networks are predicting mostly similar Length of Stay for the patient, we can see this similarity for the first five cases in Table 3.3. In Table 3.3, we can see that the observations classified by both these models are marginally similar.

case_id	**LOS predicted from Naïve Bayes**	**LOS predicted from XGBoost**	**LOS predicted from Neural Networks**
318439	21-30	0-10	1-10
318440	51-60	51-60	51-60
318441	21-30	21-30	21-30
318442	21-30	21-30	21-30
318443	31-40	51-60	51-60

Table 3.4: Predicted LOS for first five cases from different models

Examining these predictions, many of the patients are staying in the hospital for 21-30 days and very few people are staying for 61-70 days. As far as the distribution of Length of Stay is concerned, 13% of the patients are discharged from the hospital within 20 days and 1% of the overall patients are staying in the hospital for more than 60 days.

Future Insights:

• Smart Staffing & Personnel Management: having a large volume of quality data helps health care professionals in allocating resources efficiently. Healthcare professionals can analyse the outcomes of check-ups among individuals in various demographic groups and determine what factors prevent individuals from seeking treatment.

• Advanced Risk & Disease Management: Healthcare institutions can offer accurate, preventive care. Effectively decreasing hospital admissions by digging into insights such as drug type, conditions, and the duration of patient visits, among many others.

• Real-time Alerting: Clinical Decision Support (CDS): applications in hospitals analyses patient evidence on the spot, delivering recommendations to health professionals when they make prescriptive choices. However, to prevent unnecessary in-house procedures, physicians prefer people to stay away from hospitals

• Enhancing Patient Engagement: Every step they take, heart rates, sleeping habits, can be tracked for potential patients (who use smart wearables). All this information can be correlated with other trackable data to identify potential health risks.

Conclusion:

In this project, different variables were analysed that correlate with Length of Stay by using patient-level and hospital-level data.

By predicting a patient’s length of stay at the time of admission helps hospitals to allocate resources more efficiently and manage their patients more effectively. Identifying factors that associate with LOS to predict and manage the number of days patients stay, could help hospitals in managing resources and in the development of new treatment plans. Effective use of hospital resources and reducing the length of stay can reduce overall national medical expenses.