HOSPITAL RESOURCE MANAGEMENT AND INPATIENT'S LENGTH OF STAY ANALYSIS AND PREDICTION USING MACHINE LEARNING TECHNIQUES

A PROJECT REPORT

Submitted by

E .SELVA KUMAR (710419104044)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

CHRIST THE KING ENGINEERING COLLEGE, KARAMADAI

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2023

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that report **"HOSPITAL RESOURCE MANAGEMENT AND INPATIENT'S LENGTH OF STAY ANALYSIS AND PREDICTION USING MACHINE LEARNING"** is the bonafide work of **E SELVA KUMAR (710419104044),** who carried out the project work under my supervision.

SIGNATURE	SIGNATURE
Prof. Dr. N.R .GAYATHRI M.E.,PhD.,	**Prof. Mr.T.B.DHARMARAJ M.E.,(PhD.,)**
HEAD OF THE DEPARTMENT	SUPERVISOR
Associate Professor	Associate Professor
Department of CSE	Department of CSE
Christ the King Engineering College	Christ the King Engineering College

Submitted for Anna university project viva- voice examination held on _____2023

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are very grateful and gifted in taking up this opportunity to thank the **LORD ALMIGHTY** for showering his unlimited blessings upon us.

We wish to express our sincere thanks to our beloved Administrator **Rev. Fr.Xavier Manoj DMI** and our beloved Principal **Dr.M.Jeyakumar M.E., Ph.D.,** for their stable and ethical support encouragement towards our project work.

We express our deep sense of gratitude to our esteemed Head of Department **Prof. Dr.N.R.Gayathri M.E., PhD.,** of **Computer Science and Engineering** for his scintillating discussion and encouragement towards our project work.

We are immensely pleased to thank our project guide, **Prof. Mr.T.B.Dharmaraj M.E., (PhD.,)** Department of **Computer Science and Engineering** for his valuable guidance and support throughout this project. His insights and expertise have been instrumental in shaping this project and bringing it to fruition. I would also like to thank all the **Teaching** and **Non-Teaching staffs** for their contributions to this project.

We thank our **family member** and **friends** for their **honorable support.**

E.SELVA KUMAR

(710419104044)

ABSTRACT

The goal of this project is to predict the length of stay for patients in a hospital using machine learning techniques. Accurately predicting patient stay duration can help hospitals optimize resource utilization, improve patient outcomes, and reduce healthcare costs. The project will involve analyzing a large dataset of patient information, including demographics, medical history, diagnoses, procedures, and medication orders. The dataset will be preprocessed and feature engineered to create a set of relevant predictors for the machine learning model. Several machine learning algorithms will be evaluated to determine the best performing model for predicting patient stay duration. The performance of the models will be evaluated using various metrics, including r2_score, mean_square_error, root_square_error and mean_absolute_error. The project will also involve interpreting the results of the machine learning models to gain insights into the factors that contribute to longer or shorter patient stays. These insights can help hospitals identify areas for improvement in patient care and resource utilization. Overall, the hospital stay prediction using machine learning project aims to develop an accurate and interpretable model for predicting patient stay duration that can help hospitals improve patient outcomes and reduce healthcare costs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATIONS

Sl.No.	ABBREVATION	EXPANSION
1.	SVR	Support Vector Regression
2.	LR	Liner Regression
3.	PCA	Principal Components Analysis
4.	RFR	Randaom Forest Regression
5.	LGBMR	LightGBM Regression
6.	DTR	Decision Tree Regression
7.	GBR	Gradient Boosting Regression

CHAPTER – 1

INTRODUCTION

1.1 General

The hospital stay prediction machine learning project is aimed at predicting the length of stay of a patient in a hospital using machine learning algorithms. The length of stay in a hospital is an important metric for healthcare providers as it affects the hospital's capacity to accommodate new patients and also has a financial impact on the patient and the hospital.

By predicting the length of stay, hospitals can better plan their resources and allocate beds and staff efficiently. This project involves using historical patient data, such as medical history, demographic information, lab results, and other relevant features, to train a machine learning model that can accurately predict the length of stay for future patients. The goal is to develop a model that can help hospitals optimize their operations and provide better patient care.

1.2 Aim

The length of hospital stays can have a significant impact on patients' health outcomes and hospital costs. Accurately predicting a patient's length of stay can help hospitals better manage their resources, plan for patient care, and improve patient outcomes. Therefore, the problem statement for this project is to predict the length of a patient's hospital stay based on various medical factors.

1.3 Objective

The objective of a hospital stay prediction project in machine learning is to develop a model that can predict the length of a patient's hospital stay based on various factors such as demographics, medical history, laboratory results, and vital signs. The goal is to improve patient care by providing accurate estimates of the expected length of hospitalization, which can help hospitals allocate resources more effectively and plan for patient discharge.

Some specific objectives of a hospital stay prediction project may include :

Improving patient outcomes: By accurately predicting the length of a patient's hospital stay, healthcare providers can better plan and coordinate care, resulting in improved patient outcomes.

Reducing healthcare costs: Hospital stays are a significant cost driver in healthcare. Accurate predictions can help hospitals better manage patient flow, optimize resource allocation, and reduce unnecessary hospitalizations, resulting in cost savings.

Supporting clinical decision-making: Hospital stay prediction models can provide valuable insights for clinical decision-making, such as identifying patients at risk of longer stays or complications.

Overall, the goal of a hospital stay prediction project is to improve patient care, reduce healthcare costs, and enhance hospital efficiency by leveraging machine learning to predict the length of a patient's hospital stay.

1.4 Scope

The scope of inpatient length of stay (LOS) prediction in hospitals using machine learning is quite significant. Predicting the length of stay for patients admitted to a hospital can provide valuable insights and benefits in various aspects, including healthcare management, resource allocation, and patient care. Here are some key areas where machine learning-based LOS prediction can be beneficial:

Resource Planning and Management: Accurate prediction of inpatient LOS can help hospitals optimize resource allocation, such as beds, staff scheduling, and medical supplies. By estimating the length of stay, hospitals can effectively manage resources and improve operational efficiency.

Patient Flow and Bed Occupancy: LOS prediction enables hospitals to better manage patient flow by estimating how long a patient is likely to stay. This information helps with bed allocation, discharge planning, and reducing wait times for new admissions.

Care Coordination and Discharge Planning: Predicting the length of stay allows healthcare providers to plan and coordinate the patient's care more effectively. It enables proactive discharge planning, identifying patients who are likely to have extended stays and providing appropriate interventions and resources to facilitate their timely discharge.

Quality of Care: Accurate LOS prediction can contribute to improving the quality of care provided to patients. It helps healthcare professionals identify patients who may be at risk of extended stays or complications, allowing for early intervention and tailored care plans.

Resource Optimization and Cost Reduction: With better LOS prediction, hospitals can optimize resource utilization, leading to cost reduction. By accurately estimating patient lengths of stay, unnecessary resource overutilization can be minimized, resulting in more efficient operations and potential cost savings.

Machine learning techniques, such as regression models, time series analysis, and deep learning models, can be applied to LOS prediction. These models can leverage various patient-specific features, such as demographic information, medical history, laboratory results, vital signs, and procedures, to make accurate predictions.

It's important to note that LOS prediction models should be developed and validated using large and diverse datasets to ensure generalizability. Additionally, ethical considerations, data privacy, and compliance with regulatory requirements should be taken into account when implementing machine learning solutions in healthcare settings.

Overall, the scope of inpatient LOS prediction using machine learning is broad and holds potential for improving hospital operations, patient care, and resource management in healthcare facilities.

CHAPTER 2

REVIEW OF LITERATURE

2.1 Introduction

The length of hospital stay and its implications have a significant economic and human impact. As a consequence, the prediction of that key parameter has been subject to previous research in recent years. Most previous work has analysed length of stay in particular hospital departments within specific study groups, which has resulted in successful prediction rates, but only occasionally reporting predictive patterns. In this work we report a predictive model for length of stay (LOS) together with a study of trends and patterns that support a better understanding on how LOS varies across different hospital departments and specialties. We also analyse in which hospital departments the prediction of LOS from patient data is more insightful. After estimating predictions rates, several patterns were found; those patterns allowed, for instance, to determine how to increase prediction accuracy in women admitted to the emergency room for enteritis problems. Overall, concerning these recognised patterns, the results are up to 21.61% better than the results with baseline machine learning algorithms in terms of error rate calculation, and up to 23.83% in terms of success rate in the number of predicted which is useful to guide the decision on where to focus attention in predicting LOS.

2.2 Existing system

Inpatient length of stay prediction using machine learning is a valuable application in healthcare that can help hospitals and healthcare providers optimize resource allocation, improve patient care, and enhance operational efficiency. There have been several approaches and models developed for predicting the length of stay for inpatients. Here are a few common techniques and methods used in the existing systems.

Regression models are widely used for length of stay prediction. Techniques such as linear regression, multiple regression, and logistic regression are employed to establish a relationship between input features and the length of stay. These models can handle both continuous and categorical variables, and they provide insights into the impact of each feature on the length of stay.

Decision trees are effective for length of stay prediction due to their interpretability and ability to handle complex feature interactions. Decision tree-based algorithms, such as Random Forest and Gradient Boosting, can capture non-linear relationships between predictors and length of stay. They can handle various types of features, including categorical and numerical variables.

Time series analysis techniques can be applied when temporal information is available, allowing predictions to be made based on historical patterns. Methods such as autoregressive integrated moving average (ARIMA) and seasonal decomposition of time series (STL) can capture trends, seasonality, and cyclic patterns in the length of stay data.

Neural Networks:Deep learning models, particularly recurrent neural networks (RNNs) and long short-term memory (LSTM) networks, have shown promise in length of stay prediction. These models can effectively capture sequential dependencies in patient data and handle variable-length input sequences, such as time series or electronic health records.

Ensemble methods combine predictions from multiple models to improve accuracy and robustness. Techniques like model averaging, stacking, and boosting can be applied to length of stay prediction. For example, an ensemble of regression models, decision trees, and neural networks can provide a more accurate prediction by leveraging the strengths of each individual model.

Furthermore, it's essential to train and validate these models using robust datasets with diverse patient populations to ensure generalizability and reliability. Local regulations and privacy concerns should also be considered when developing and deploying these prediction systems in real-world hospital settings.

2.2.1 Disadvantages

To address these disadvantages, ongoing research and development are necessary to improve data quality, model interpretability, generalizability, fairness, and integration with existing healthcare systems. Additionally, collaboration between machine learning experts and healthcare professionals is vital to ensure that the predictions align with clinical needs and support informed decision-making.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In this project using the different type of regression algorithm (LinearRegression, KNearestNeighbors Regressor, Linear SVR, SVR,DecisionTreeRegressor,RandomForestRegressor,GradientBoostingRegressor, LGBMRegressor).

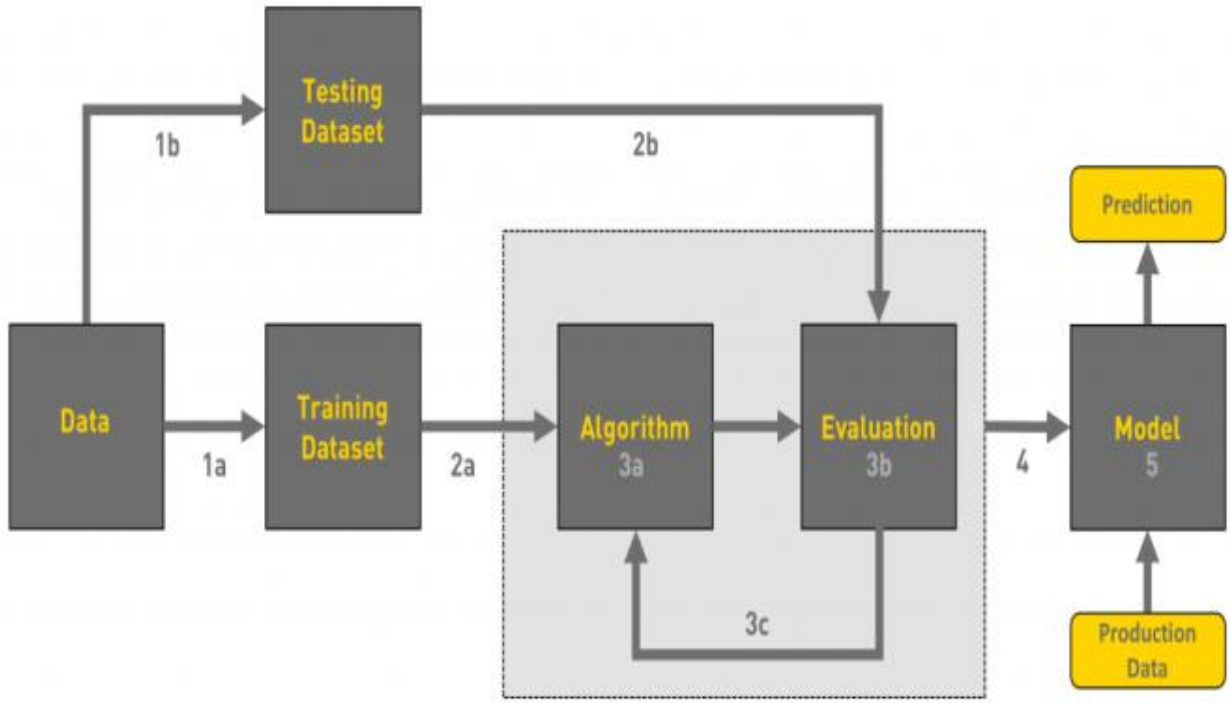


Figure: 3.1

Gather data: The dataset for this project will be obtained from various sources such as electronic medical records, administrative databases, and patient surveys. The dataset will include information such as ‘Location', 'Time', 'Hospital_Stay', 'MRI_Units', 'CT_Scanners', 'Hospital_Beds' to predict the patient stay in the hospital.

Preprocess data: The dataset will be preprocessed to clean the data and prepare it for analysis. This will include removing any missing or irrelevant data, converting categorical variables into numerical values, and scaling the data to ensure that all variables have a similar range.

Split data: Split the data into training and testing sets. Typically, a split of 70% training and 30% testing is used.Feature engineering plays a crucial role in length of stay prediction. Relevant features can include demographic information, medical history, diagnostic codes, laboratory results, and vital signs. Feature engineering techniques, such as feature selection, dimensionality reduction, and feature transformation, help identify the most informative and relevant predictors.

Train the model: Train a regression model on the training data. The goal is to find the coefficients of the linear equation that best fit the data.

Evaluate the model: The performance of the trained model will be evaluated using cross-validation techniques and other statistical measures. The model will also be tested on a separate dataset to ensure that it can generalize to new data and make accurate predictions. It's important to note that the choice of machine learning technique and model depends on the specific dataset, available features, and the desired accuracy. The performance of the prediction system can be evaluated using metrics such as mean absolute error (MAE), root mean squared error (RMSE), or R-squared (R^2) to assess the accuracy of the length of stay predictions.

Use the model: Once the model is trained and evaluated, it can be used to predict the length of stay for new patients based on their features.

CHAPTER – 4

TOOLS AND SOFTWARE

There are many tools and software used in machine learning, ranging from programming languages and libraries to specialized platforms and frameworks. Here are some of the most popular tools and software used in machine learning.

4.1 Programming Languages

Python is a popular programming language for machine learning due to its simplicity, readability, and vast collection of libraries and frameworks. Here are some of the key libraries and frameworks used in machine learning with Python.

Python has a thriving ecosystem of tools and libraries that make it a powerful choice for machine learning projects. Its ease of use and versatility make it accessible to beginners while providing advanced features for experienced users.

4.2 Data Analysis Libraries

Libraries such as Pandas, NumPy, and Scikit-learn can be used for data preprocessing, feature selection, model training, and evaluation.

4.2.1 Pandas

Pandas is an open-source Python library that is widely used for data manipulation and analysis. It provides data structures for efficiently storing and manipulating large datasets, as well as a wide range of tools for data cleaning, transformation, and analysis.Here are some of the key features of Pandas:

Data structures: Pandas provides two primary data structures for storing data - Series and DataFrame. A Series is a one-dimensional array-like object that can hold any data type, such as integers, floating-point numbers, strings, and Python objects. A DataFrame is a two-dimensional table-like structure that contains rows and columns of data.

Data cleaning and transformation: Pandas provides a range of tools for data cleaning and transformation, such as filling missing values, filtering rows based on conditions, selecting and manipulating columns, and converting data types.

Data analysis: Pandas provides a range of tools for data analysis, such as groupby operations, aggregation, and pivot tables. It also provides tools for merging, joining, and reshaping datasets.

Data input and output: Pandas can read and write data in a variety of formats, including CSV, Excel, SQL databases, and JSON.

Integration with other libraries: Pandas integrates well with other Python libraries, such as NumPy, Matplotlib, and Scikit-learn, making it a powerful tool for data analysis and machine learning.

Overall, Pandas is a powerful and flexible tool for data manipulation and analysis in Python. Its intuitive and easy-to-use interface makes it accessible to users of all levels, from beginners to advanced data analysts and data scientists.

4.2.1 Numpy

NumPy is a popular Python library used for numerical computing and scientific computing. It provides a powerful array object for working with multidimensional arrays and matrices, as well as a range of tools for performing mathematical and statistical operations on arrays.Here are some of the key features of NumPy:

Ndarray: The ndarray object is the core of NumPy. It provides an efficient and flexible container for storing large multidimensional arrays and matrices, and supports a wide range of operations on these arrays, such as indexing, slicing, and reshaping.

Mathematical and statistical operations: NumPy provides a range of tools for performing mathematical and statistical operations on arrays, such as addition, subtraction, multiplication, division, and trigonometric functions. It also provides tools for statistical analysis, such as mean, median, standard deviation, and correlation coefficients.

Broadcasting: Broadcasting is a powerful feature of NumPy that allows arithmetic operations to be performed on arrays of different shapes and sizes. This can greatly simplify code and make it more readable.

Integration with other libraries: NumPy integrates well with other Python libraries, such as Matplotlib, Scipy, and Pandas, making it a powerful tool for scientific computing and data analysis.

Performance: NumPy is designed to be fast and efficient, and provides optimized routines for performing many common mathematical and statistical operations. It also supports parallel computing, allowing operations to be performed in parallel on multiple processors or cores.

Overall, NumPy is a powerful and versatile library for numerical computing and scientific computing in Python. Its efficient array object and powerful tools make it a popular choice for data analysis, machine learning, and scientific computing applications.

4.2.3 Scikit-learn

Scikit-learn is a popular open-source Python library used for machine learning and data analysis. It provides a range of tools for building and training machine learning models, as well as tools for data preprocessing, feature selection, and model evaluation.Here are some of the key features of Scikit-learn:

Supervised learning: Scikit-learn provides a range of tools for supervised learning, such as regression, classification, and ensemble methods. It also provides tools for feature selection, such as recursive feature elimination and feature importance.

Unsupervised learning: Scikit-learn provides a range of tools for unsupervised learning, such as clustering, dimensionality reduction, and density estimation.

Data preprocessing: Scikit-learn provides tools for data preprocessing, such as scaling, normalization, and missing value imputation.

Model evaluation: Scikit-learn provides tools for model evaluation, such as cross-validation, grid search, and learning curves. It also provides tools for model selection, such as model comparison and model stacking.

Integration with other libraries: Scikit-learn integrates well with other Python libraries, such as NumPy, Pandas, and Matplotlib, making it a powerful tool for data analysis and machine learning.

Performance: Scikit-learn is designed to be fast and efficient, and provides optimized routines for performing many common machine learning tasks. It also supports parallel computing, allowing operations to be performed in parallel on multiple processors or cores.

Overall, Scikit-learn is a powerful and flexible library for machine learning and data analysis in Python. Its intuitive and easy-to-use interface makes it accessible to users of all levels, from beginners to advanced data analysts and data scientists. It is widely used in industry and academia for a variety of applications, such as predictive modeling, data mining, and natural language processing.

4.3 Visualization Libraries

Libraries such as Matplotlib and Seaborn can be used to visualize the data, explore correlations and trends, and generate insights.

4.3.1 Matplotlib

Matplotlib is a popular Python library used for data visualization and graphical plotting. It provides a range of tools for creating high-quality plots, charts, and figures for scientific and technical applications.Here are some of the key features of Matplotlib:

Data visualization: Matplotlib provides a range of tools for data visualization, such as line plots, scatter plots, bar charts, histograms, and heatmaps. It also provides tools for visualizing geographic data, network graphs, and 3D plots.

Customization: Matplotlib provides a wide range of options for customizing plots and figures, such as changing colors, line styles, markers, and fonts. It also provides tools for adding labels, titles, annotations, and legends to plots.

Integration with other libraries: Matplotlib integrates well with other Python libraries, such as NumPy, Pandas, and Scikit-learn, making it a powerful tool for data analysis and machine learning.

Output formats: Matplotlib can output plots and figures in a range of formats, such as PDF, SVG, PNG, and JPEG. It can also generate interactive plots for web applications and Jupyter notebooks.

Performance: Matplotlib is designed to be fast and efficient, and can handle large datasets and complex visualizations. It also supports parallel computing, allowing multiple plots to be generated in parallel on multiple processors or cores.

Overall, Matplotlib is a powerful and versatile library for data visualization and graphical plotting in Python. Its intuitive and easy-to-use interface makes it accessible to users of all levels, from beginners to advanced data analysts and data scientists. It is widely used in industry and academia for a variety of applications, such as data analysis, machine learning, and scientific computing.

4.3.2 Seaborn

Seaborn is a popular data visualization library in Python that is built on top of Matplotlib. It provides a high-level interface for creating informative and attractive statistical graphics.Here are some of the key features of Seaborn:

Statistical data visualization: Seaborn provides a range of tools for visualizing statistical data, such as bar plots, box plots, violin plots, and heatmaps. It also provides tools for visualizing relationships between variables, such as scatter plots, line plots, and regression plots.

Data exploration: Seaborn provides tools for exploring relationships between variables and identifying patterns in data, such as pair plots and joint plots.

Customization: Seaborn provides a range of options for customizing plots and figures, such as changing colors, line styles, markers, and fonts. It also provides tools for adding labels, titles, annotations, and legends to plots.

Integration with other libraries: Seaborn integrates well with other Python libraries, such as Pandas and Scikit-learn, making it a powerful tool for data analysis and machine learning.

Performance: Seaborn is designed to be fast and efficient, and can handle large datasets and complex visualizations. It also supports parallel computing, allowing multiple plots to be generated in parallel on multiple processors or cores.

Overall, Seaborn is a powerful and versatile library for data visualization and statistical graphics in Python. Its high-level interface and attractive default styles make it easy to create informative and appealing visualizations. It is widely used in industry and academia for a variety of applications, such as data analysis, machine learning, and scientific computing.

4.4 Integrated Development Environments (IDEs):

4.4.1 Jupiter Notebook:

Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It supports various programming languages such as Python, R, Julia, and others. Jupyter Notebook is a popular tool for data analysis, scientific computing, and machine learning.Here are some of the key features of Jupyter Notebook:

Interactive computing: Jupyter Notebook provides an interactive computing environment that allows users to write and execute code, visualize data, and explore results in real-time. It supports many popular Python libraries such as NumPy, Pandas, Matplotlib, and Scikit-learn.

Markdown support: Jupyter Notebook supports markdown, which allows users to create rich text documents with headings, lists, tables, and images. Markdown cells can also contain LaTeX equations for mathematical notation.

Code sharing: Jupyter Notebook allows users to share their notebooks with others, making it easy to collaborate on code and data analysis. Notebooks can be shared as files or hosted on online services such as GitHub, Azure, and Google Colab.

Visualization: Jupyter Notebook supports inline visualization with Matplotlib, Seaborn, Plotly, and other popular Python libraries. This allows users to create interactive and high-quality visualizations directly in the notebook.

Extensions: Jupyter Notebook has a large and active community of developers who have created many useful extensions and plugins. These extensions provide additional functionality such as code linting, code formatting, and keyboard shortcuts.

Overall, Jupyter Notebook is a powerful tool for interactive computing, data analysis, and machine learning. Its ease of use, flexibility, and ability to integrate with many popular Python libraries make it a popular choice for researchers, data scientists, and educators.

CHAPTER - 5

ALGORITHM

The regression algorithm for predicting the length of patient stay in a hospital could be Linear Regression, which aims to find a linear relationship between the patient's features and their length of stay.

We have dataset of the Inpatient patient stay in the hospital details is linearly increasing the features and target value. If the patient had a viral fever and cold they just stay in the hospital for one week. In case one another patient admitted the hospital for emergency situation, they have a heart attack or any dangerous means that patient stay in the hospital and some need depends upon the disease. It is vary from one to one but the patient had a high level disease means they stay long in the hospital else the patient in small problem means they stay 2 or 3 days. So, this length of stay is depends on the patient problem in linear relationship.

5.1 Linear Regression

Linear regression is a machine learning algorithm used to predict a numerical target variable based on one or more input variables. It is a supervised learning algorithm that uses a linear relationship between the input variables and the output variable to make predictions. In mathematical terms, linear regression can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of linear regression is to find a set of weights w = [w0, w1, w2, ..., wn] such that the linear equation y = w0 + w1x1 + w2x2 + ... + wnxn best approximates the relationship between X and y.

The linear equation can be rewritten in matrix notation as y = Xw, where X is the input matrix, y is the target vector, and w is the weight vector. The goal of linear regression is to find the weight vector w that minimizes the sum of squared errors between the predicted values and the actual values of y.The sum of squared errors is defined as follows:

SSE = (y - Xw)^T(y - Xw)

where ^T denotes the transpose of a matrix.

The weight vector w can be found by solving the normal equations, which are given by:

$$X^T Xw = X^T y$$

where X^T is the transpose of the input matrix X.

Once the weight vector w is computed, the model can be used to make predictions for new input values by simply computing the dot product of the weight vector and the input vector.

Linear regression is a simple and widely used algorithm for predicting numerical values. It is easy to implement and can be extended to handle more complex relationships between the input variables and the target variable, such as polynomial regression and multiple regression.

5.2 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) regression is a machine learning algorithm used to predict a numerical target variable based on one or more input variables. It is a supervised learning algorithm that uses the K-nearest neighbors of a new input point to predict its target value. In mathematical terms, KNN regression can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of KNN regression is to predict the target value y* for a new input point x* by finding the K closest neighbors in the training set and computing the average of their target values.

To find the K closest neighbors, we first define a distance metric d(x, x') that measures the distance between two input points x and x'. A common distance metric used in KNN regression is Euclidean distance, which is given by:

$$d(x, x') = \sqrt{(x1 - x'1)^2 + (x2 - x'2)^2 + ... + (xn - x'n)^2}$$

Once we have defined the distance metric, we can find the K nearest neighbors of the new input point x* by computing the distance between x* and each point in the training set, and selecting the K points with the smallest distances.

Finally, we compute the predicted target value y* for the new input point x* by taking the average of the target values of its K nearest neighbors:

$$y^* = (1/K) * \text{sum}(y_i)$$

where y_i is the target value of the i-th nearest neighbor.

KNN regression is a simple and intuitive algorithm that can be used for both regression and classification tasks. It is non-parametric, which means that it does not make any assumptions about the underlying distribution of the data. However, KNN regression can be computationally expensive, especially for large datasets and high-dimensional input spaces. Additionally, the choice of the distance metric and the value of K can have a significant impact on the performance of the algorithm.

5.3 Support Vector Machine Linear kernel (SVM)

Support Vector Machine (SVM) is a machine learning algorithm used for classification and regression tasks. SVM with a linear kernel is a variant of the SVM algorithm that uses a linear decision boundary to separate the classes in the input space. In mathematical terms, SVM with a linear kernel can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, where y = {-1, 1}, the goal of SVM with a linear kernel is to find a hyperplane that best separates the two classes in the input space.

A hyperplane is a linear decision boundary that separates the input space into two regions. In a two-dimensional input space, a hyperplane is a line; in a three-dimensional input space, it is a plane, and in higher dimensions, it is a hyperplane. The hyperplane is defined by a weight vector w and a bias term b, such that the decision boundary is given by the equation:

$$w^T x + b = 0$$

where ^T denotes the transpose of a matrix.

The distance between a point x and the hyperplane is given by:

$$\text{distance} = |w^T x + b| / ||w||$$

where ||w|| is the Euclidean norm of the weight vector.

The SVM algorithm aims to find the weight vector w and the bias term b that minimize the classification error and maximize the margin, which is the distance between the hyperplane and the closest data points from each class. The margin is given by:

margin = 2 / ||w||

The SVM optimization problem can be expressed as:

minimize 1/2 * ||w||^2

subject to y_i (w^T x_i + b) >= 1

where y_i is the target value of the i-th input point and x_i is the i-th input vector.

This is a quadratic programming problem that can be solved using various optimization algorithms, such as Sequential Minimal Optimization (SMO).

Once the weight vector w and the bias term b are computed, the SVM model can be used to make predictions for new input values by simply computing the sign of the decision boundary:

y* = sign(w^T x + b)

where y* is the predicted target value for the new input point x.

SVM with a linear kernel is a powerful and widely used algorithm for binary classification tasks. It is efficient, scalable, and can handle high-dimensional input spaces. However, it may not perform well on nonlinearly separable data, in which case a kernel trick can be used to map the input space to a higher-dimensional feature space where the data is linearly separable.

5.4 Support Vector Regression

Support Vector Machine (SVM) is a machine learning algorithm used for classification and regression tasks. Support Vector Regression (SVR) is a variant of SVM that is used for regression tasks. The goal of SVR is to find a function that approximates the

mapping from input variables to target variables. In mathematical terms, SVR can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of SVR is to find a function f(x) that approximates the mapping from X to y, such that the error between the predicted value f(x) and the true value y is minimized.

The SVR algorithm uses a linear or nonlinear kernel function to map the input space to a higher-dimensional feature space, where the data is more easily separable. The kernel function is a similarity measure that computes the dot product between two feature vectors in the higher-dimensional space.

In SVR, the function f(x) is defined as a linear combination of the kernel function and a set of coefficients alpha:

$$f(x) = \text{sum}(\text{alpha\_i} * K(x\_i, x)) + b$$

where K(x_i, x) is the kernel function between the i-th training point x_i and the new input point x, and b is a bias term.

The coefficients alpha are computed by solving the following optimization problem:

$$\text{minimize } 1/2 * ||w||^2 + C * \text{sum}(\max(0, |y\_i - f(x\_i)| - \text{epsilon}))$$

where ||w|| is the L2 norm of the weight vector w, C is a regularization parameter that controls the tradeoff between the margin and the error, and epsilon is a tolerance parameter that specifies the size of the margin.

The objective function minimizes the L2 norm of the weight vector w and the error between the predicted value f(x) and the true value y, subject to the constraints that the error is less than or equal to epsilon for all training points.

Once the coefficients alpha and the bias term b are computed, the SVR model can be used to make predictions for new input values by simply computing the function f(x):

$$y^* = f(x)$$

where y* is the predicted target value for the new input point x.

SVR is a powerful and widely used algorithm for regression tasks. It is efficient, scalable, and can handle nonlinearly separable data. However, it requires careful selection of the kernel function and the regularization parameter, which can have a significant impact on the performance of the algorithm.

5.5 Decision Tree Regression

Decision tree regression is a non-parametric algorithm used for both classification and regression tasks. In decision tree regression, the goal is to learn a decision tree that can predict the target value for a new input point based on its features. The decision tree is constructed by recursively splitting the data into smaller subsets, based on the values of the features, until a stopping criterion is reached.

Mathematically, the decision tree regression algorithm can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of decision tree regression is to learn a function f(x) that predicts the target variable for a new input point x. The function f(x) is defined as a series of binary decisions that split the input space into smaller subsets.

The decision tree can be represented as a set of nested if-then statements that split the input space based on the values of the features. Each internal node of the decision tree represents a binary decision, and each leaf node represents a prediction for the target variable.

The decision tree is constructed by recursively splitting the data into smaller subsets, based on the values of the features. The splitting criterion is chosen to maximize the information

gain, which measures the reduction in entropy (or increase in purity) of the target variable due to the split.

At each internal node of the decision tree, a splitting criterion is applied to the data to create two new subsets. The splitting criterion is chosen to maximize the information gain, which is defined as:

information_gain = entropy(parent) - sum(weighted_entropy(child))

where entropy(parent) is the entropy (or impurity) of the target variable for the parent node, and weighted_entropy(child) is the weighted entropy of the target variable for each child node. The weight of each child node is the proportion of the total number of instances that belong to that node.

The entropy of the target variable is calculated as:

$$\text{entropy} = -\text{sum}(p_i * \log2(p_i))$$

where p_i is the proportion of instances that belong to class i.

Once the decision tree is constructed, it can be used to make predictions for new input values by traversing the tree from the root node to a leaf node that corresponds to the predicted target value.

Decision tree regression is a powerful and widely used algorithm for regression tasks. It is flexible, interpretable, and can handle nonlinearly separable data. However, it can be prone to overfitting, especially for high-dimensional data or data with complex interactions between the features. Regularization techniques such as pruning or ensemble methods such as Random Forest can be used to mitigate overfitting and improve the performance of the algorithm.

5.6 Random Forest Regression

Random Forest Regression is an ensemble learning method that combines multiple decision trees to make predictions. Each decision tree in the random forest is trained on a randomly selected subset of the data and a random subset of the features, which helps to reduce overfitting and increase generalization performance.

The mathematical formulation of Random Forest Regression can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of Random Forest Regression is to learn a function f(x) that predicts the target variable for a new input point x. The function f(x) is defined as the average of the predictions of multiple decision trees, each trained on a randomly selected subset of the data and a random subset of the features.

Let T be the number of decision trees in the random forest, and let Ti(x) be the prediction of the ith decision tree for input point x. Then the prediction of the random forest model for input point x is given by:

$$f(x) = 1/T * sum(Ti(x))$$

Each decision tree in the random forest is trained using a modified version of the decision tree algorithm. At each internal node of the decision tree, a splitting criterion is applied to the data to create two new subsets. The splitting criterion is chosen to maximize the information gain, which measures the reduction in variance (or increase in homogeneity) of the target variable due to the split.

The variance of the target variable is calculated as:

$$variance = 1/n * sum((y_i - mean(y))^2)$$

where n is the number of instances in the subset, y_i is the target variable for the ith instance, and mean(y) is the mean target variable for the subset.

Once the decision trees are trained, the random forest can be used to make predictions for new input values by averaging the predictions of all the decision trees.

Random Forest Regression is a powerful and widely used algorithm for regression tasks. It is flexible, interpretable, and can handle high-dimensional data with complex interactions between the features. It is also less prone to overfitting than individual decision trees, due to the use of multiple trees and feature subsampling. However, it can be computationally expensive for large datasets or complex models.

5.7 Gradient Boosting Regression

Gradient Boosting Regression is another popular machine learning algorithm for regression tasks, which also works by combining multiple weak models to form a strong model. The algorithm trains a series of decision trees, where each new tree is trained to correct the residual errors of the previous tree.

The mathematical formulation of Gradient Boosting Regression can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of Gradient Boosting Regression is to learn a function f(x) that predicts the target variable for a new input point x. The function f(x) is defined as the sum of predictions of multiple decision trees, where each tree is trained to correct the residual errors of the previous tree.

Let T be the number of decision trees in the gradient boosting model, and let hi(x) be the prediction of the ith decision tree for input point x. Then the prediction of the gradient boosting model for input point x is given by:

$$f(x) = \text{sum}(hi(x))$$

where hi(x) is defined recursively as:

$$hi(x) = gi(x) + hi\text{-}1(x)$$

where gi(x) is the negative gradient of the loss function with respect to the previous prediction hi-1(x), evaluated at the training data.

The loss function is a measure of the error between the predicted and actual values of the target variable. Common loss functions for regression tasks include the mean squared error (MSE) and the mean absolute error (MAE).

The negative gradient of the loss function with respect to the previous prediction hi-1(x) is calculated as:

$$gi(x) = -dL(y, hi\text{-}1(x)) / dhi\text{-}1(x)$$

where L is the loss function, y is the actual value of the target variable, and hi-1(x) is the previous prediction.

The decision trees in Gradient Boosting Regression are usually shallow, with few nodes and small depth, to prevent overfitting. The trees are trained using a modified version of the decision tree algorithm, where the splitting criterion is chosen to minimize the loss function.

Gradient Boosting Regression is a powerful algorithm for regression tasks, which can achieve high accuracy and is less prone to overfitting than other methods. However, it can be computationally expensive and sensitive to hyperparameter tuning.

5.8 LightGBM Regression

LightGBM is a popular gradient boosting algorithm that is designed to be efficient and scalable for large datasets. It is similar to Gradient Boosting Regression, but uses a different approach to constructing decision trees that allows it to train faster and use less memory.

The mathematical formulation of LightGBM Regression can be expressed as follows:

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of LightGBM Regression is to learn a function f(x) that predicts the target variable for a new input point x. The function f(x) is defined as the sum of predictions of multiple decision trees, where each tree is trained to correct the residual errors of the previous tree.

Let T be the number of decision trees in the LightGBM model, and let hi(x) be the prediction of the ith decision tree for input point x. Then the prediction of the LightGBM model for input point x is given by:

$$f(x) = \text{sum}(hi(x))$$

where hi(x) is defined recursively as:

$$hi(x) = \text{sum}(wj * qj(x))$$

where wj is the weight of the jth leaf node, and qj(x) is a binary indicator function that represents whether input point x belongs to the jth leaf node.

The decision trees in LightGBM are constructed using a technique called gradient-based one-side sampling, which involves splitting the data based on the gradients of the loss function. This allows the algorithm to focus on the most informative data points and avoid overfitting.

The weights wj of the leaf nodes are calculated using a technique called leaf-wise growth, which grows the tree by adding new nodes to the leaf nodes that minimize the loss function. This approach is more efficient than the traditional level-wise growth, which adds new nodes to all levels of the tree.

The binary indicator function qj(x) is calculated using a histogram-based approach, which bins the input variables and uses a histogram of the bin counts to quickly evaluate the function.

LightGBM also includes several advanced features, such as regularization, early stopping, and categorical feature handling, which can improve its performance and reduce overfitting.

Overall, LightGBM is a powerful and efficient algorithm for regression tasks, which can achieve high accuracy and handle large datasets. However, it may require more careful hyperparameter tuning and data preprocessing than other methods.

CHAPTER - 6

IMPLEMENTATION OF PROPOSED METHODOLOGY

1. Gather data: Collect data on past patient stays, including Time, Hospital Beds, CT Scanners, MRI Units, and Hospita of stay.
2. Preprocess data: Clean the data, remove any missing values or outliers, and perform feature scaling if necessary.
3. Split data: Split the data into training and testing sets. Typically, a split of 70% training and 30% testing is used
4. Train the model: Train a regression model on the training data. The goal is to find the coefficients of the linear equation that best fit the data.
5. Evaluate the model: Evaluate the performance of the model on the testing data. Metrics such as mean squared error or R-squared can be used to measure the accuracy of the model.
6. Use the model: Once the model is trained and evaluated, it can be used to predict the length of stay for new patients based on their features.

6.1 Data Collection

The dataset for this project will be obtained from various sources such as electronic medical records, administrative databases, and patient surveys. The dataset will include information such as ‘Location', 'Time', 'Hospital_Stay', 'MRI_Units', 'CT_Scanners', 'Hospital_Beds' to predict the patient stay in the hospital.

6.1.1 Importing the libraries

Import the essential libraries which are used to data analysis and data visualization. Read the comma separated value (‘csv’) for run program.

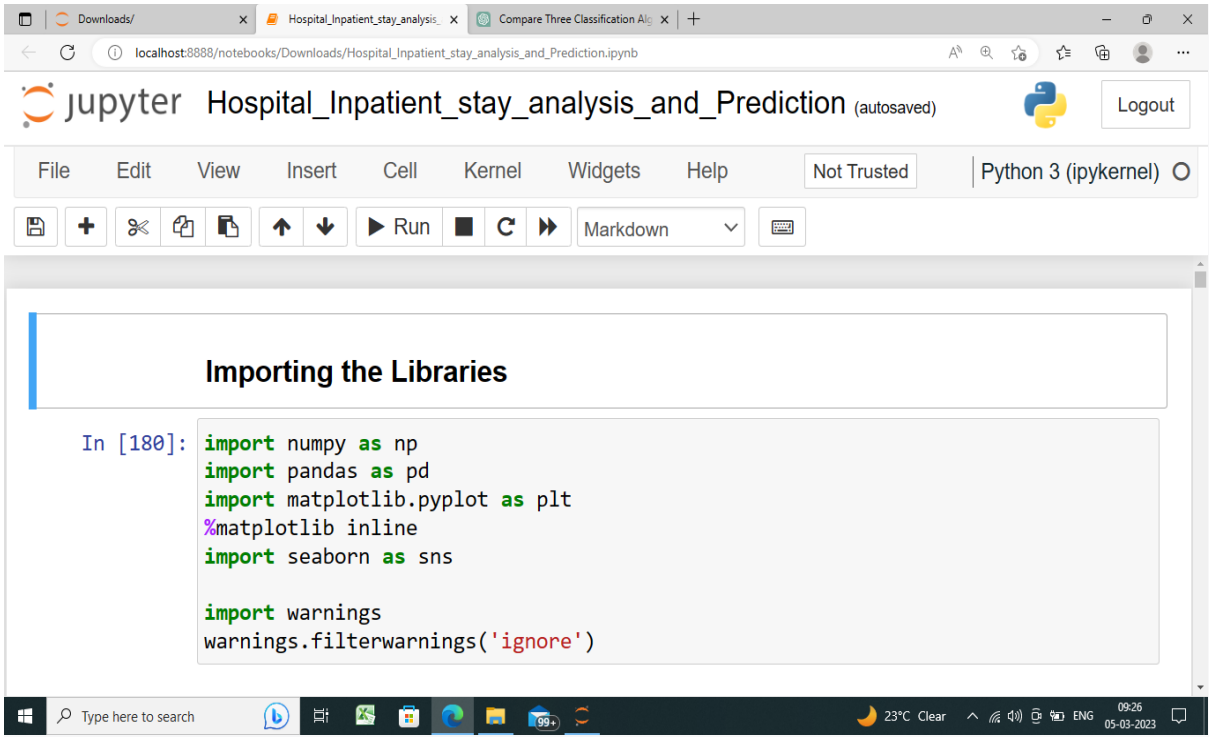


Figure : 6.1

There is an four libraries are import for this exprimment. NumPy is a popular Python library used for numerical computing and scientific computing. Pandas is an open-source Python library that is widely used for data manipulation and analysis. Matplotlib is a popular Python library used for data visualization and graphical plotting. Seaborn is a popular data visualization library in Python that is built on top of Matplotlib. It provides a high-level interface for creating informative and attractive statistical graphics.

6.1.2 Import tools and algorithm from sklearn

Scikit-learn is a popular open-source Python library used for machine learning and data analysis. It provides a range of tools for building and training machine learning models, as well as tools for data preprocessing, feature selection, and model evaluation.

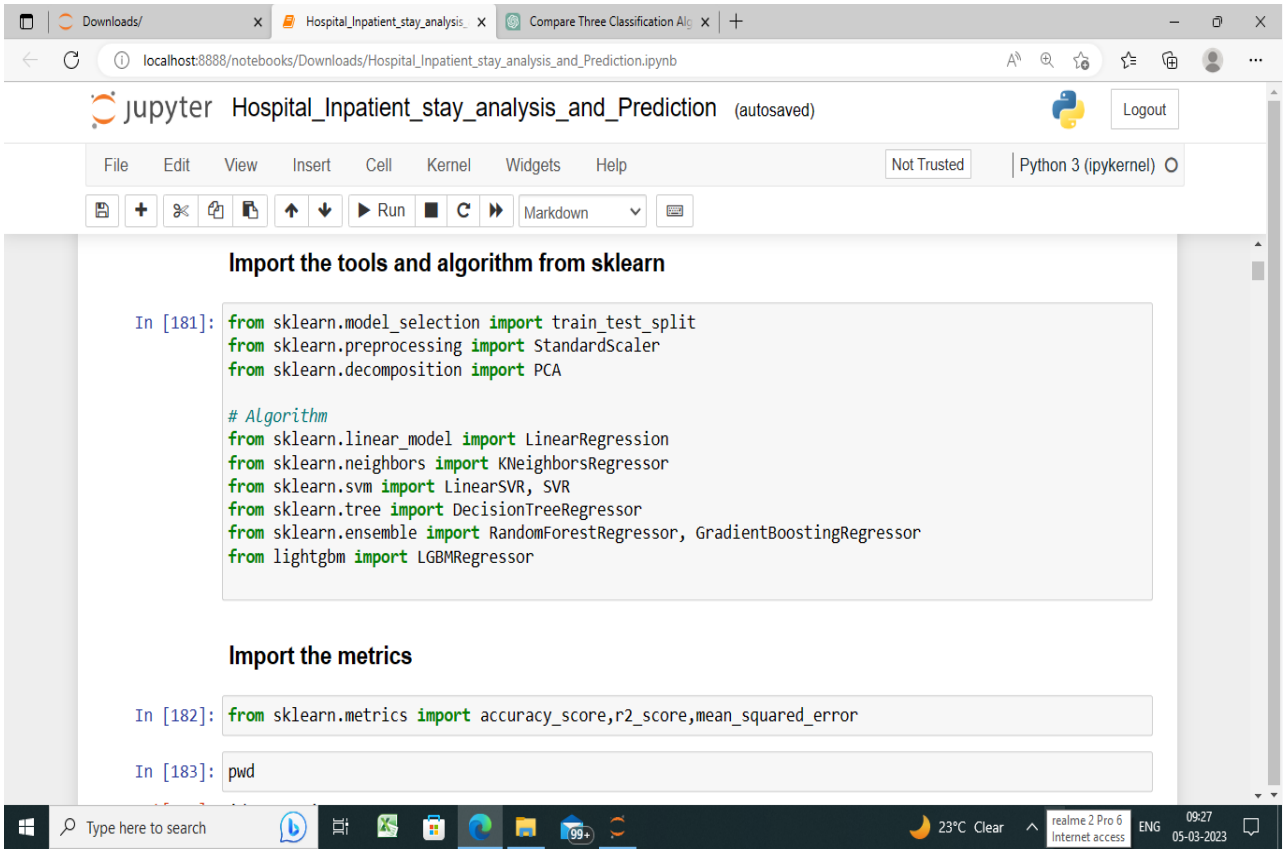


Figure : 6.2

Metrics is a statistical measure that indicates the proportion of variance in the dependent variable that is explained by the independent variables in a regression model. It is a value between 0 and 1, where 1 indicates a perfect fit of the model to the data, and 0 indicates that the model does not explain any of the variance in the dependent variable. A negative value for **r2_score** indicates that the model fits the data worse than a horizontal line.

MSE is a measurement of the average squared difference between the predicted values and the actual values. It is calculated by taking the average of the squared differences between the predicted and actual values for each data point. A smaller MSE indicates that the model is better at predicting the outcome variable.

6.1.3 Read the dataset

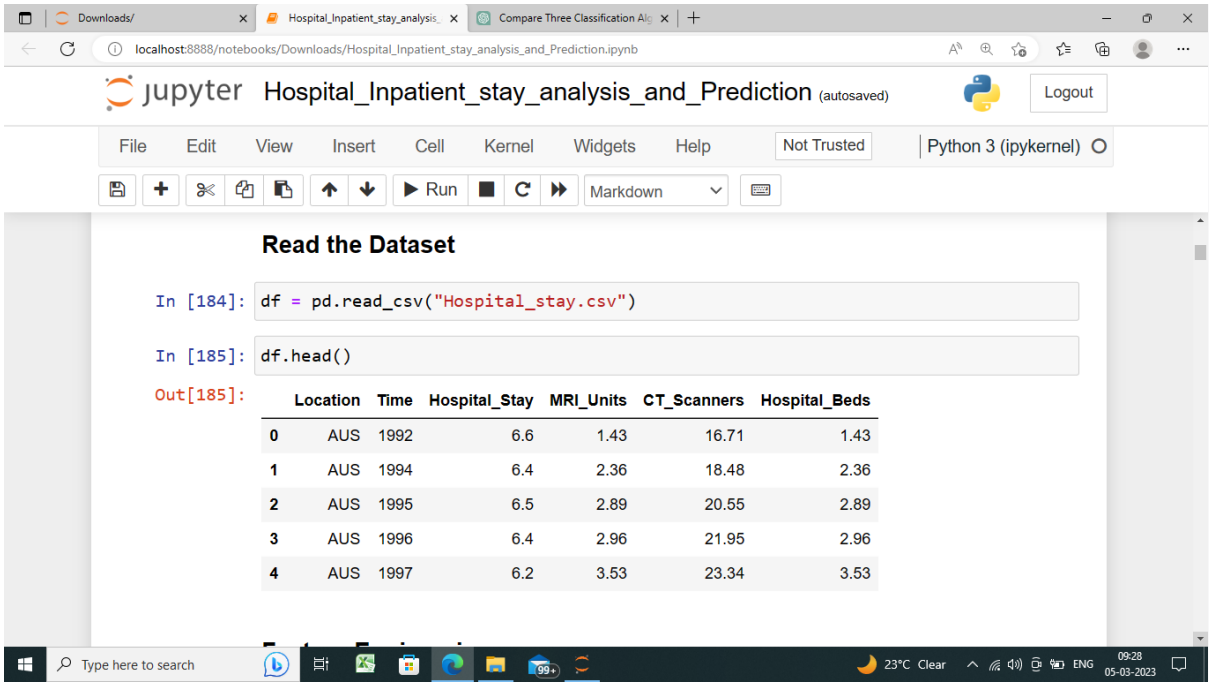


Figure : 6.3

6.2 Data Preprocessing

The dataset will be preprocessed to clean the data and prepare it for analysis. This will include removing any missing or irrelevant data, converting categorical variables into numerical values, and scaling the data to ensure that all variables have a similar range.

6.2.1 Statistical analysis

Statistical analysis plays a significant role in machine learning. It involves using statistical methods to analyze data, identify patterns, and make predictions. Here are some of the common statistical techniques used in machine learning.

Descriptive statistics summarize the characteristics of a dataset, such as mean, median, mode, variance, and standard deviation.

Inferential statistics help to draw conclusions about a population based on a sample. For example, hypothesis testing is a commonly used inferential statistical technique in machine learning.

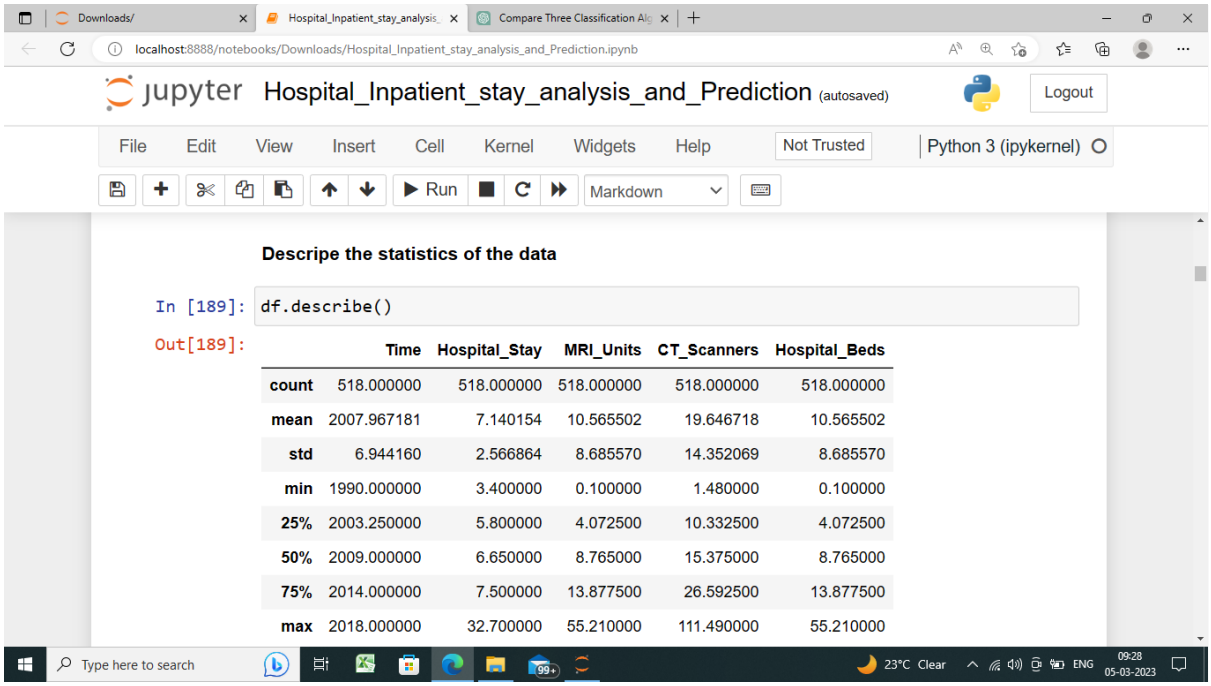


Figure : 6.4

6.2.2 Check the Missing values

Use the isnull() function to check for missing values. The function returns a boolean value indicating whether each value in the dataframe is null (True) or not null (False). To see a summary of missing values across all columns, use the sum() function on the boolean dataframe.

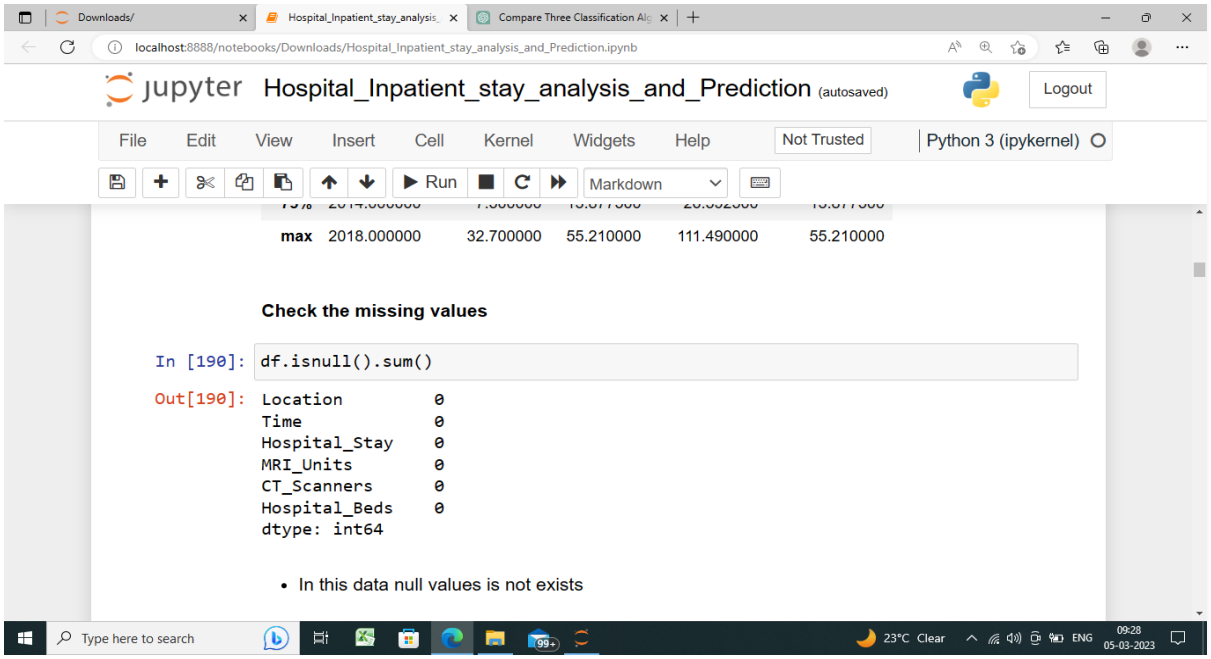


Figure : 6.5

6.3 Exploratary Data Analysis:

Exploratory data analysis (EDA) is a crucial step in the data analysis process. It involves a variety of techniques and methods used to 30nalyse and understand the characteristics of a dataset, including its distribution, structure, and relationships between variables. The goal of EDA is to gain insights into the data and identify patterns, trends, and anomalies that may be useful in subsequent analysis.

This involves creating visual representations of the data, such as histograms, scatter plots, box plots, and heatmaps, to help identify patterns and relationships between variables.

6.3.1 Distribution plot

Distplot is a function from the Python data visualization library Seaborn that allows you to plot a histogram and a kernel density estimate (KDE) of a single variable in a single plot. A histogram is a graphical representation of the distribution of numerical data, where the data is divided into a set of bins, and the count of observations falling within each bin is

displayed. A KDE is a non-parametric way to estimate the probability density function of a random variable.

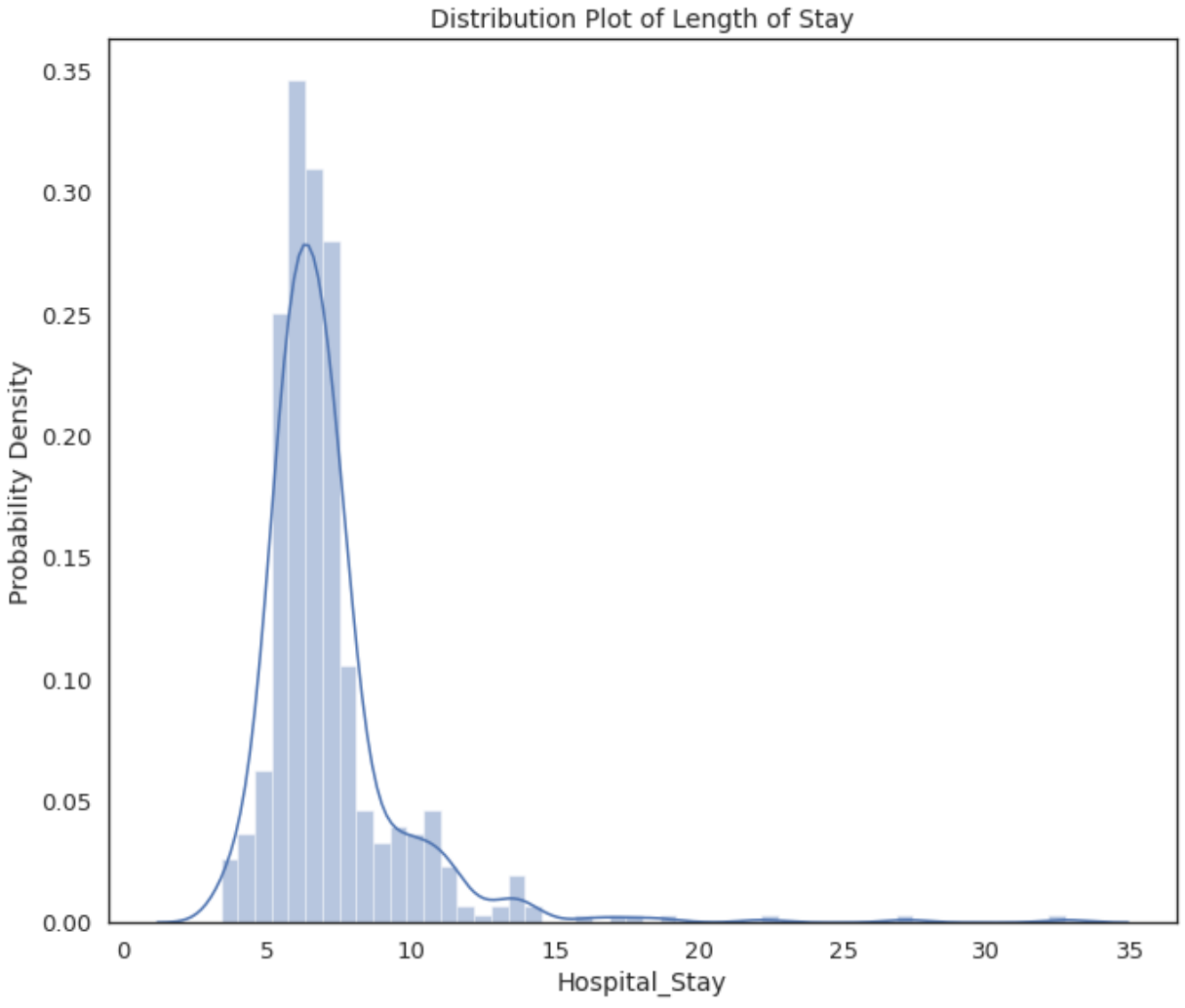


Figure : 6.6

6.3.2 Pair Plot

Pairplot is a function from the Python data visualization library Seaborn that allows you to plot pairwise relationships between variables in a dataset. It creates a matrix of scatter plots and histograms, where each plot shows the relationship between two variables in the dataset. Pairplot is a powerful tool for visualizing complex datasets and identifying patterns and relationships between variables.

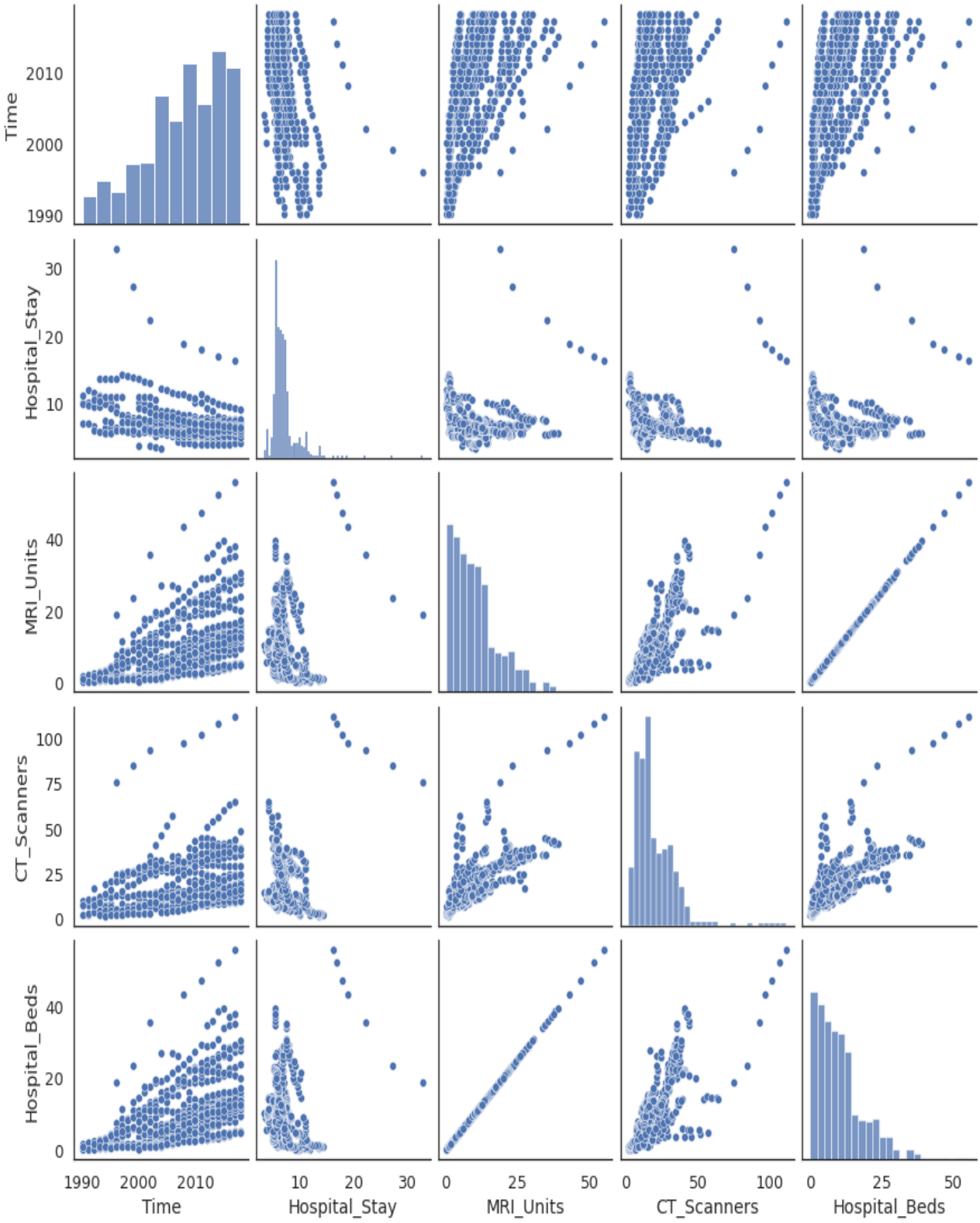


Figure : 6.7

Pairplot is a useful tool for quickly visualizing the relationships between variables in a dataset. By examining the scatter plots, you can identify patterns and trends in the data, such as linear or nonlinear relationships, clusters or groups of data points, or outliers. By

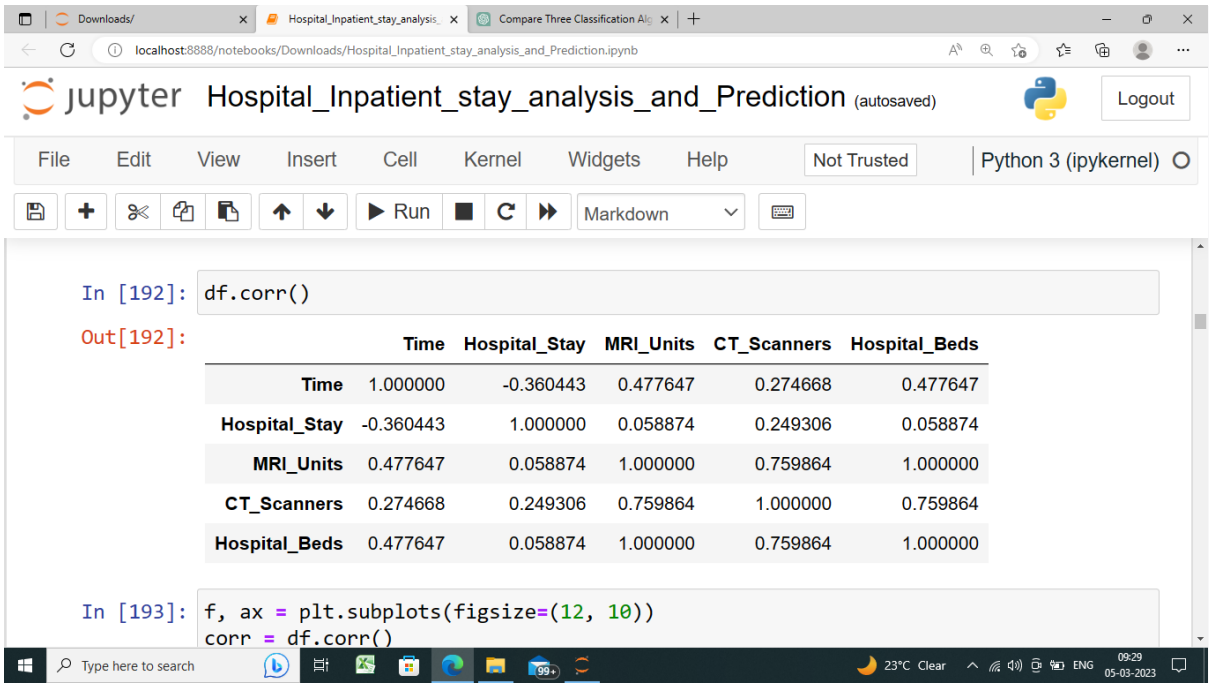
examining the histograms, you can identify the distribution of each variable and detect any skewness or outliers.

Pairplot can also be customized to include additional information, such as different colors or markers for different groups of data points, or different regression models to fit the data. By customizing the pairplot, you can gain deeper insights into the relationships between variables and better understand the patterns and trends in your data.

6.3.3 Features Correlation

To create a correlation matrix in Python, you can use the Pandas library, which provides the corr() function. The corr() function calculates the correlation coefficient between all pairs of variables in a dataset and returns a dataframe that contains the correlation matrix.

A correlation matrix is a table that shows the correlation coefficients between variables in a dataset. It is a useful tool in machine learning for identifying the relationships between variables and for identifying variables that may be redundant or highly correlated with each other.



	Time	Hospital_Stay	MRI_Units	CT_Scanners	Hospital_Beds
Time	1.000000	-0.360443	0.477647	0.274668	0.477647
Hospital_Stay	-0.360443	1.000000	0.058874	0.249306	0.058874
MRI_Units	0.477647	0.058874	1.000000	0.759864	1.000000
CT_Scanners	0.274668	0.249306	0.759864	1.000000	0.759864
Hospital_Beds	0.477647	0.058874	1.000000	0.759864	1.000000

Figure : 6.8

The output will be a square matrix where the rows and columns correspond to the variables in the dataset. Each cell in the matrix represents the correlation coefficient between the variable in the row and the variable in the column. The correlation coefficient ranges from -1 to 1, with -1 indicating a perfect negative correlation, 0 indicating no correlation, and 1 indicating a perfect positive correlation

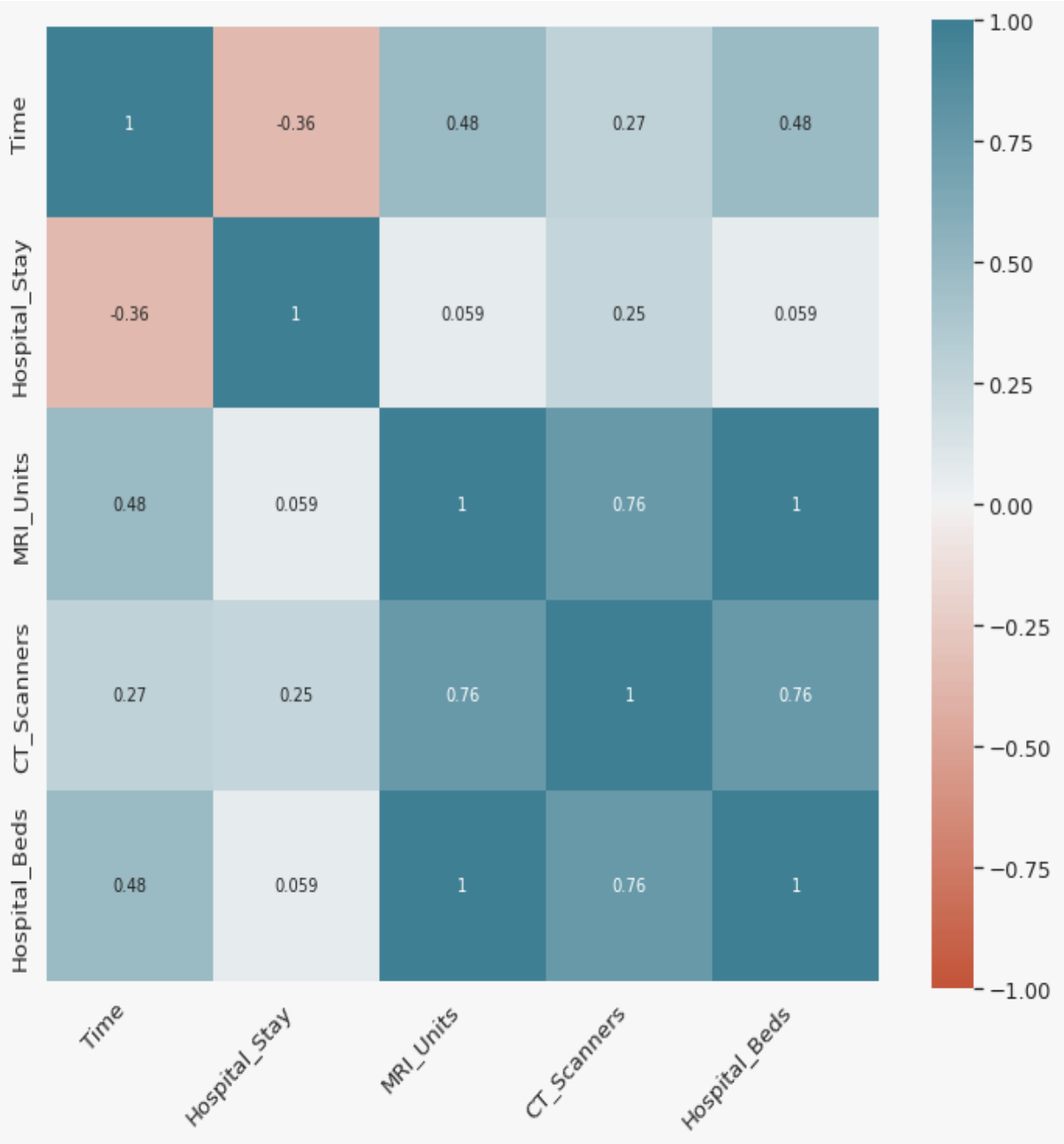


Figure : 6.9

6.3.4 Histogram

Histogram is a type of graph that displays the distribution of a set of continuous numerical data. It consists of a set of rectangles, or bins, where the width of each bin represents a range of data values, and the height of each bin represents the frequency, or count, of data values that fall within that range.

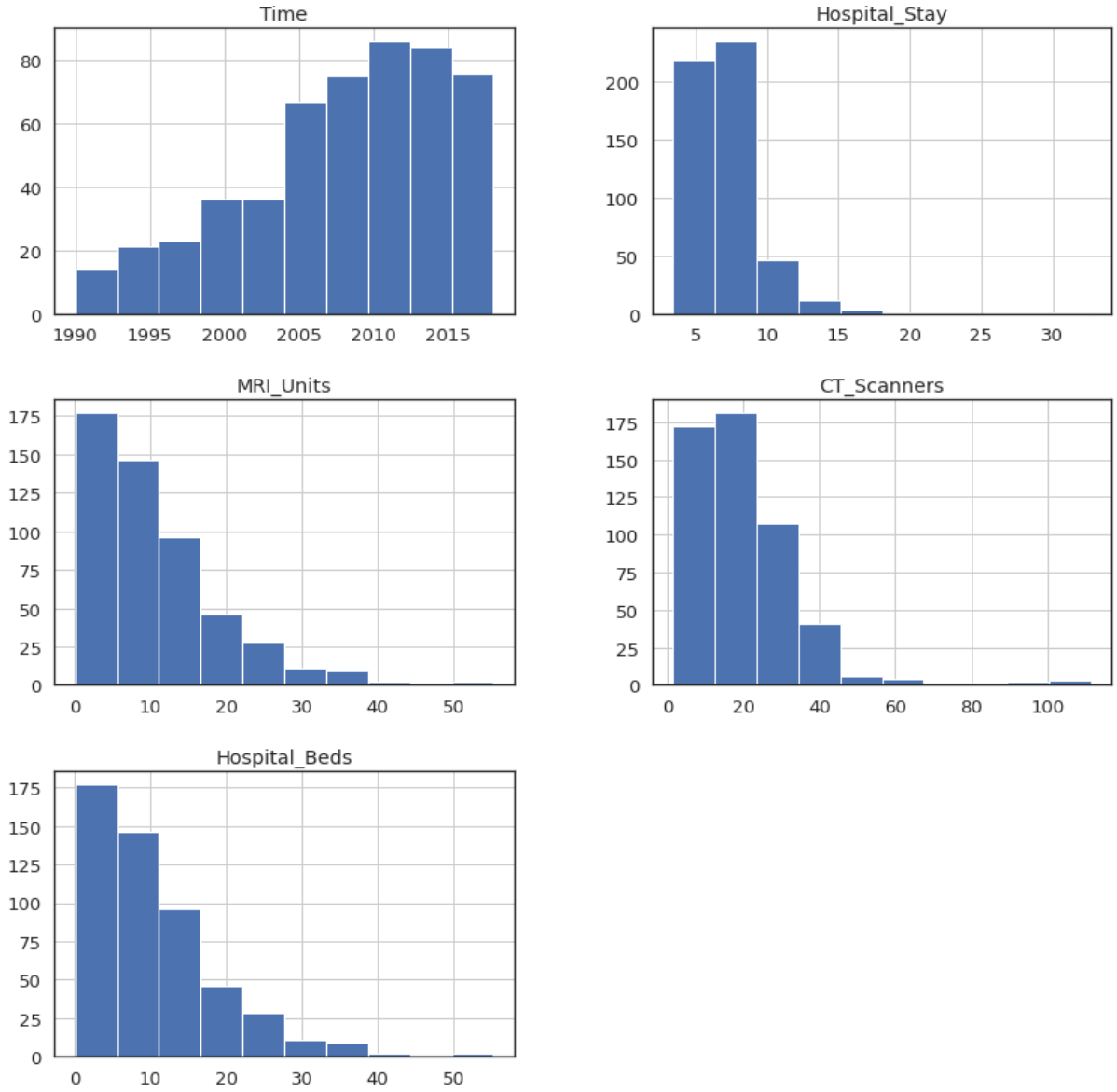


Figure : 6.10

Histograms are a useful tool for visualizing the distribution of a single variable and identifying patterns and anomalies in the data. By examining the shape of the histogram, you can identify whether the data is normally distributed, skewed to the left or right, or has multiple peaks or modes. You can also use histograms to compare the distribution of two or more datasets, by plotting them on the same histogram with different colors or markers

Histograms can be customized in a variety of ways to better suit your data and visualization needs. You can change the number of bins, the bin width or range, the colors and markers used to display the data, and the axis labels and title. By customizing the histogram, you can gain deeper insights into the distribution of your data and better understand its patterns and trends.

6.3.5 Box Plot

A boxplot is a type of chart that displays the distribution of a set of continuous numerical data and is used to identify outliers and assess the variability and skewness of the data.A boxplot is made up of several parts:

The box, which represents the middle 50% of the data, or the interquartile range (IQR). The bottom of the box is the first quartile (Q1), which represents the 25th percentile of the data, and the top of the box is the third quartile (Q3), which represents the 75th percentile of the data. The height of the box represents the variability of the data within this range.

The whiskers, which extend from the top and bottom of the box to the minimum and maximum values within 1.5 times the IQR. Any data points outside this range are considered outliers and are plotted as individual points.

The median, which is the line inside the box that represents the middle value of the data.

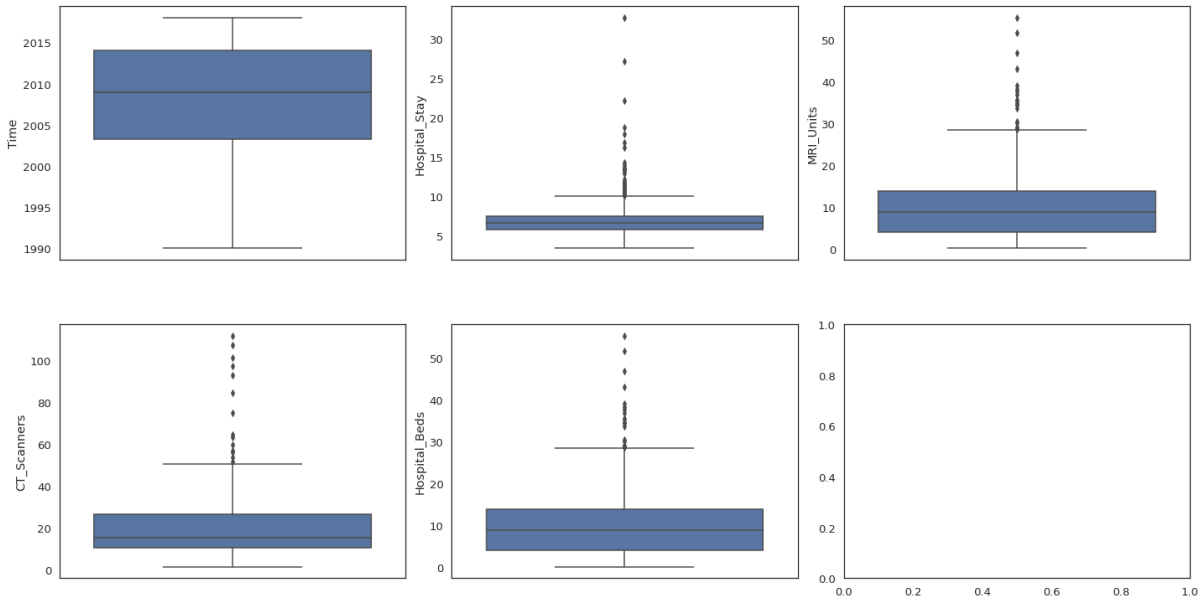


Figure : 6.11

6.4 Data Cleaning

6.4.1 Drop the column

To drop a column in a Pandas dataframe, you can use the drop() function. The drop() function takes the name of the column to be dropped as its argument.

In this project we have a column as “Location” in object type. Therefore we gone drop the column and use the dummies on the value of location.

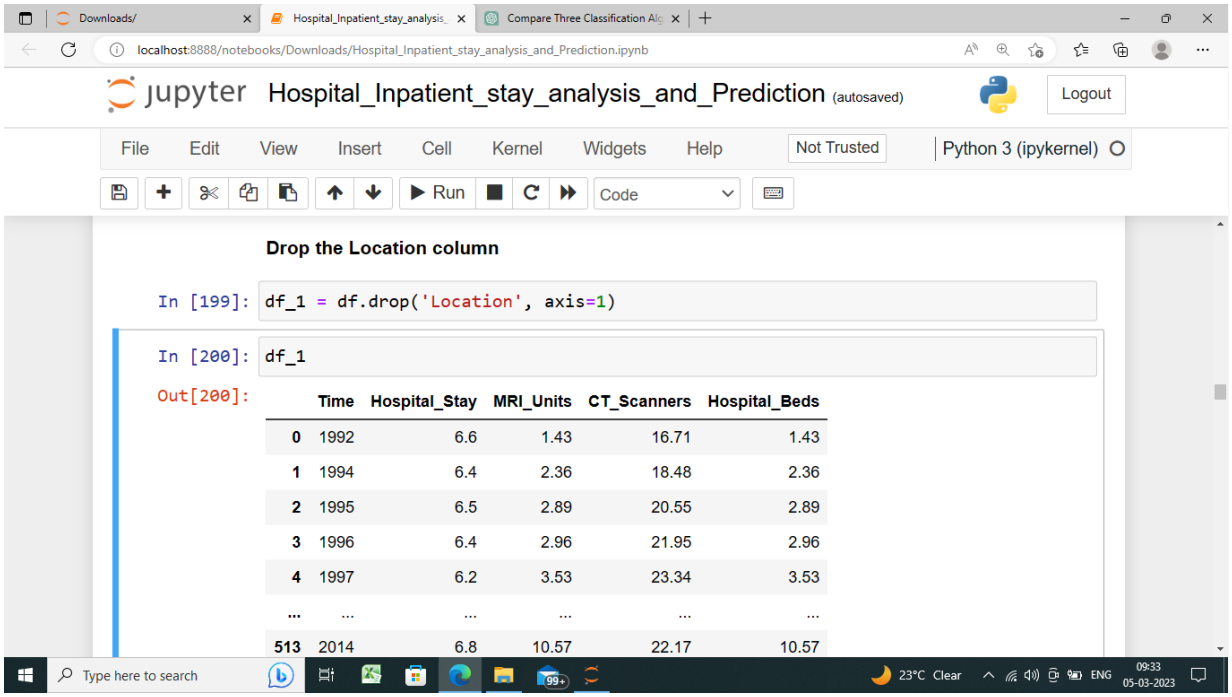


Figure : 6.12

6.4.2 Feature Encoding

Feature encoding is the process of converting categorical data into numerical data that can be used in machine learning algorithms. There are several methods for feature encoding, including one-hot encoding, label encoding, and binary encoding.

One popular method for feature encoding is one-hot encoding. One-hot encoding creates a new binary column for each category in a categorical feature. Each column represents a single category, and the value in the column is 1 if the category is present for that row, and 0 otherwise. We have a one categorical feature called 'Location' with more possible values: AUS, USA, CAN, DEU, EST, ESP, FIN, TUR, SVK and etc... One-hot encoding would

create three new columns called ‘AUS’, 'USA', and 'SVK'. For each row, the value in the appropriate column would be 1 if that Location was present, and 0 otherwise.

Label encoding is another method for feature encoding that assigns a numerical value to each category in a categorical feature. Each category is assigned a unique integer value, such that the values are ordered and have some meaning. For example, in the 'Location' feature, red might be assigned a value of 1, AUS might be assigned a value of 2, and USA might be assigned a value of 3.

Binary encoding is a method for feature encoding that uses binary digits to represent each category. Each category is assigned a unique binary code, such that the codes are ordered and have some meaning. For example, in the 'Location' feature, AUS might be represented as 001, USA be represented as 010, and SVK be represented as 100.

Feature encoding is an important step in machine learning because many machine learning algorithms require numerical data as input. By converting categorical data into numerical data, we can use these algorithms to make predictions and analyze the data.

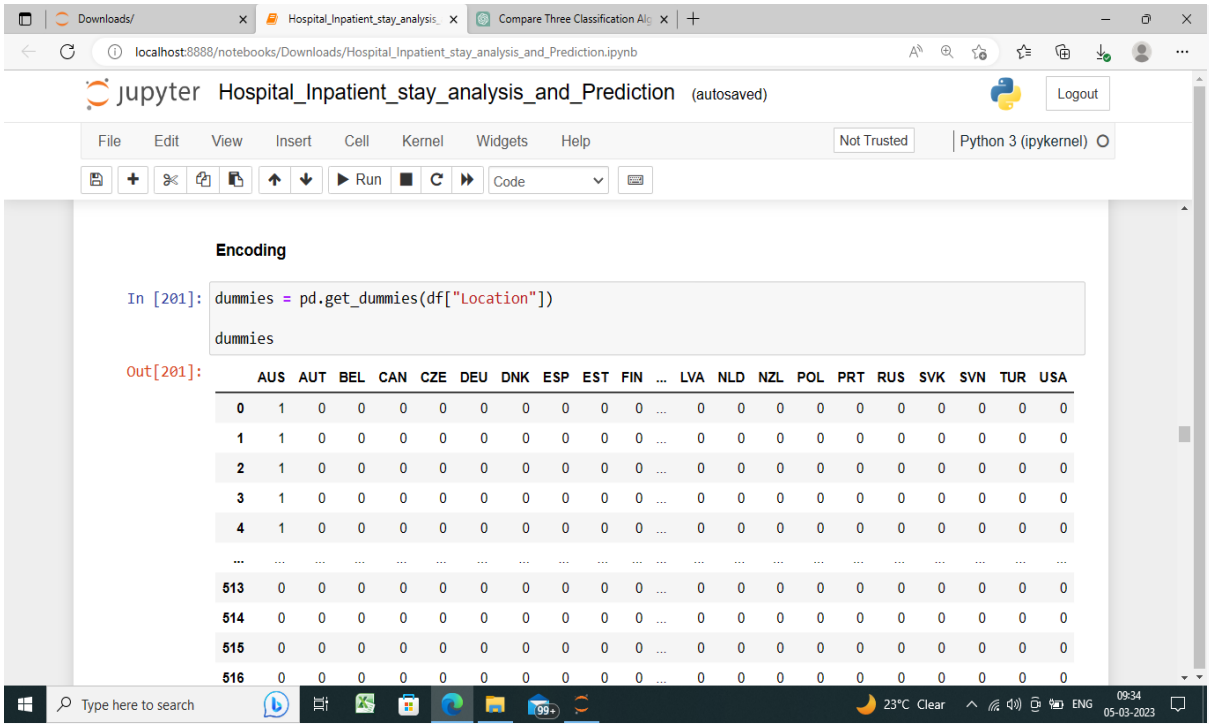


Figure : 6.13

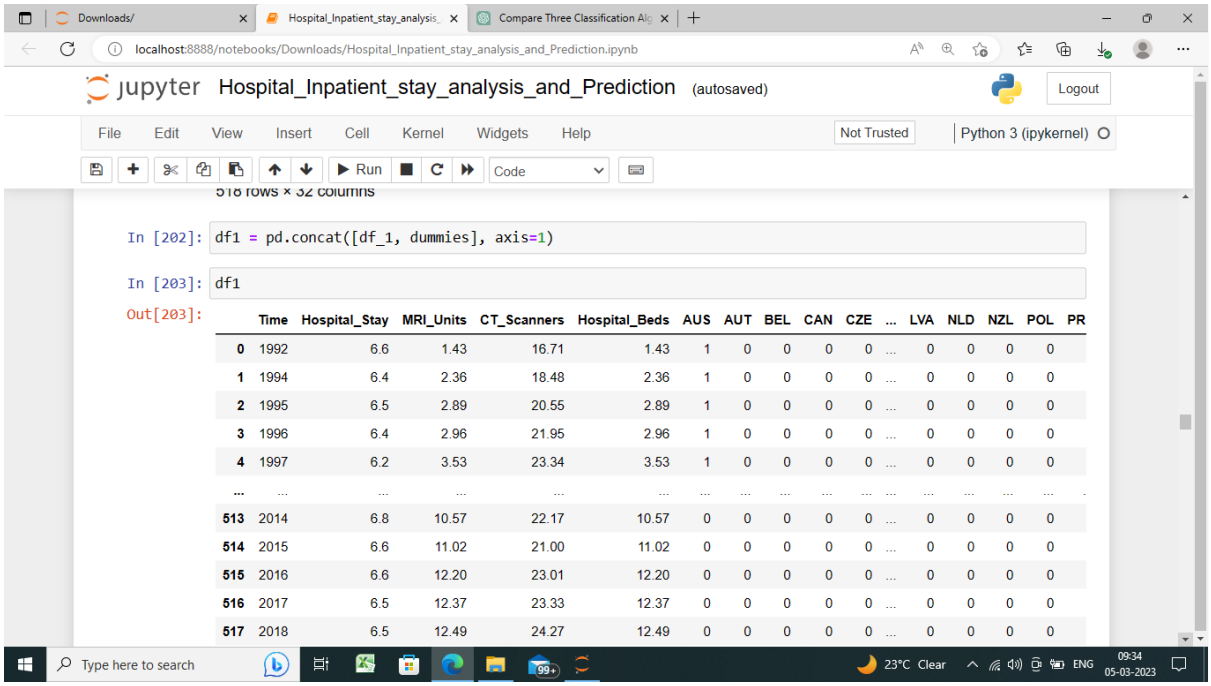


Figure : 6.14

6.5 Split the Data

In machine learning, the process of splitting data refers to dividing a dataset into different subsets for training, validation, and testing. This division is essential for developing and evaluating machine learning models. Typically, the dataset is split into two or three subsets, depending on the specific requirements of the task at hand.

Let's discuss the common types of data splits used in machine learning:

6.5.1 Training Set

The training set is the largest portion of the dataset, used to train the machine learning model. It contains labeled examples that the model will use to learn patterns and make predictions. The training set should be representative of the overall data distribution to ensure that the model learns generalizable patterns.

6.5.2 Validation Set

The validation set is a subset of the dataset used to fine-tune the model's hyperparameters and assess its performance during training. It helps in making decisions

regarding model architecture, feature selection, and regularization techniques. The validation set consists of labeled examples, and its performance metrics guide the model's optimization process.

6.5.3 Test Set

The test set is an independent subset of the dataset used to evaluate the final performance of the trained model. It serves as an unbiased measure of the model's accuracy and generalization capabilities. The test set remains untouched during the training and validation phases to provide an objective assessment of the model's performance on unseen data.

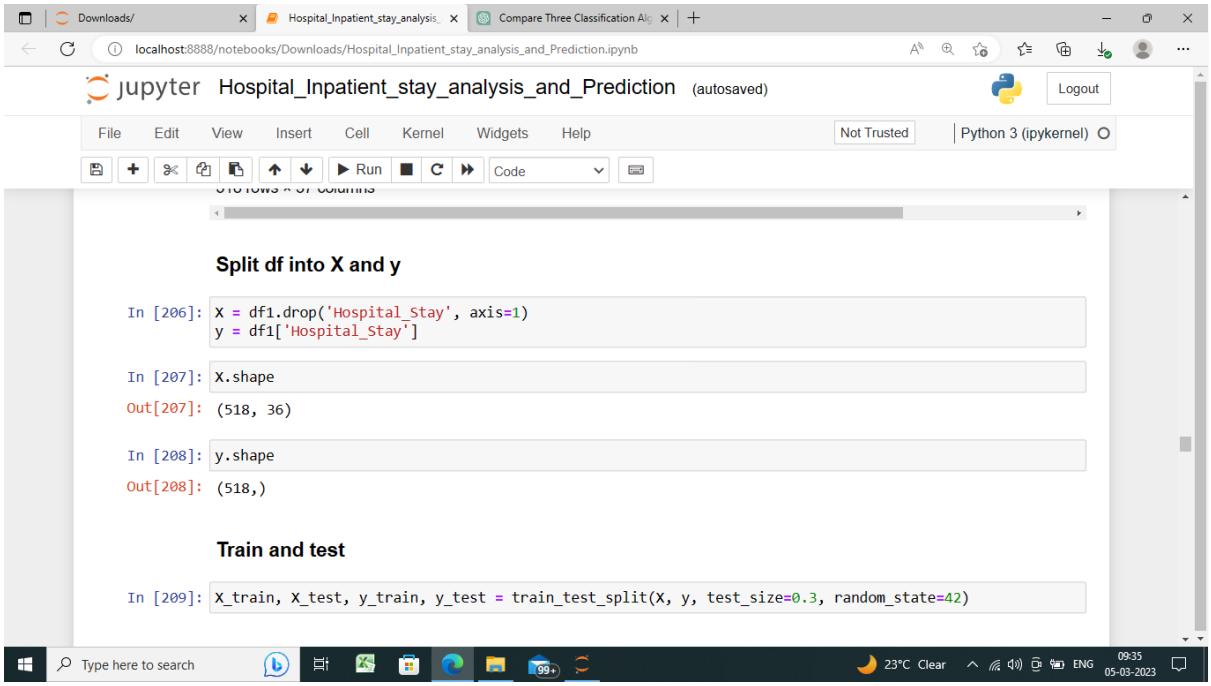


Figure : 6.15

It's important to ensure that the data split is done randomly and maintains the original distribution of the dataset. Random splitting helps avoid any bias that might be present in the data ordering. The common practice is to use a fixed proportion of the data for each split, such as an 80-20 split for training and testing or a 60-20-20 split for training, validation, and testing, respectively.

Additionally, it's crucial to consider the specific requirements of the problem and the characteristics of the dataset when deciding on the data split. For example, in cases where the dataset is imbalanced (i.e., some classes have significantly fewer examples than others), techniques like stratified sampling can be employed to ensure that each subset retains the same class distribution as the original dataset.

Overall, proper data splitting allows for the development of robust and reliable machine learning models by providing an objective evaluation of their performance on unseen data

6.6 Feature Engineering:

Feature engineering is an important step in developing a machine learning model. In this project, we will use various techniques such as feature selection and feature scaling to identify the most important features and transform them into a format suitable for the machine learning algorithm.

6.6.1 Scaling

Standardization is a common technique used in machine learning to rescale numerical features so that they have a mean of zero and a standard deviation of one. Standardization is useful because it allows us to compare features that are on different scales, and it can improve the performance of some machine learning algorithms.

Here's how standardization works in practice. Let's say we have a feature vector x with n numerical features :

$$x = [x1, x2, \ldots, xn]$$

To standardize the feature vector x, we first calculate the mean (mu) and standard deviation (sigma) of each feature:

$$mu = [mean(x1), mean(x2), \ldots, mean(xn)]$$

$$sigma = [std(x1), std(x2), \ldots, std(xn)]$$

We then subtract the mean from each feature and divide by the standard deviation:

$$x\_std = [(x1 - mu[1]) / sigma[1], (x2 - mu[2]) / sigma[2], \ldots, (xn - mu[n]) / sigma[n]]$$

The resulting feature vector x_std has a mean of zero and a standard deviation of one.

Standardization can be expressed mathematically as:

$$x\_std = (x - mu) / sigma$$

where x is the original feature vector, mu is the mean vector, and sigma is the standard deviation vector.

Standardization can be implemented using various machine learning libraries in Python, such as scikit-learn. Here's an example of how to standardize a dataset using scikit-learn:

The StandardScaler object is first initialized, then fitted to the dataset using the fit() method. The fit() method calculates the mean and standard deviation of each feature in the dataset. The transform() method is then used to transform the dataset into a standardized form. The resulting standardized dataset is stored in the df_std variable.

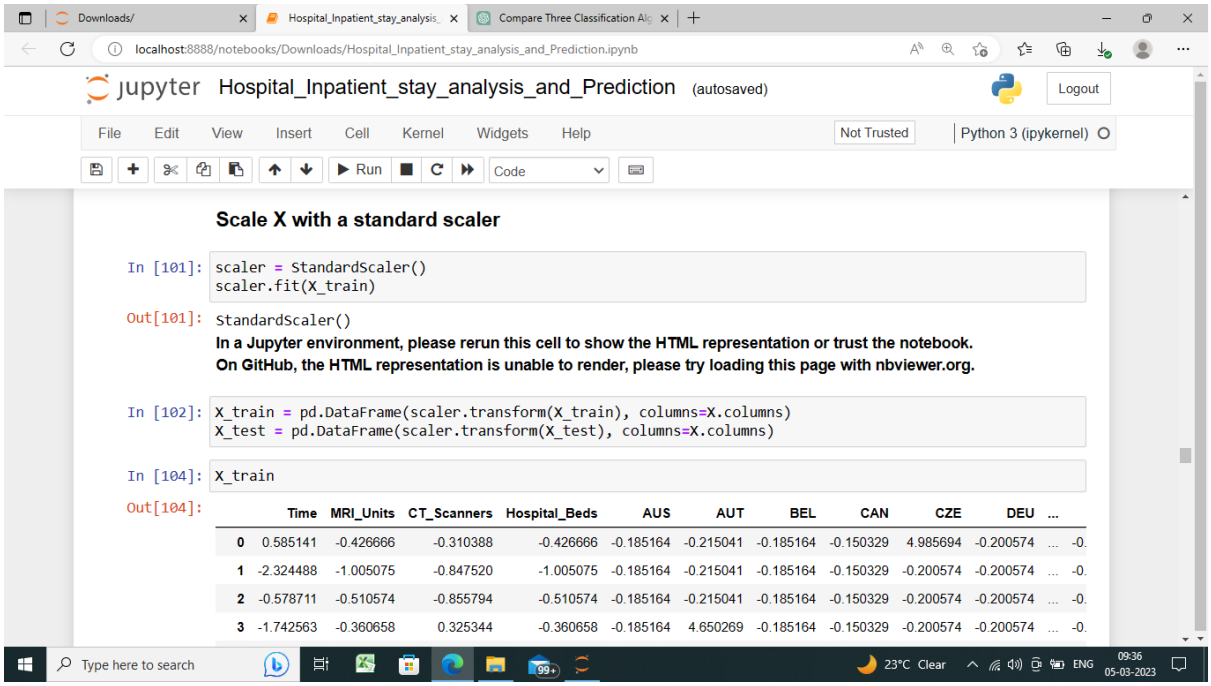


Figure : 6.16

6.6.2 Principal Component Analysis

Principal Component Analysis (PCA) is a statistical technique that is commonly used to analyze and reduce the dimensionality of a large dataset. The goal of PCA is to find a new

set of orthogonal variables, called principal components, that explain the maximum amount of variance in the original dataset.

PCA is a mathematical technique that involves linear algebra and matrix operations. Here's a step-by-step breakdown of the math behind PCA:

Standardize the data: Let X be an n x p matrix of n observations of p variables. The first step in PCA is to standardize the data by subtracting the mean from each variable and dividing by its standard deviation, resulting in a new matrix X', where each variable has a mean of zero and a standard deviation of one.

Compute the covariance matrix: The covariance matrix S is calculated as follows:

$$S = (1/n)\ X'\ X$$

where (1/n) is a normalization factor.

Calculate the eigenvectors and eigenvalues: The eigenvectors and eigenvalues of the covariance matrix S are calculated using the following equation:

$$S\ v = \lambda\ v$$

where v is the eigenvector, λ is the corresponding eigenvalue, and S v is the product of the covariance matrix S and the eigenvector v.

Select the principal components: The principal components are selected based on the magnitude of their corresponding eigenvalues. The top k principal components are chosen such that they explain a certain percentage of the total variance in the data, typically 80-90%.

Transform the data: Finally, the original data is transformed into the new principal component space using the following equation:

$$X' = X\ V$$

where X' is the transformed data matrix, X is the standardized data matrix, and V is the matrix of the top k eigenvectors.

PCA is a powerful technique that can help to identify patterns and relationships in complex datasets. By reducing the dimensionality of the data, PCA can simplify data analysis and

visualization, making it a valuable tool in many applications, including data compression, feature selection, and machine learning.

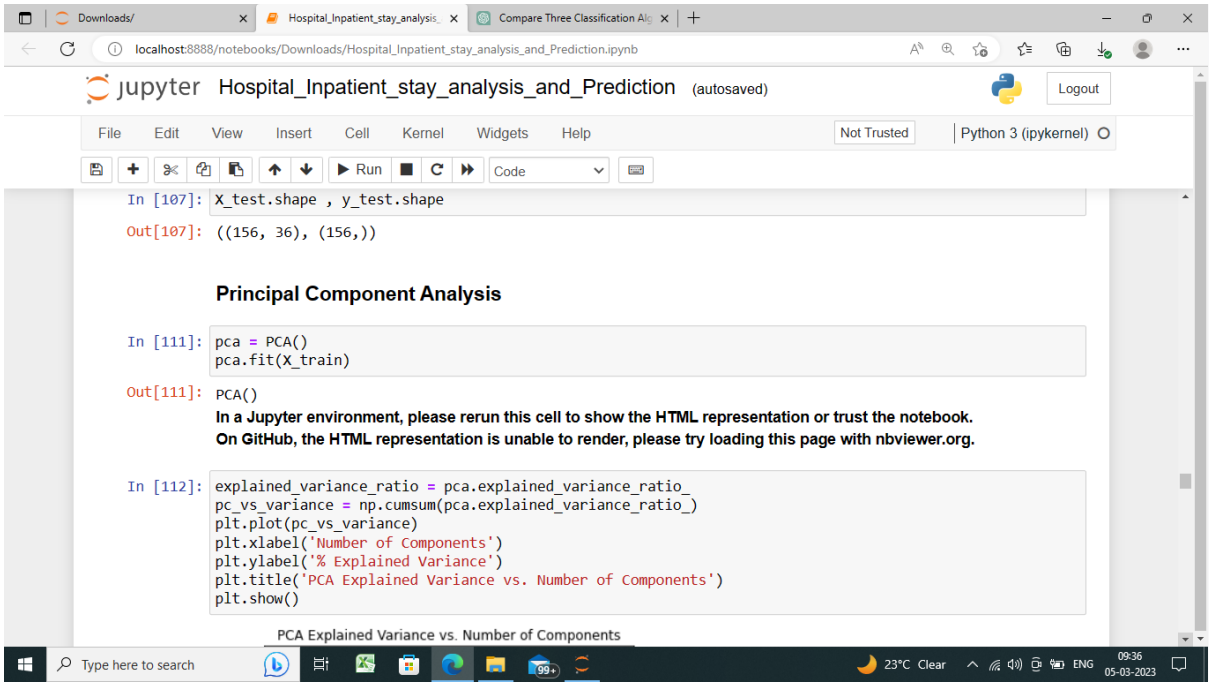


Figure : 6.17

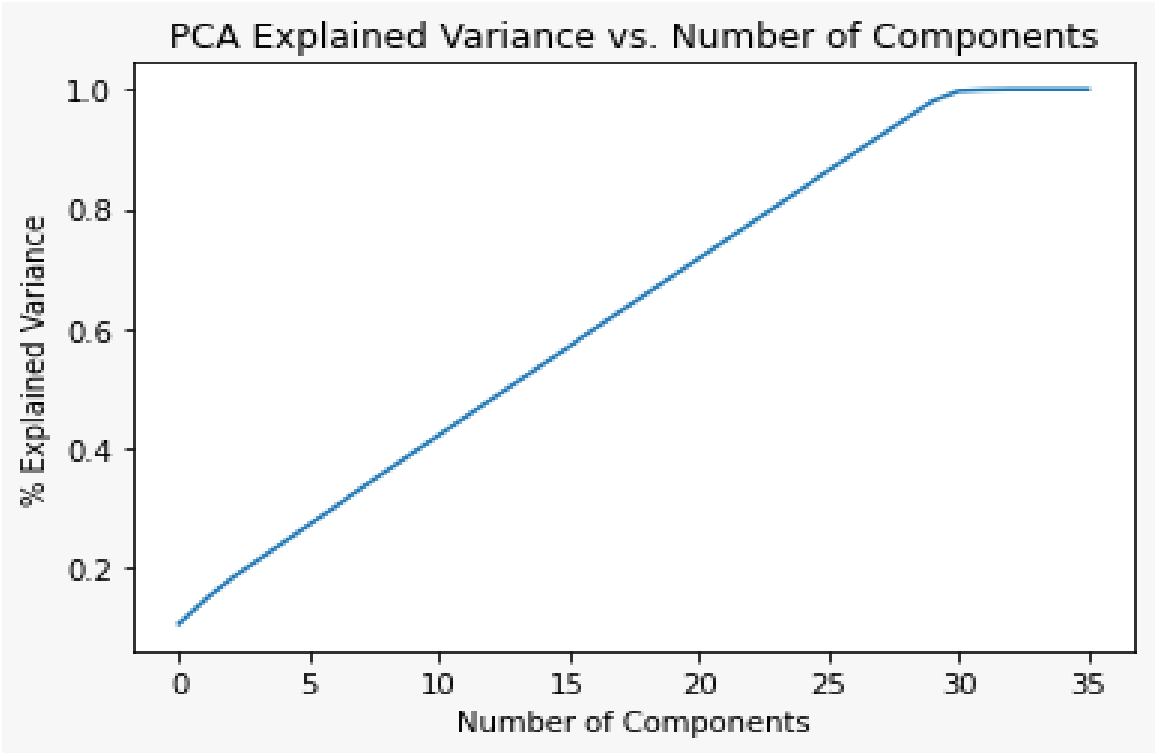


Figure : 6.18

6.7 Model Fit and Evaluation:

The performance of the trained model will be evaluated using cross-validation techniques and other statistical measures. The model will also be tested on a separate dataset to ensure that it can generalize to new data and make accurate predictions.

6.7.1 Linear Regression:

Linear regression is a machine learning algorithm used to predict a numerical target variable based on one or more input variables. It is a supervised learning algorithm that uses a linear relationship between the input variables and the output variable to make predictions.

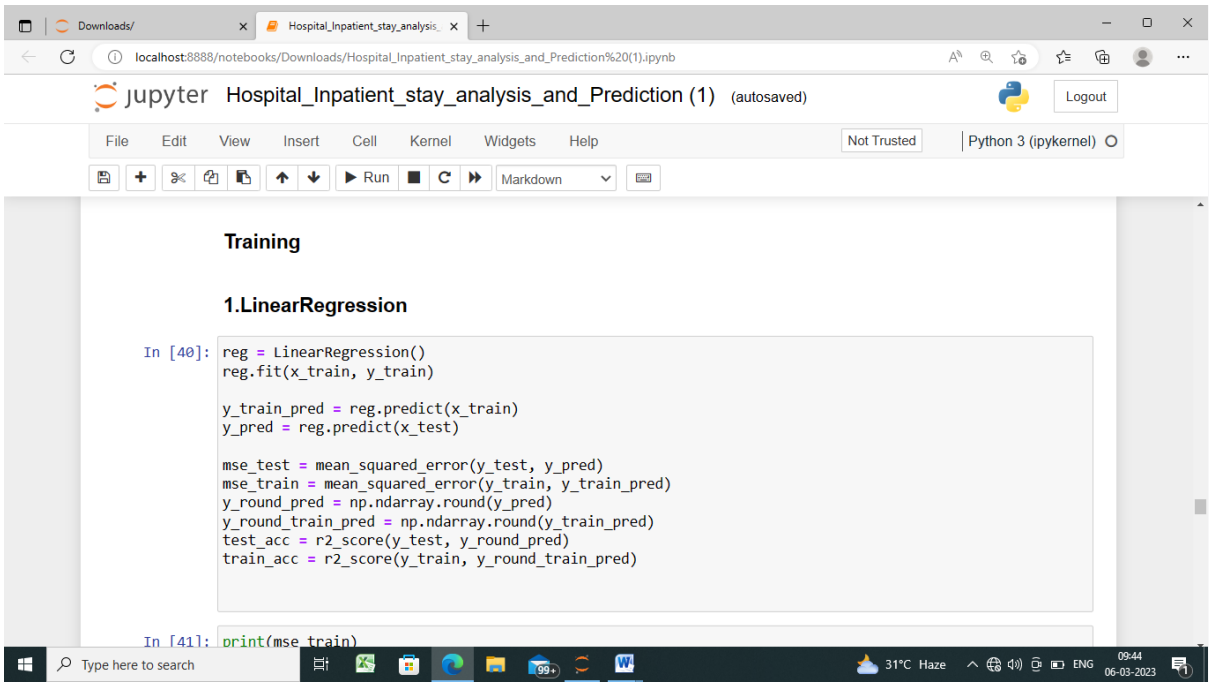


Figure : 6.19

Linear regression algorithm result is :

Train Score = 0.826609654516072

Test Score = 0.79322192441684

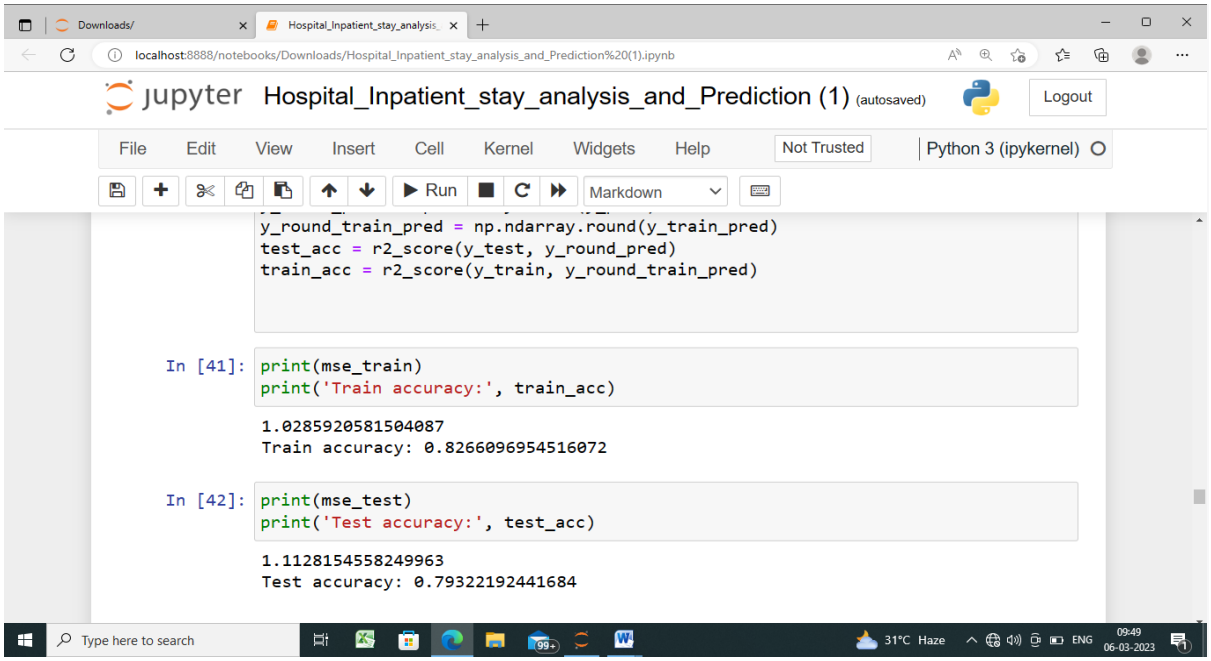
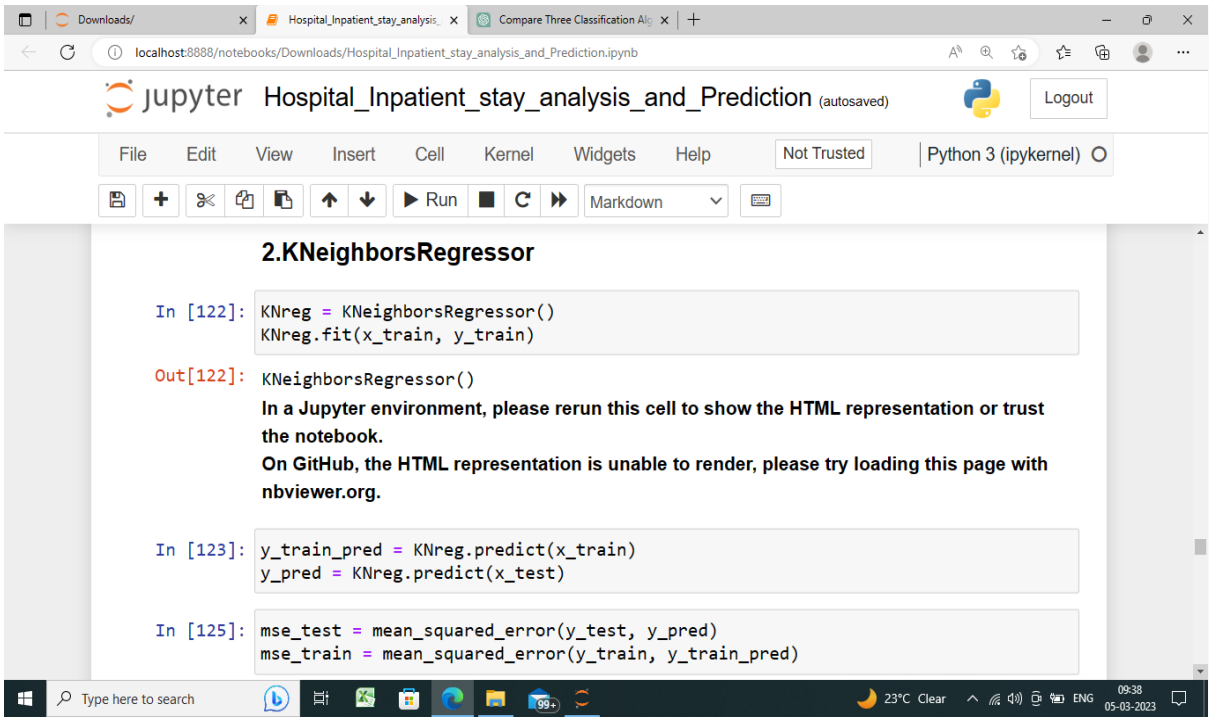


Figure : 6.20

6.7.2 K-Nearest Neighbors

K-Nearest Neighbors (KNN) regression is a machine learning algorithm used to predict a numerical target variable based on one or more input variables. It is a supervised learning algorithm that uses the K-nearest neighbors of a new input point to predict its target value.

Figure : 6.21



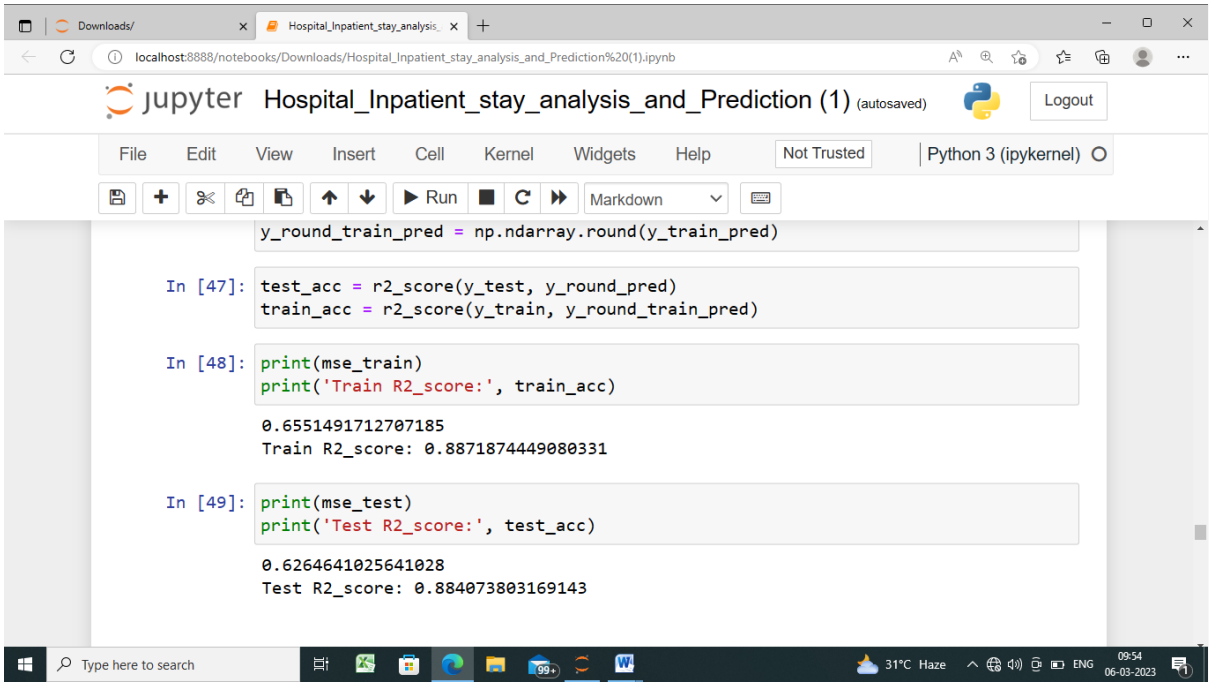


Figure : 6.22

K-NNeighbors algorithm result is :

Training result = 0.8871874449080331

Testing result = 0.884073803169143

6.7.3 Linear SVR

Support Vector Machine (SVM) is a machine learning algorithm used for classification and regression tasks. SVM with a linear kernel is a variant of the SVM algorithm that uses a linear decision boundary to separate the classes in the input space.

Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, where y = {-1, 1}, the goal of SVM with a linear kernel is to find a hyperplane that best separates the two classes in the input space.

Linear SVR algorithm result is :

Training result = 0.7714332542457543

Testing result = 0.7147771304280676

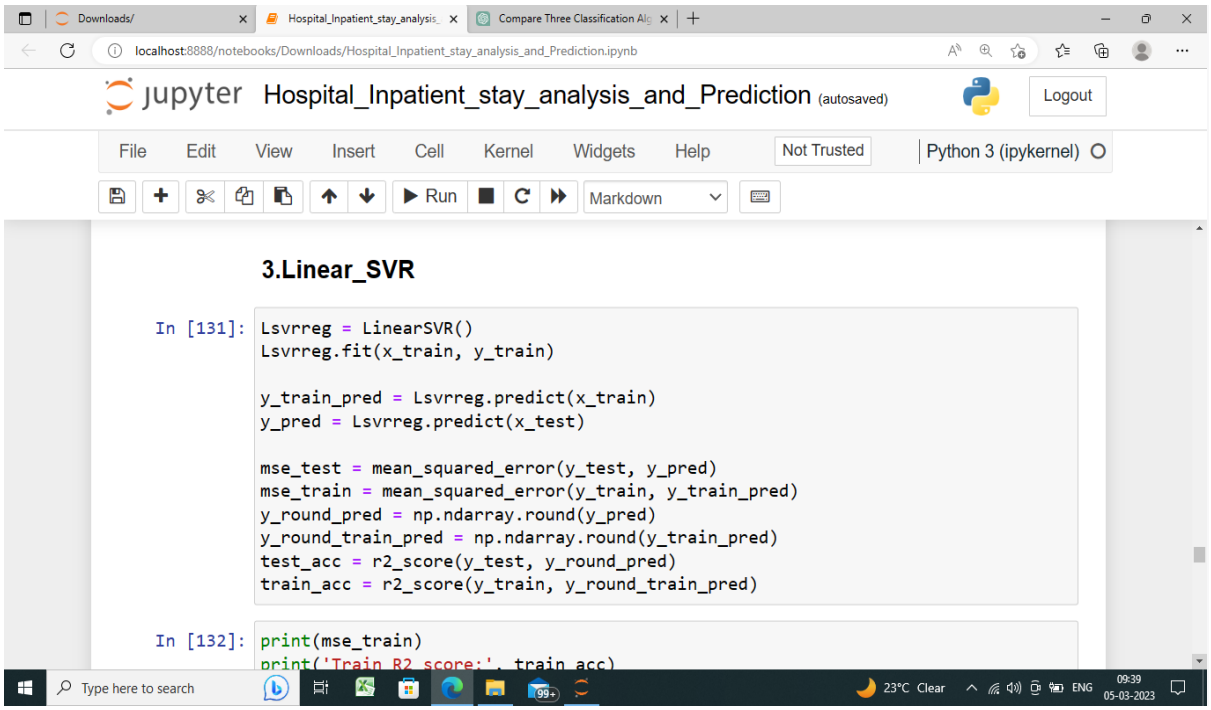


Figure : 6.23

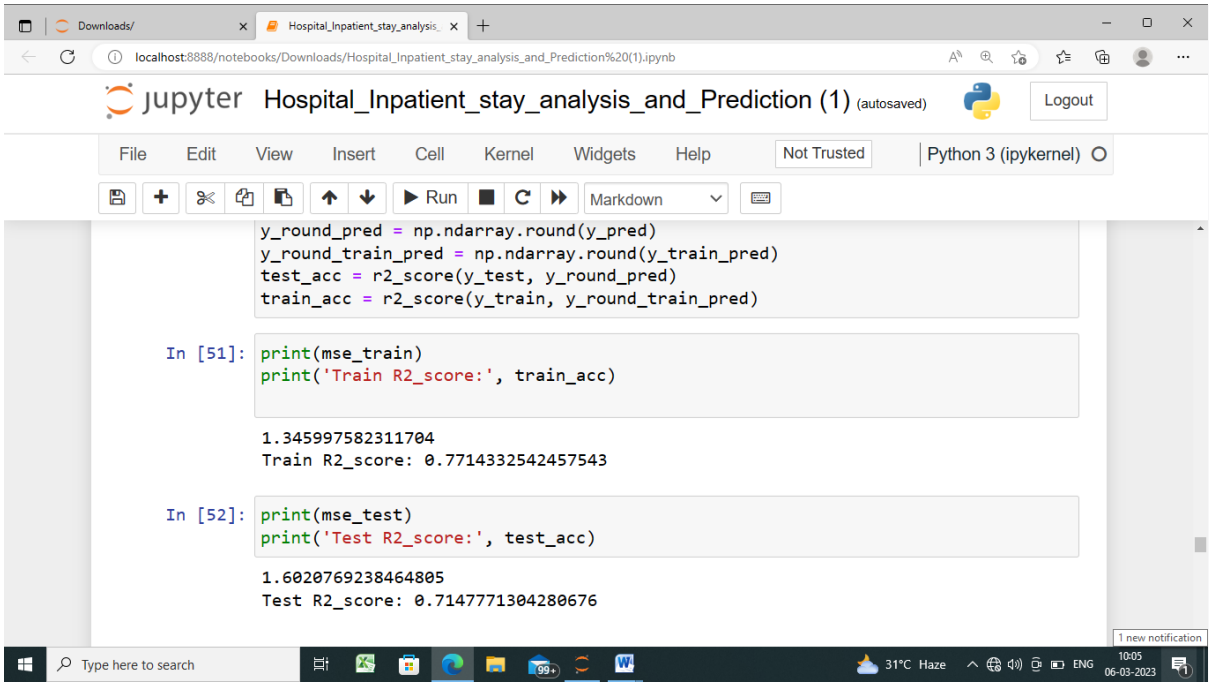


Figure :6.24

6.7.4 Support Vector Regressor

Support Vector Machine (SVM) is a machine learning algorithm used for classification and regression tasks. Support Vector Regression (SVR) is a variant of SVM that is used for regression tasks. The goal of SVR is to find a function that approximates the mapping from input variables to target variables.

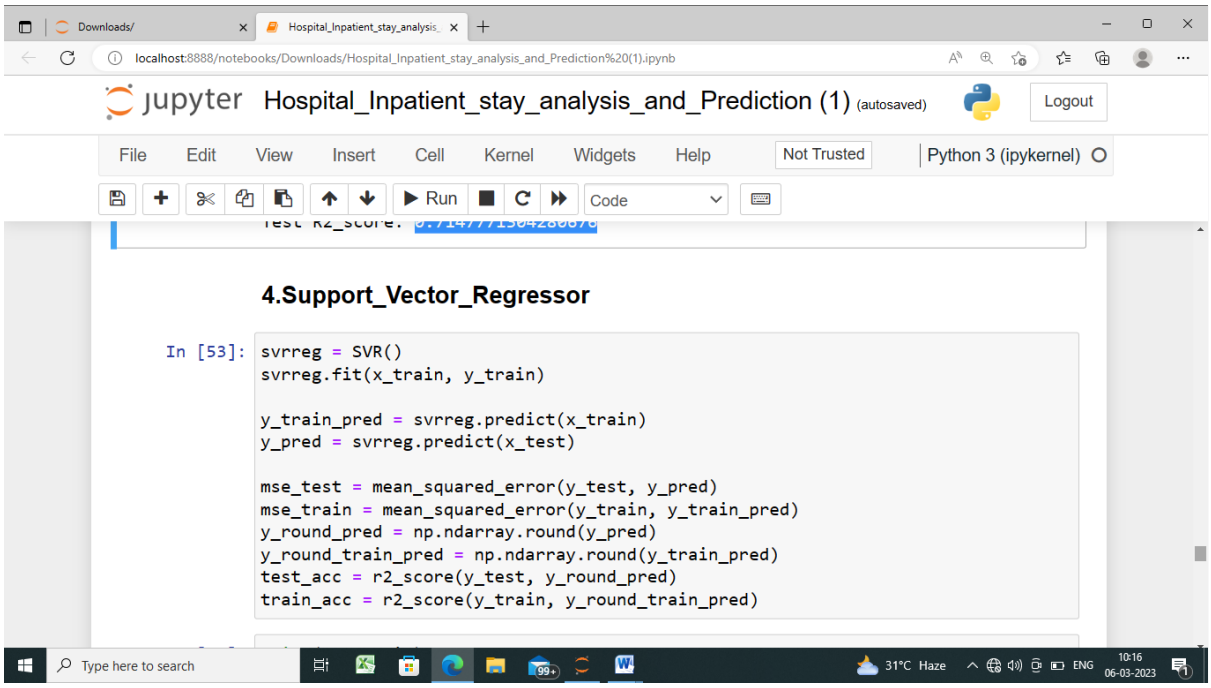
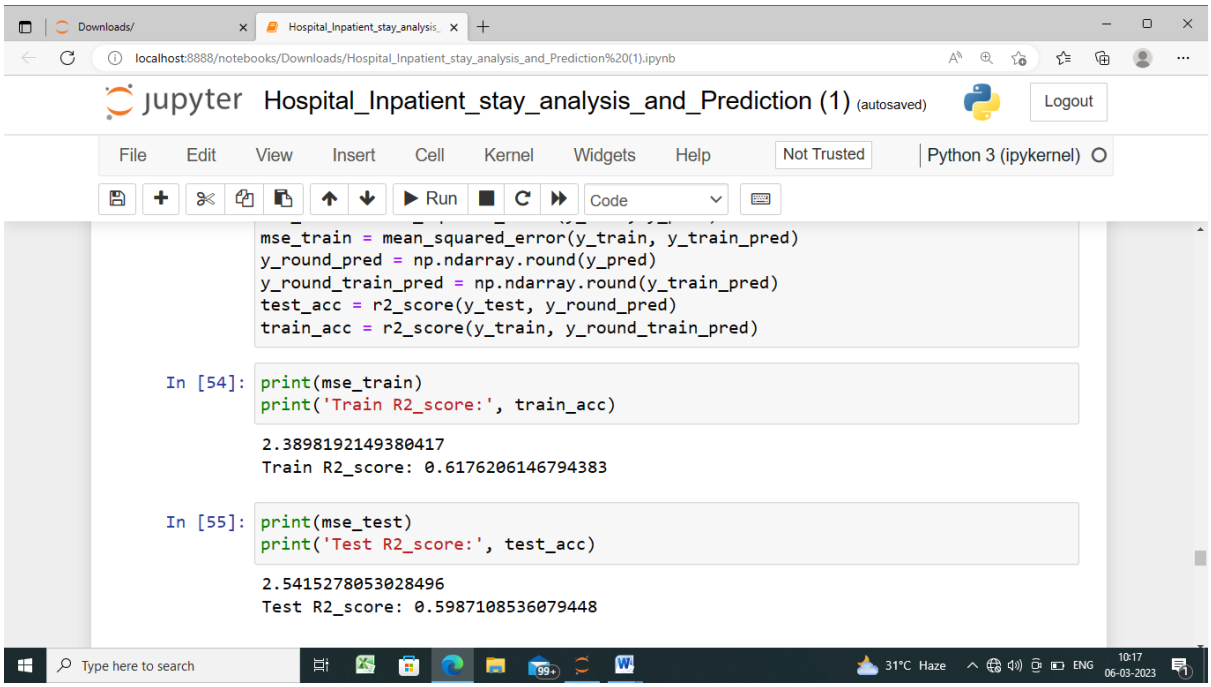


Figure : 6.25



SVR algorithm result is :

Training result = 0.6176206146794383

Testing result = 0.5987108536079448

6.7.5 Decision Tree Regression

Decision tree regression is a non-parametric algorithm used for both classification and regression tasks. In decision tree regression, the goal is to learn a decision tree that can predict the target value for a new input point based on its features. The decision tree is constructed by recursively splitting the data into smaller subsets, based on the values of the features, until a stopping criterion is reached.

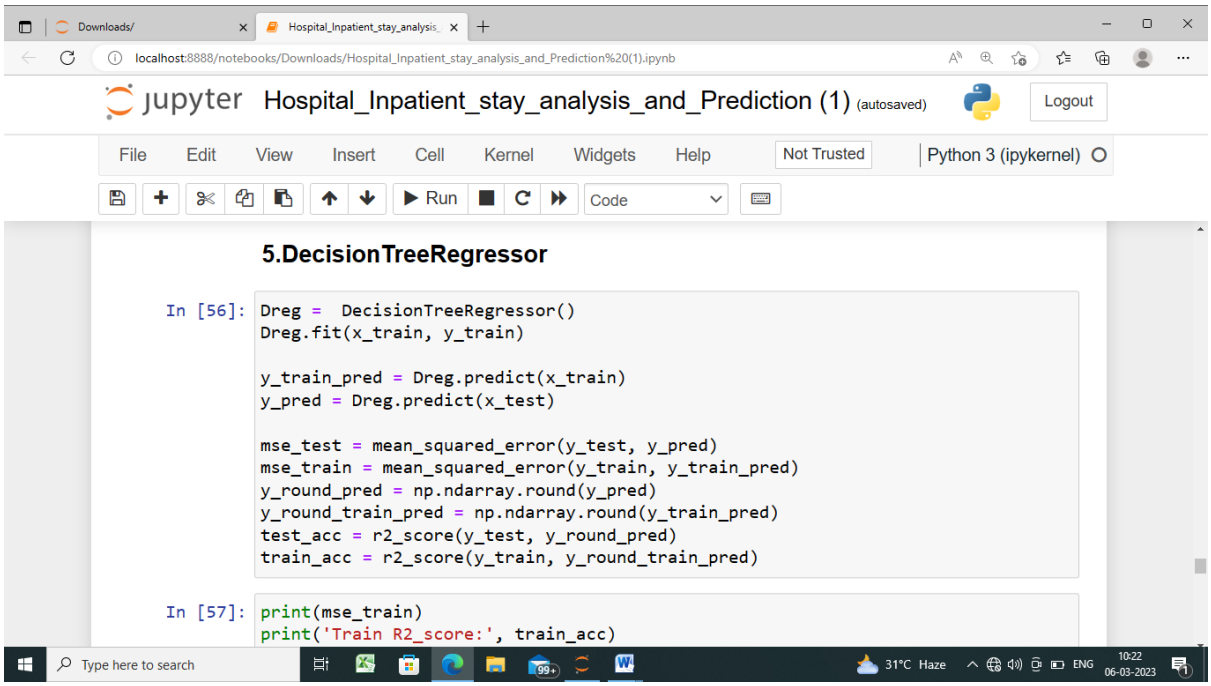


Figure :6.26

Decision Tree Regresion algorithm result is :

Training result = 0.9880672636187342

Testing result = 0.9090880869665832

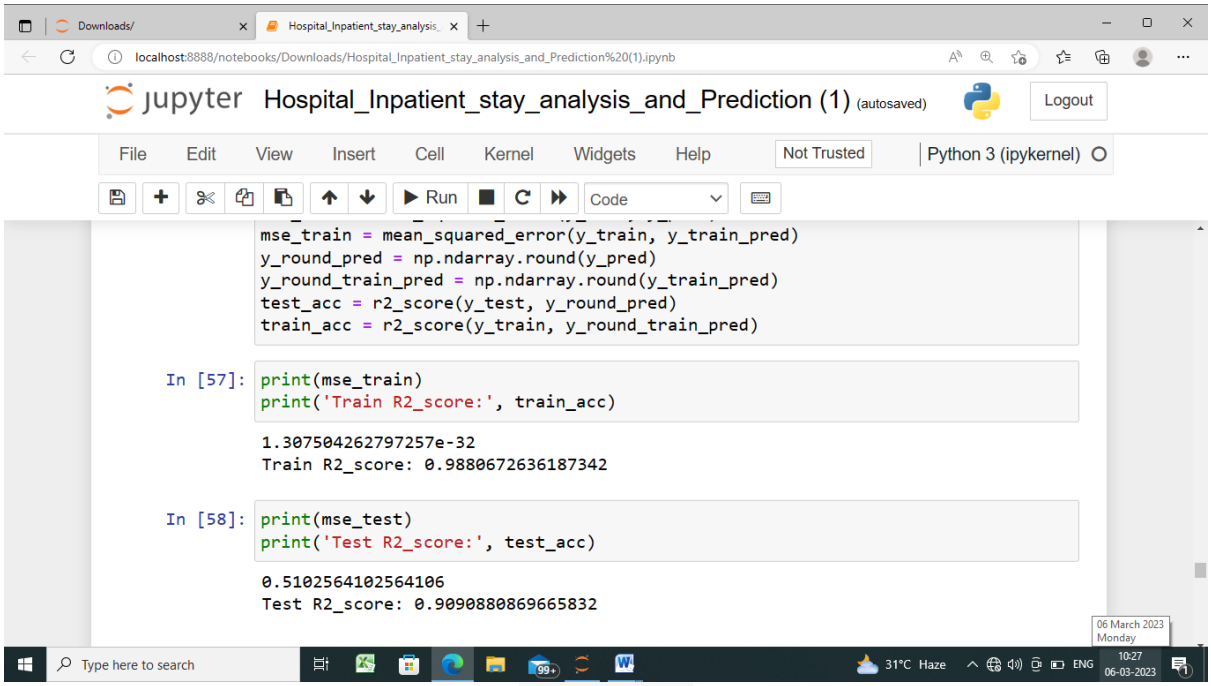


Figure : 6.27

6.7.6 Random Forest Regression

Random Forest Regression is an ensemble learning method that combines multiple decision trees to make predictions. Each decision tree in the random forest is trained on a randomly selected subset of the data and a random subset of the features, which helps to reduce overfitting and increase generalization performance.

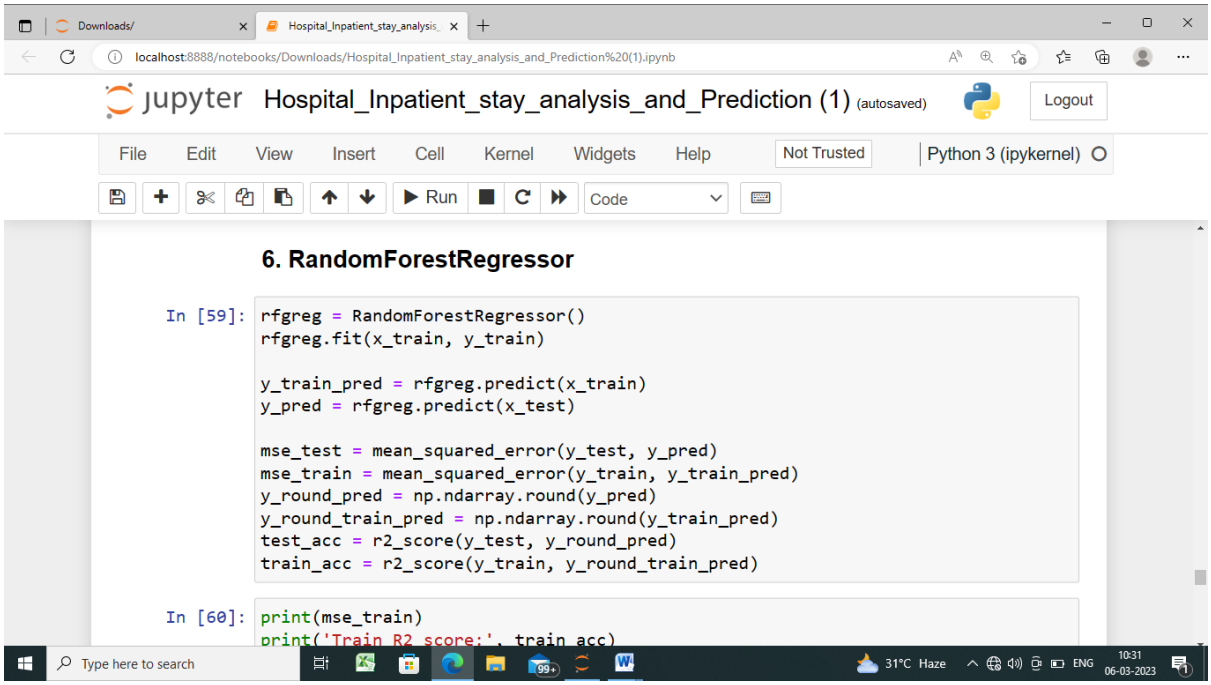


Figure : 6.28

Random Forest Regresion algorithm result is :

Training result = 0.9732759887171651

Testing result = 0.9421069415792044

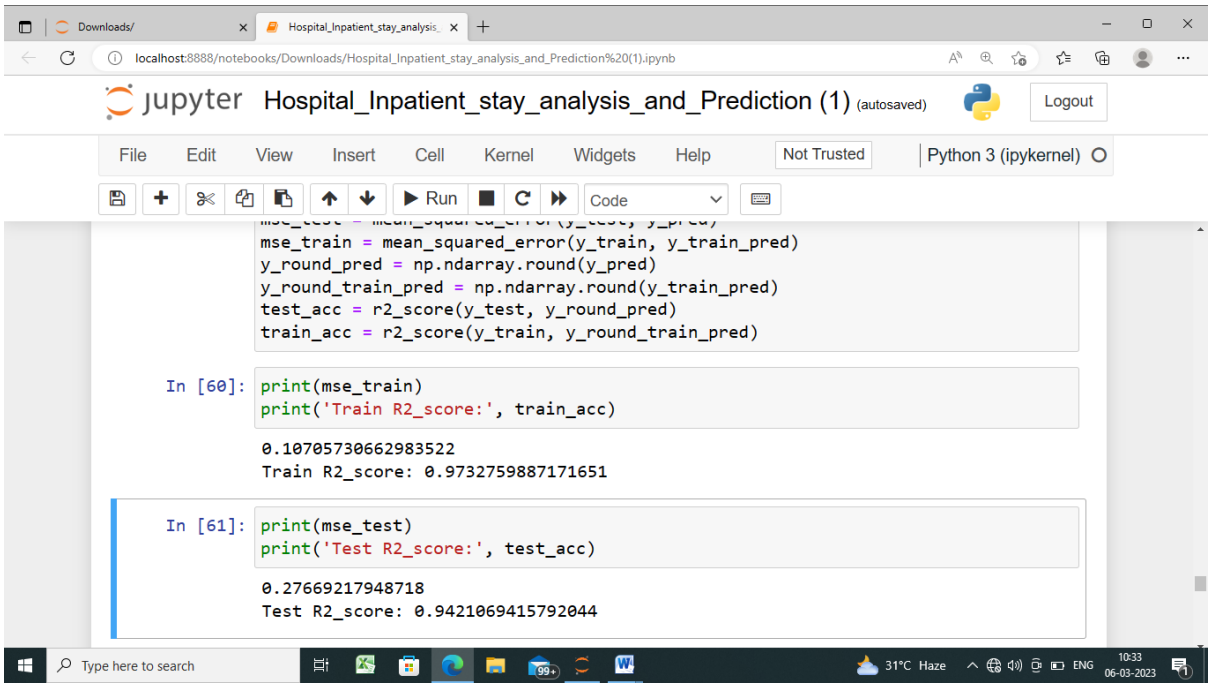


Figure : 6.29

6.7.7 Gradient Boosting Regression

Gradient Boosting Regression is another popular machine learning algorithm for regression tasks, which also works by combining multiple weak models to form a strong model. The algorithm trains a series of decision trees, where each new tree is trained to correct the residual errors of the previous tree.

Gradient Boosting Regresion algorithm result is :

Training result = 0.9833307317682318

Testing result = 0.9549142548834937

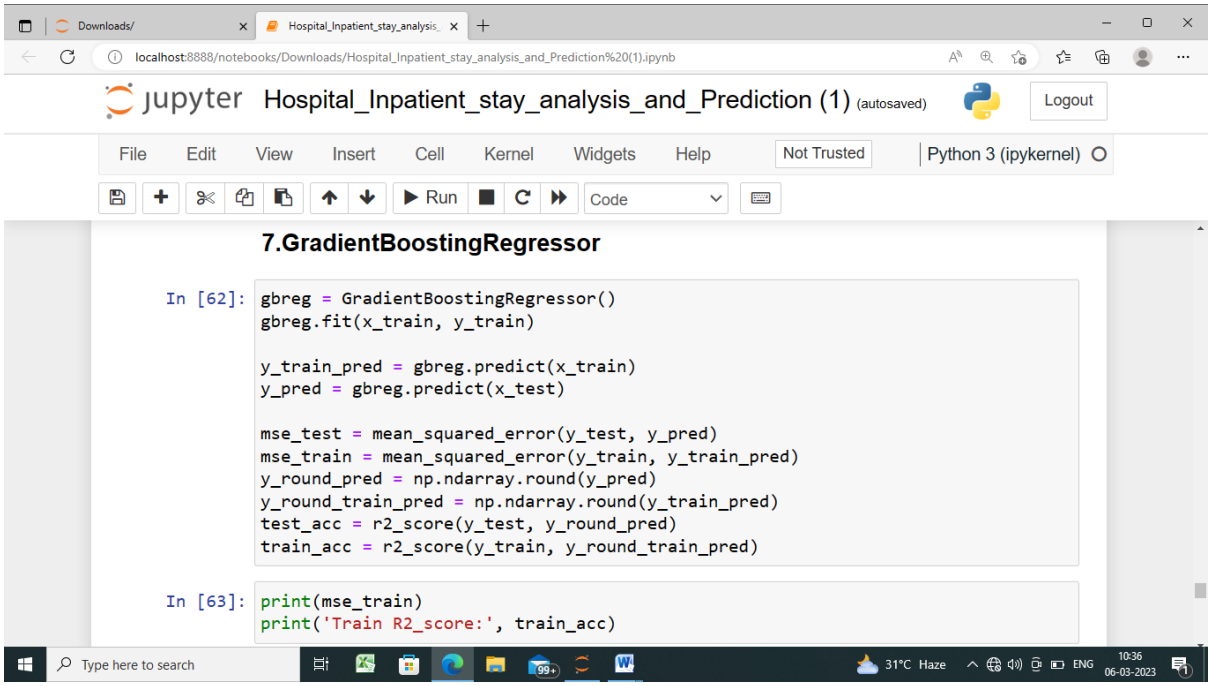


Figure : 6.30

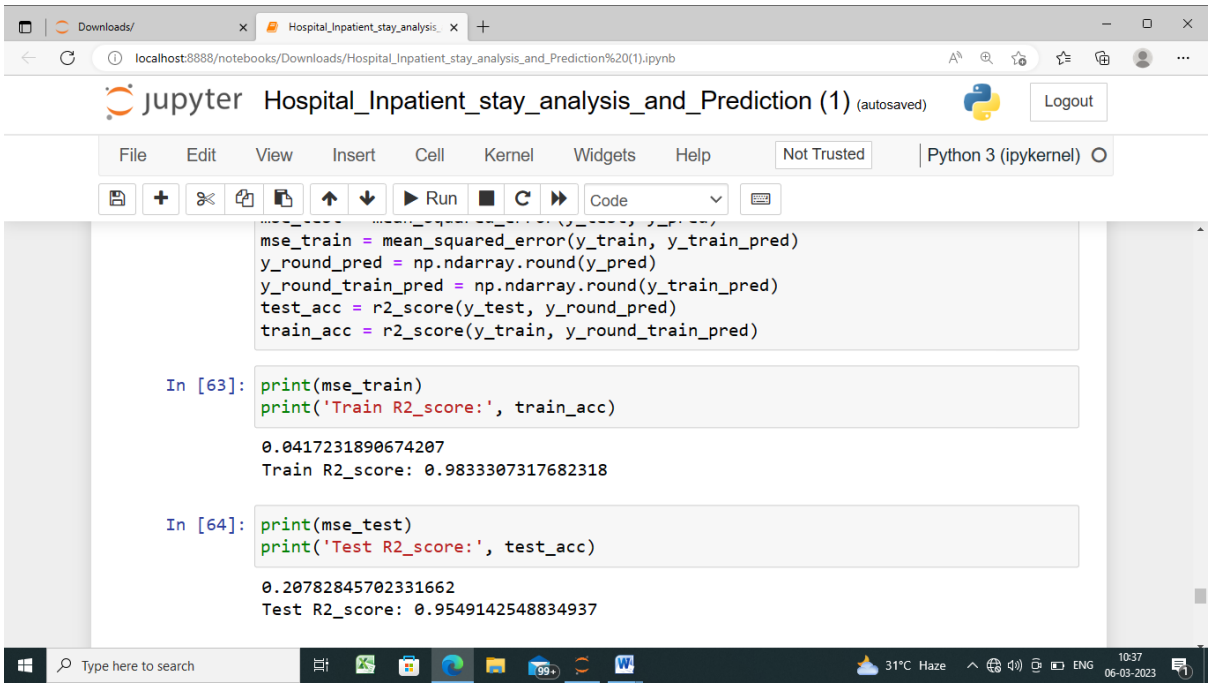


Figure : 6.31

6.7.8 LightGBM Regression

LightGBM is a popular gradient boosting algorithm that is designed to be efficient and scalable for large datasets. It is similar to Gradient Boosting Regression, but uses a different approach to constructing decision trees that allows it to train faster and use less memory.

LightGBM Regresion algorithm result is :

Training result = 0.9071307369101487

Testing result = 0.8752687752724441

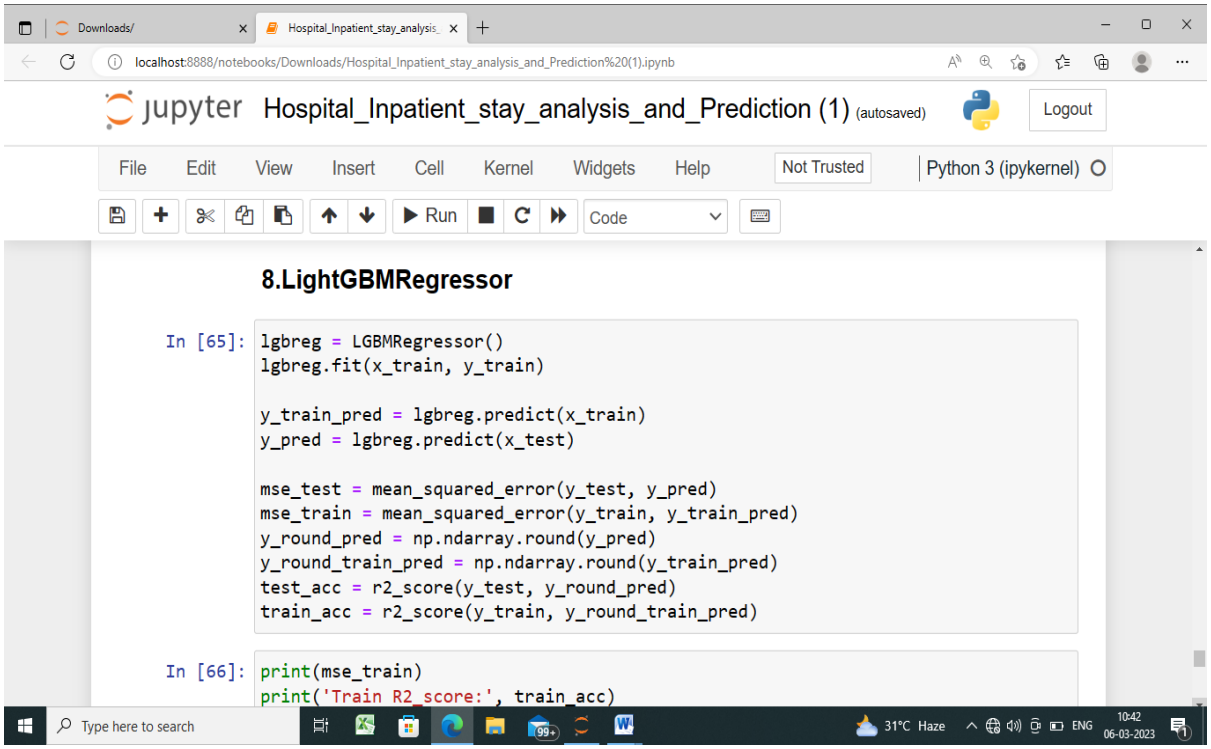


Figure : 6.32

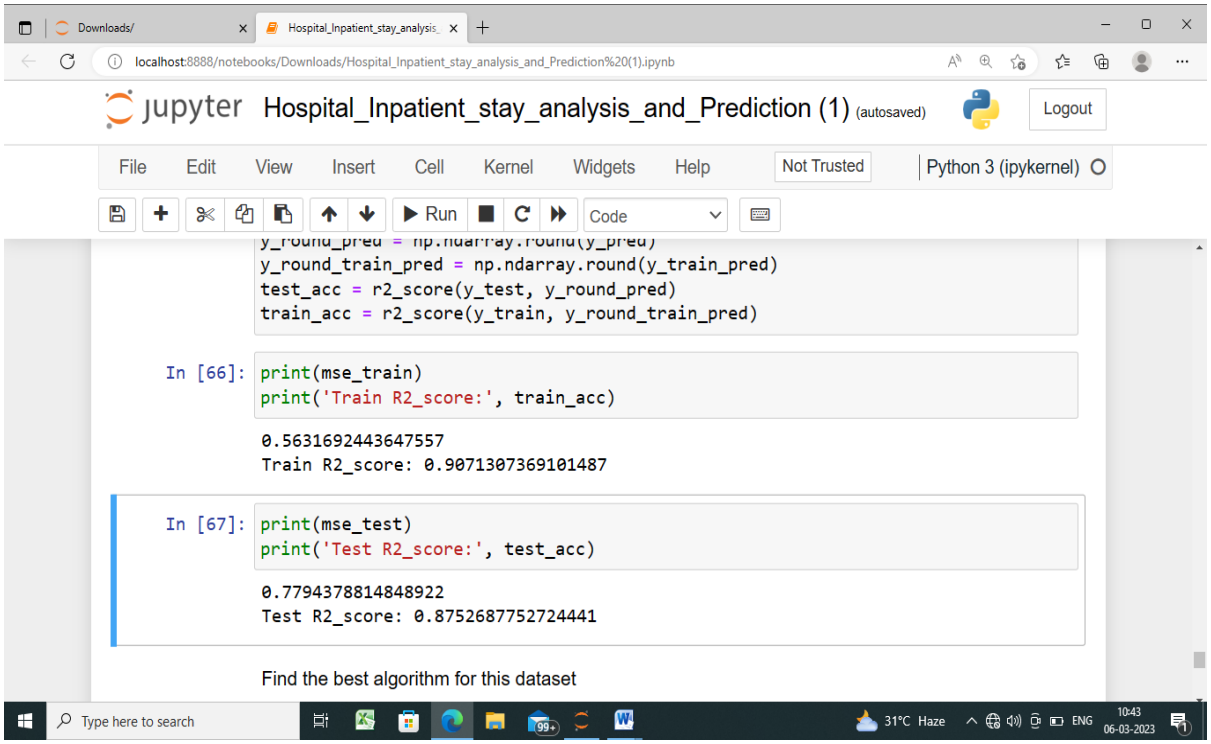


Figure : 6.33

CHAPTER -7

RESULT

7.1 Experinmental result

Each and every algorithm give differnet level of metrics score and we consider that evaluation metrics to select the best algorthim for this dataset to find the patient stay in the hospital.

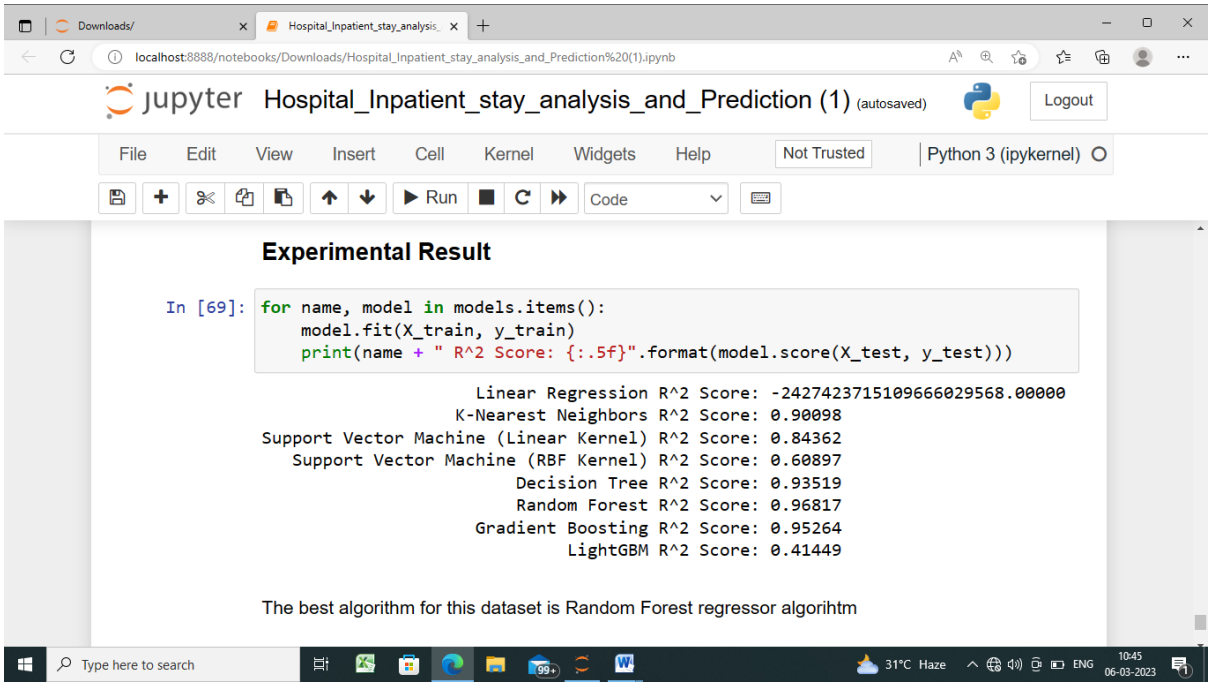


Figure : 7.1

While Random Forest Regression is a popular and effective algorithm for many prediction tasks, it may not necessarily be the best algorithm for all situations. The choice of algorithm ultimately depends on the specific characteristics of the data and the objectives of the analysis. Finally we choose the RandomForestRegressor algorithm it the score of 0.95997 is the best for this dataset for finding the patient stay in the hospital.

7.2 Prediction with new data

Finally we choose the Random Forest Regression is the best model for this dataset and we predict the patient stay in the hospital with given new data it will give the result of ten days .[10.597] .Given a set of input variables X = [x1, x2, ..., xn] and corresponding target variable y, the goal of Random Forest Regression is to learn a function f(x) that predicts the target variable for a new input point x. The function f(x) is defined as the average of the predictions of multiple decision trees, each trained on a randomly selected subset of the data and a random subset of the features

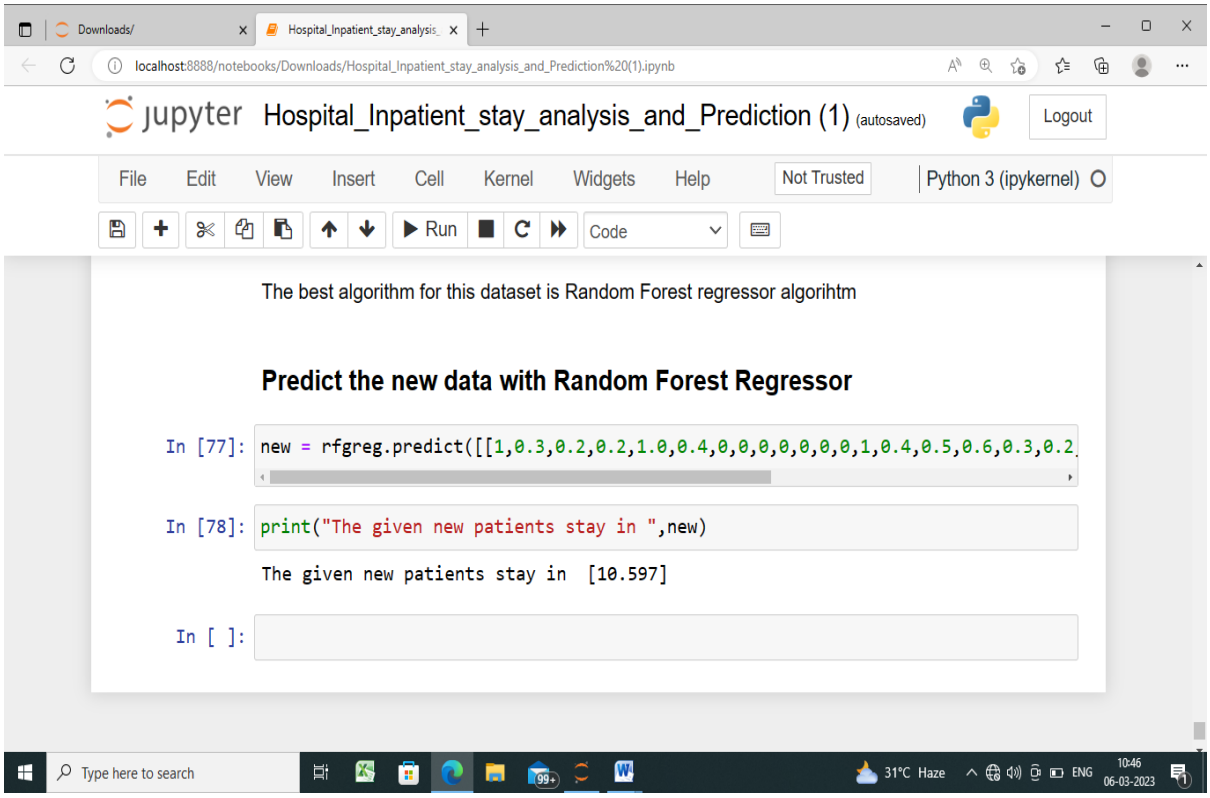


Figure : 7.2

CHAPTER - 8

CONCLUSION

In conclusion, hospital stay prediction using machine learning techniques has the potential to greatly improve healthcare outcomes by allowing healthcare professionals to better allocate resources, plan patient care, and reduce the overall length of hospital stays. Through the use of various algorithms and predictive models, machine learning can accurately predict the length of a patient's stay based on various factors such as demographics, comorbidities, and laboratory results. However, the effectiveness of these models is highly dependent on the quality and quantity of data used for training, as well as the choice of appropriate features and algorithms. Therefore, further research is needed to refine and improve the accuracy of hospital stay prediction models. Overall, the development and implementation of machine learning models for hospital stay prediction is an important step towards improving the efficiency and effectiveness of healthcare deliver.

REFERENCE

[1] S. Aghajani and M. Kargari, ‘‘Determining factors influencing length of stay and predicting length of stay using data mining in the general surgery department,’’ Hospital Practices Res., vol. 1, no. 2, pp. 51–56, May 2016.

[2] S. Barnes, E. Hamrock, M. Toerper, S. Siddiqui, and S. Levin, ‘‘Real-time prediction of inpatient length of stay for discharge prioritization,’’ J. Amer. Med. Inform. Assoc., vol. 23, no. e1, pp. e2–e10, Apr. 2016.

[3] P. Baylis, ‘‘Better health care with data mining,’’ SPSS, Shared Med. Syst. Ltd., London, U.K., 2009.

[4] C.-L. Chang and P.-Y. Lu, ‘‘The study on evaluating length of hospital stay for myomectomy,’’ Int. J. Sci. Eng. Invest., vol. 5, no. 59, pp. 157–162, 2016.

[5] M.-T. Chuang, Y.-H. Hu, C.-F. Tsai, C.-L. Lo, and W.-C. Lin, ‘‘The identification of prolonged length of stay for surgery patients,’’ in Proc. IEEE Int. Conf. Syst., Man, Cybern., Oct. 2015, pp. 3000–3003.

[6] C. Combes, F. Kadri, and S. Chaabane, ‘‘Predicting hospital length of stay using regression models: Application to emergency department,’’ in Proc. 10th Conférence Francophone de Modélisation, Optimisation et Simulation (MOSIM), 2014. [Online]. Available: https://hal.archivesouvertes.fr/hal-01081557

[7] E. El-Darzi, R. Abbi, C. Vasilakis, F. Gorunescu, M. Gorunescu, and P. Millard, ‘‘Length of stay-based clustering methods for patient grouping,’’ in Intelligent Patient Management (Series Studies), vol. 189. Berlin, Germany: Springer, 2009, pp. 39–56. VOLUME 9, 2021 44679 J. M. P. Gutiérrez et al.: Predicting LOS Across Hospital Departments

[8] E. Yasinski, D. Reilly, N. Duggal, B. S. Walker, E. Carpintero, S. Nag, and C. Hentz, ‘‘Understanding & predicting length of stay (LOS) using machine learning,’’ in Proc. Dexur. 15th Floor, 575 5th Avenue, New York, NY, USA, vol. 10017, 2017. [Online]. Available: https://dexur.com/a/mlresearch-los/6/