Mini-Project Report On

COLLEGE ADMISSION SUGGESTION SYSTEM

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science & Engineering

By

Alan Baby George (U2003019)
Alen T Babu (U2003026)
Alex Santhosh (U2003027)
Amal Stephen (U2003032)

Under the guidance of
Ms. Jyotsna A

Department of Computer Science & Engineering
Rajagiri School of Engineering and Technology (Autonomous)
Rajagiri Valley, Kakkanad, Kochi, 682039

July 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RAJAGIRI SCHOOL OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
RAJAGIRI VALLEY, KAKKANAD, KOCHI, 682039

CERTIFICATE

This is to certify that the mini-project report entitled ***"COLLEGE ADMISSION SUGGESTION SYSTEM"*** *is a bonafide work done by* ***Alan Baby George (U2003019), Alen T Babu (U2003026), Alex Santhosh (U2003027), Amal Stephen (U2003032)****, submitted to the RAJAGIRI SCHOOL OF ENGINEERING AND TECHNOLOGY in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2022-2023.*

Dr. Preetha K. G.	**Dr. Sminu Izudheen.**	**Ms. Jyotsna A**
Head of Department	Mini-Project Coordinator	Mini-Project Guide
Dept. of CSE	Professor	Asst. Professor
RSET	Dept. of CSE	Dept. of CSE
	RSET	RSET

ACKNOWLEDGEMENTS

We wish to express our sincere gratitude towards **Dr. P. S. Sreejith**, Principal of RSET, and **Dr. Preetha K. G.**, Head of Department of Computer Science and Engineering for providing us with the opportunity to undertake our mini-project, "College Admission Suggestion System".

We are highly indebted to our mini-project coordinators, **Dr. Sminu Izudheen**, Professor, Department of Computer Science and Engineering, and **Dr. Renu Mary Daniel**, Assistant Professor, Department of Computer Science and Engineering for their valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our mini-project guide **Ms. Jyotsna A**, for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Amal Stephen

Alex Santhosh

Alen T Babu

Alan Baby George

ABSTRACT

The college admission process can be overwhelming for students seeking to pursue higher education. With numerous colleges and courses available, it becomes challenging for students to identify suitable options based on their academic performance. Additionally, the lack of easily accessible and reliable information on college admission cutoffs further complicates the decision-making process. Therefore, there is a need for a college recommendation system that simplifies and streamlines the college selection process for students. The main purpose of the project is to reduce the challenges faced by high school graduates during the college admission process. It should provide personalized recommendations to students who are seeking guidance in identifying colleges that align with their academic aspirations and overall preferences. A suggestion system can help optimize the allocation of resources by guiding students towards colleges where they have a higher chance of acceptance, leading to more targeted applications and potentially reducing the burden on admission offices. College admission suggestion system has the potential to level the playing field by providing equal access to information and guidance for all students, regardless of their background or resources.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

1.1.1 Admission Process

The college admission process can be an overwhelming and complex journey for students aspiring to pursue higher education. With a vast array of colleges, universities, and courses available, making the right decision can be a daunting task, especially when it comes to identifying suitable options based on academic performance, interests, and career aspirations. Moreover, the lack of easily accessible and reliable information on college admission cutoffs further complicates the decision-making process.

To address these challenges and alleviate the stress that students face during college selection, there is a pressing need for an efficient and user-friendly college recommendation system. Such a system would aim to simplify and streamline the college selection process, providing personalized recommendations based on individual preferences and academic backgrounds.

1.1.2 Key features

Key features of an ideal college recommendation system might include:

- Academic Profiling: The system would allow students to input their academic performance, test scores, extracurricular activities, and any other relevant information. By analyzing this data, the system can generate a comprehensive academic profile for each student.

- Interest Assessment: The system should incorporate tools to gauge students' interests, passions, and career goals. This information is crucial in recommending colleges and courses that align with their aspirations.

- Comprehensive College Database: The system would maintain an up-to-date and extensive database of colleges, universities, and courses worldwide, along with relevant information about each institution's strengths, faculty, facilities, and placement records.

- Smart Matching Algorithm: Utilizing machine learning algorithms, the recommendation system would process the academic profiles and interests of students and perform a sophisticated matching process to suggest a list of suitable colleges and courses.Smart Matching Algorithm: Utilizing machine learning algorithms, the recommendation system would process the academic profiles and interests of students and perform a sophisticated matching process to suggest a list of suitable colleges and courses.

- Admission Cutoff Prediction: The system should analyze historical admission data and trends to predict potential cutoff ranges for different colleges and courses. This would provide students with insights into their chances of getting accepted.

- Reviews and Testimonials: Integrating reviews and testimonials from current students and alumni would offer valuable insights into the overall college experience, campus life, and the quality of education.

- Financial Aid and Scholarships: The system could also provide information on financial aid options, scholarships, and grants available for each institution, helping students make well-informed decisions considering their financial constraints.

- User-Friendly Interface: To ensure accessibility and ease of use, the recommendation system should have an intuitive and user-friendly interface that guides students through the process step by step.

- Privacy and Security: Data privacy and security are paramount when dealing with sensitive information. The system must adhere to strict privacy protocols and comply with relevant regulations to protect students' personal data.

- Continuous Improvement: The recommendation system should be regularly updated to adapt to changing admission criteria, new colleges, and emerging trends in education.

Cognitive disabilities range from less serious disabilities (like attention deficit and dyslexia disorder) to more serious disabilities (like hereditary diseases and brain damage).

1.2 Existing System

- Cappex: Cappex is an online college search and recommendation platform that helps students find colleges based on their academic profile and preferences. Reference: Cappex. (n.d.). Retrieved from https://www.cappex.com/
- College Board's BigFuture: College Board's BigFuture platform provides college search and recommendation features, allowing students to explore colleges and get personalized recommendations based on their preferences. Reference: BigFuture. (n.d.). Retrieved from https://bigfuture.collegeboard.org/
- Niche: Niche is a website that provides comprehensive college search and recommendation tools, including personalized college recommendations based on various factors. Reference: Niche. (n.d.). Retrieved from https://www.niche.com/
- CollegeVine: CollegeVine offers a college admissions guidance platform that includes a college recommendation system based on a student's profile and preferences. Reference: CollegeVine. (n.d.). Retrieved from https://www.collegevine.com/

1.3 Problem Statement

The challenge is to create a web application which will provide assistance to student who is searching for a college to pursue higher education in a college or a university.

The college admission process can be overwhelming for students seeking to pursue higher education. With numerous colleges and courses available, it becomes challenging for students to identify suitable options based on their academic performance. Additionally, the lack of easily accessible and reliable information on college admission cutoffs further complicates the decision-making process. Therefore, there is a need for a college recommendation system that simplifies and streamlines the college selection process for students.

1.4 Objectives

- **Develop a user-friendly interface**: Create an easy-to-use interface that allows students to input their academic qualifications, preferences, and other relevant information effectively.
- **Personalise recommendations**:Utilize machine learning algorithms and personalized criteria to generate tailored recommendations for students based on their academic qualifications.
- **Streamline the college selection process**: Simplify and streamline the college selection process by reducing information overload, saving time, and providing guidance to students throughout their decision-making journey.
- **Enhance accessibility**: Ensure the system is accessible to a wide range of students, considering diverse backgrounds and resources.
- **Enable Administrative Controll**: Develop an admin interface that allows administrators to manage college information, predicted cutoffs, and other relevant data, ensuring up-to-date and accurate information.

1.5 Scope

The system will allow students to register and create profiles, where they can input their academic details, standardized test scores, extracurricular activities, and any specific preferences they have for colleges.A comprehensive database of colleges and universities will be created, containing relevant information such as courses offered, admission cutoffs, campus facilities, faculty, location, and any other essential factors that might influence a student's decision.

The system will gather information from various sources, including official college websites, government databases, and educational institutions, to keep the college database up-to-date and accurate.An intelligent recommendation algorithm will be developed, which takes into account the student's profile and preferences, and matches them with suitable colleges. The algorithm might use machine learning techniques to improve its accuracy over time.The platform will allow students to interact with the system, fine-tune their

preferences, and view personalized recommendations. Students should also have the option to provide feedback on the recommendations they receive.

The system will provide visualizations and comparison tools, allowing students to compare multiple colleges based on various parameters like admission cutoffs, course offerings, and campus facilities.The platform should be accessible via web browsers to reach a broader audience of students.Adequate measures will be taken to ensure the privacy and security of user data, complying with all relevant data protection regulations. Rigorous testing will be conducted to verify the accuracy and reliability of the recommendation algorithm and the overall system functionality.The system should be designed with scalability in mind to accommodate a growing number of users. Additionally, provisions should be made for future enhancements and feature updates to keep the system relevant and up-to-date.

Chapter 2

Literature Review

2.1 Collaborative-Filtering Recommendation Systems

Collaborative Filtering evaluates products using users' ratings (explicit or implicit) from historical data. It works by developing a database of the user's preferences for items. Active users will be mapped against this database to reveal the active user's neighbors with similar purchase preferences. Collaborative filtering techniques are classified into item-based filtering and user-based filtering. User-based techniques go through two main stages to forecast items' ratings for a specific user. The first stage locates similar users to the target user. The second stage obtains rates from similar users to the active user, then using them to produce recommendations. There have been many collaborative filtering algorithm measures that calculate the similarities among users. The commonly used similarity measures in the literature include mean-squared difference, Pearson correlation, cosine similarity, Spearman correlation, and adjusted cosine similarity.An advantage of collaborative filtering is that it generates models that help users discover new interests.Although being favorable in many aspects, collaborative filtering has several disadvantages, such as the cold-start problem.

2.2 Content-Based Recommendation Systems

Content-based approaches attempt to build a user profile to predict ratings on unseen items. Successful content-based methods utilize tags and keywords. Measuring the utility of content-based filtering is commonly calculated by using heuristic functions, such as the cosine similarity metric. Content-based filtering can be employed in many cases, where the features' values can easily be extracted. Content-based filtering is not typically used in cases where features values must be manually entered. This can be manageable for small datasets, but when thousands of new products are being added daily, this task is

impossible. Content-based filtering does not require other users' data, as the predicted recommendations are user-specific. Thus, these techniques scale up the system to handle many users. Content-based filtering is user-independent since this system only requires analyzing the items and user profile for recommendations. Opposite of collaborative filtering, content-based filtering does not experience cold-start issues.If no enough information is provided in the content to differentiate products precisely, the recommendation will not be accurate. These techniques require intensive domain knowledge.Content-based systems offer a limited degree of novelty since they must match up the features of profiles and items.

2.3 Demographic-Based Recommendation Systems

As various quantitative research papers have displayed, collaborative filtering techniques can be enhanced by demographic correlation. Demographic RSs can generate recommendations by categorizing users based on demographic attributes. Demographic RSs are especially useful when the amount of product information is limited. Demographic RSs aim to tackle and solve the scalability and cold-start problems. This system employs user attributes as demographic data to obtain recommendations (i.e., recommend products based on age, gender, language, etc.). The key advantage of demographic filtering RSs is that they are fast and straightforward in obtaining results using a few observations. These approaches also do not acquire the user ratings that are essential in content-based and collaborative-based filtering techniques. Demographic-based filtering techniques have several disadvantages. For example, the entire information collection for users is impractical, considering the security and privacy issues involved. Secondly, demographic filtering is mainly based on user interests, which forces the system to recommend the same item to users of related demographic profiles. Another challenge is the difficulty of modifying a customer profile when preferences change; this is known as the stability vs. plasticity problem.

2.4 Utility-Based Recommendation Systems

Utility-based RS provides recommendations based on generating a utility model of each item for the user. This system builds multi-attribute users' utility functions and rec-

ommends the highest utility item based on each item's calculated user-utility explicitly. Utility-based RSs are useful because they can factor non-product attributes into utility functions, such as product availability and vendor reliability. They generate utility computation, which allows them to check both real-time inventory and features of an item. It enables the visualization of its status to the user. Utility-based systems do not hold on to long-term generalizations about their users. Instead, they evaluate a recommendation based on the user's current needs and the available options. A disadvantage of the utility-based system occurs when the products are not descriptive enough. They do not contain enough listed utility features; that could hide a recommendation to a user even if it fits that particular user's preferences.

2.5 Knowledge-Based Recommendation Systems

Knowledge-based RS uses explicit knowledge about products and users to create a knowledge-based criterion to generate recommendations. A knowledge-based RS does not require an initial large amount of data, as its recommendations are independent of the user's ratings. It recommends items based on the user's preferences by evaluating the products that meet the user's needs. Knowledge-based RSs are noted to be advantageous for several purposes. For example, they can avoid the typical ramp-up problem associated with machine learning approaches to recommendations. Typically, exemplary systems cannot learn until the user has rated many items. Knowledge-based RSs avoid this issue since their recommendations are not dependent on a base of user ratings. They also do not need to gather information about a particular user because the recommendations are independent of the user's tastes too. Due to these factors, knowledge-based systems are valuable as standalone systems, and they are also considered complementary to other types of RSs. One major disadvantage of knowledge-based RSs is the potential knowledge acquisition bottleneck caused by explicitly defining recommendation knowledge. Knowledge acquisition is the process of constructing the rules and requirements needed for a knowledge-based system, and it is done by gaining knowledge via rules, objects, and frame-based ontologies. Batesonian theories were used to guide the process of further learning of knowledge acquisition.

2.6 Predicting College Admission Cutoffs using Regression Analysis

This study proposes a regression-based approach to predict college admission cutoffs based on students' past academic performances and other relevant parameters. The authors employ a linear regression model to predict cutoff scores for various colleges, providing insights into how different features influence admission decisions.This research paper introduces an ensemble regression model that combines multiple regression algorithms to predict college cutoffs more accurately. The authors evaluate the performance of linear regression, support vector regression, and random forest regression in this context, highlighting the advantages of ensemble techniques.Focusing on feature selection, this research investigates the impact of various attributes on college cutoff predictions. The authors compare different regression models and identify the most influential features that significantly affect the recommendation system's performance.Regression models play a vital role in college recommendation systems by predicting college admission cutoffs based on historical data and relevant student information.

Chapter 3

System Analysis

3.1 Expected System Requirements

The system of admin which is a computer is expected to have the following features:

- Windows platform with windows 8 or above.
- A storage space of approximate 100 MB for the app and database .
- A minimum Ram size of 4GB is required in the device.
- 64 bit architecture
- Intel i3 Processor or above

The system of user which is also a pc is expected to have the following features:

- Windows platform with windows 8 or above.
- 64 bit architecture
- Intel i3 Processor or above
- Requirement of Internet connection for running the website.
- A minimum Ram size of 2GB is required in the device.
- Windows platform with windows 8 or above.

3.2 Feasibility Analysis

3.2.1 Technical Feasibility

The project is technically feasible since majority of the population are in possession of windows laptops or pcs. The app only requires minimum requirements to run on a pc .

3.2.2 Operational Feasibility

The operations are built in a simple and easy to use manner for students. No installation of the app is not required. Only admin has to install the application .

3.2.3 Economic Feasibility

The app can reduce the overhead of expense incurred by students in order to find an appropriate college they are eligible to get into. The development of the app is also zero budget as it was built using free resources.

3.3 Hardware Requirements

The following are the system requirements to develop the Unify App.

- Processor: Intel Core i3
- Hard Disk: Minimum 100GB
- RAM: Minimum 4GB

3.4 Software Requirements

The following are the softwares used in the development of the app.

- Operating System: Windows or Linux
- Python 3
- Sqlite3
- Regression model
- Tkinter
- Flask
- HTML and CSS
- SQL

3.4.1 Visual Studio Code

Visual Studio Code (VS Code) is a widely used code editor with excellent support for Python coding. After installing the Python extension, developers can create and run Python files easily. The integrated terminal allows running scripts, while the debugging feature helps identify and fix issues efficiently. With language services like IntelliSense, code completion, and syntax highlighting, writing Python code becomes more productive. Optionally, managing virtual environments within VS Code keeps projects organized. Overall, the combination of VS Code and Python extension provides a powerful and user-friendly environment for Python development.

3.4.2 SQLite 3

SQLite is a lightweight, self-contained, serverless, and open-source relational database management system (RDBMS). It is built on the C programming language and operates on a single file, making it easy to set up and use without the need for a separate server or complex configurations. SQLite is widely used in various applications, including mobile apps, web browsers, operating systems, and embedded systems, due to its small footprint, efficiency, and portability. Despite being lightweight, SQLite supports most standard SQL features, providing a reliable and efficient way to store and manage relational data. It is an ideal choice for projects that require a local database or need a database that can be easily distributed along with the application.

3.4.3 Flask

Flask is a lightweight and flexible Python web framework that enables seamless connectivity between Python code and HTML templates. Through its support for HTML templates and template engines like Jinja2, Flask allows developers to render dynamic web pages by inserting dynamic content into placeholders within the HTML. With Flask's routing capabilities, developers can associate specific Python functions with URL patterns, enabling the processing of data, database interactions, and rendering of dynamic content based on user requests. Additionally, Flask simplifies form handling, making it easy to receive user input, process it in Python, and respond accordingly. Its support for serving static files further enhances the user interface of web applications. Overall,

Flask's HTML connectivity empowers developers to create interactive and data-driven web applications efficiently and effectively.

3.4.4 Tkinter

Tkinter is a standard Python library used for creating graphical user interfaces (GUIs). It provides a set of tools and widgets that allow developers to build interactive desktop applications with ease. Tkinter is based on the Tcl/Tk GUI toolkit and is included with most Python installations, making it readily available without requiring any additional installations. Developers can design windows, buttons, labels, text boxes, and various other GUI components, and then bind them with event handlers to respond to user interactions. Tkinter's simplicity and versatility make it suitable for both beginners and experienced developers, enabling them to create cross-platform applications that run seamlessly on Windows, macOS, and Linux systems. Despite its simplicity, Tkinter can be used to develop sophisticated desktop applications with a wide range of functionalities.

3.4.5 Scikit-learn

Scikit-learn, often referred to as sklearn, is a widely used and powerful machine learning library for Python. It provides a rich set of tools and algorithms for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and more. Sklearn is built on top of other popular scientific libraries in Python, such as NumPy, SciPy, and matplotlib, making it a comprehensive and efficient solution for data analysis and modeling. With its easy-to-use interface and extensive documentation, sklearn is favored by data scientists and machine learning practitioners to develop and deploy machine learning models effectively. From preprocessing data to evaluating model performance, sklearn offers a unified and consistent API, enabling users to build sophisticated machine learning pipelines with minimal effort.

Chapter 4

Methodology

Block diagram + Explain each block.

4.1 Proposed Method

The proposed method outlines the approach to developing a college admission system that assists students in finding suitable colleges based on their KEAM rank, desired course, and preferred location. The system aims to provide accurate predictions of college cutoffs using machine learning algorithms and facilitate an informed decision-making process for prospective college applicants.

- Data Collection and Preprocessing: Collect relevant data from official college admission websites and other reliable sources. Preprocess the data to handle missing values, remove duplicates, and perform data cleaning. Engineer features to enhance the predictive capability of the machine learning models.

- Machine Learning Models: Utilize various machine learning algorithms to predict college cutoffs, taking into account factors such as course, location, and KEAM rank. Evaluate and compare the performance of different algorithms (Linear Regression, Polynomial Regression, Random Forest, and Support Vector Regression) to select the best-performing model.

- User Interface Development: Design an intuitive and user-friendly web-based interface to interact with the college admission system. Implement input forms for users to enter their KEAM rank, course preferences, and desired location. Create an output page to display the list of colleges matching the user's criteria, including predicted cutoffs and other relevant information.

- Backend and Database Integration: Develop the backend of the system using Flask, a Python web framework, to handle user requests and process data. Integrate the SQLite database to store college data, including cutoffs, course details, and locations.

- Error Handling and Validation: Implement server-side validation to ensure that user inputs are valid and meet the required criteria. Display appropriate error messages if invalid inputs, such as non-numeric or negative ranks, are provided.

- Performance Evaluation: Measure the performance of the machine learning models using metrics such as Mean Absolute Error (MAE) and accuracy. Analyze the model results to assess their reliability and accuracy in predicting college cutoffs.

4.1.1 Machine Learning Models

Linear Regression

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable. The linear regression model provides a sloped straight line representing the relationship between the variables.

Mathematically, we can represent a linear regression as: y= a0+a1x+ E where, Y= Dependent Variable (Target Variable) X= Independent Variable (predictor Variable) a0= intercept of the line (Gives an additional degree of freedom) a1 = Linear regression coefficient (scale factor to each input value). E = random error

The values for x and y variables are training datasets for Linear Regression model representation.

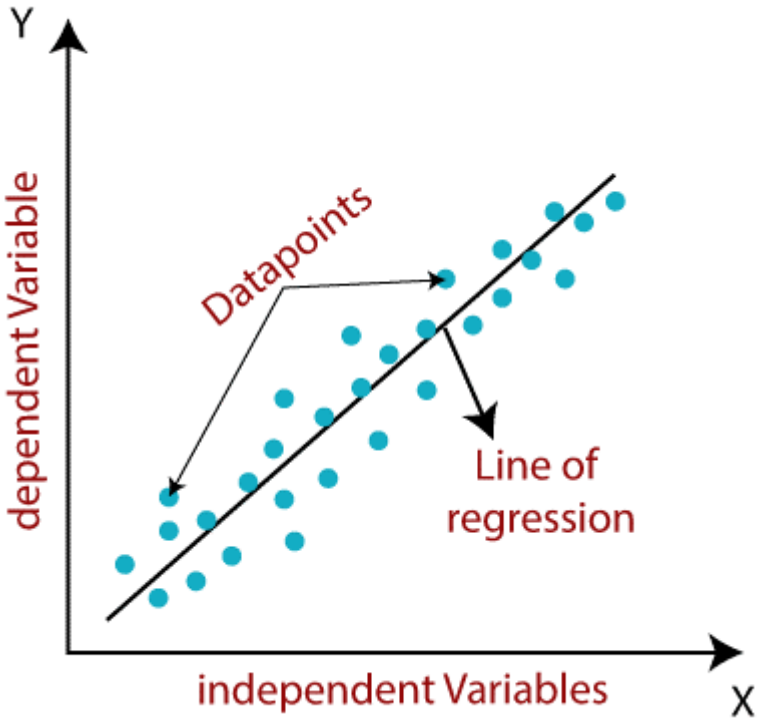

Figure 4.1: Linear Regression Graph

Linear Regression Line:

Positive Linear Relationship: If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship

Negative Linear Relationship: If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error. For Linear Regression, we use the Mean Squared Error (MSE) cost function, which is the average of squared error occurred between the predicted values and actual values. It can be written as:

N=Total number of observation

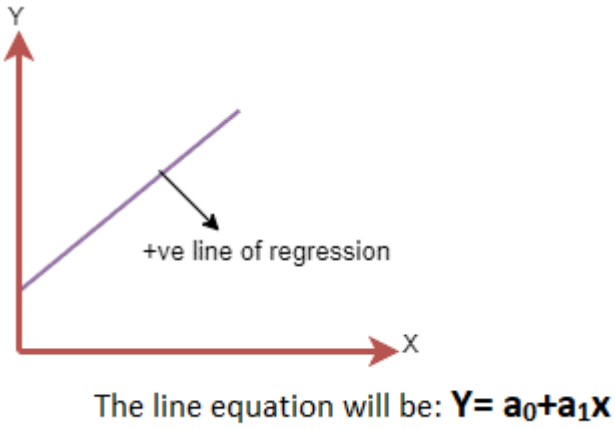

Figure 4.2: Positive line of regression

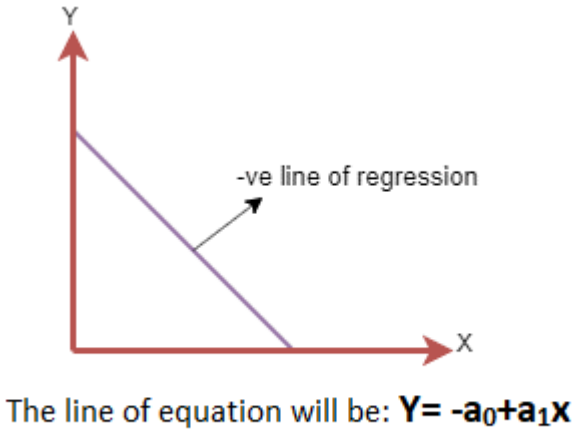

Figure 4.3: Negative line of regression

Yi = Actual value

(a1xi+a0)= Predicted value.

The distance between the actual value and predicted values is called residual. If the observed points are far from the regression line, then the residual will be high, and so cost function will high. If the scatter points are close to the regression line, then the residual will be small and hence the cost function.

$$MSE = 1\frac{1}{N}\sum_{i=1}^{n}(y_i - (a_1 x_i + a_0))^2$$

Figure 4.4: Equation

Random Forest Regression

Random forest regression is a supervised learning algorithm that uses an ensemble learning method for regression. The trees in random forests run in parallel, meaning there is no interaction between these trees while building the trees. Decision Trees are used here. To get a better understanding of what a decision tree is: It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

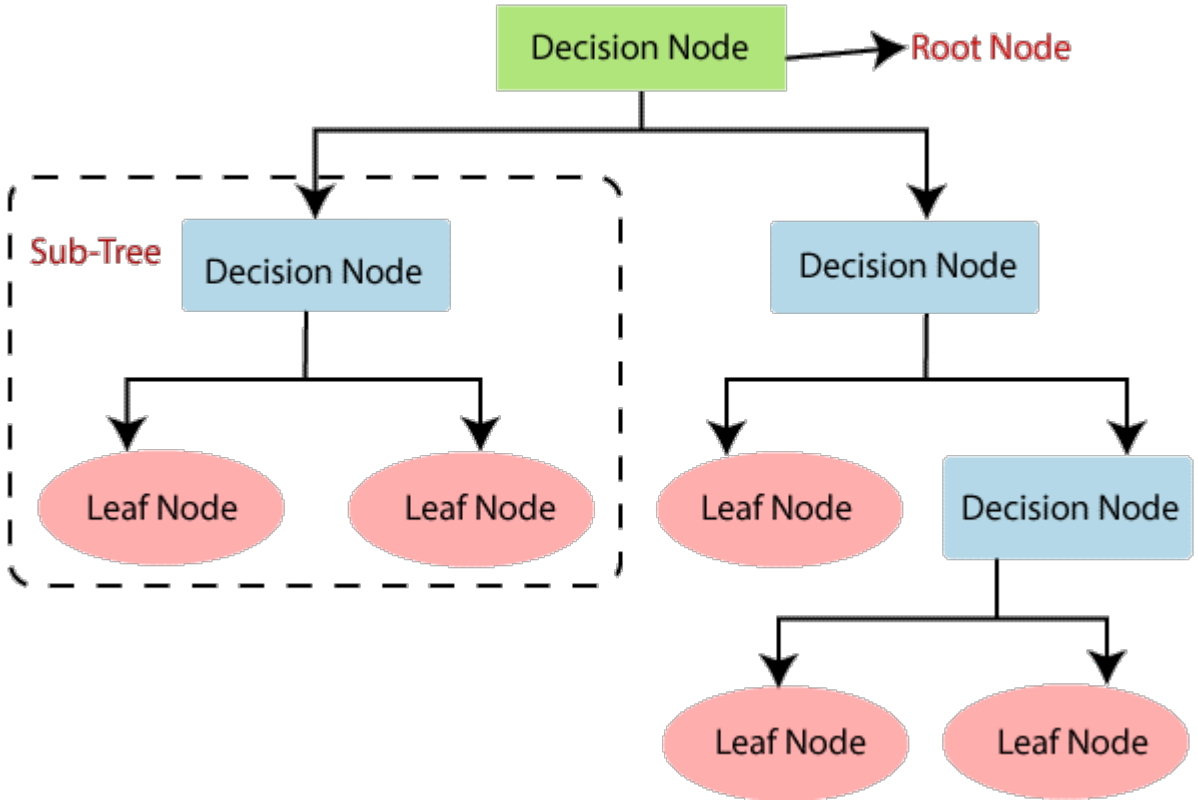

Figure 4.5: Decision tree

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that

dataset.Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Random forest operates by constructing a multitude of decision trees at training time and outputting the class that's the mode of the classes (classification) or mean prediction (regression) of the individual trees. A random forest is a meta-estimator (i.e. it combines the result of multiple predictions), which aggregates many decision trees with some helpful modification:

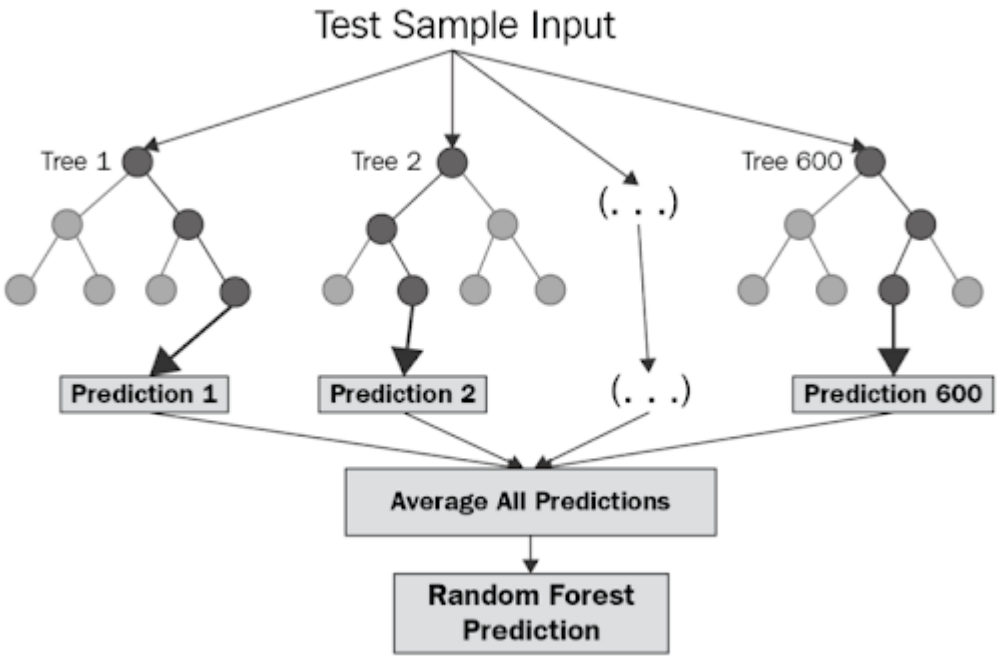

Figure 4.6: Aggregation of decision trees

• Each tree draws a random sample from the original data set when generating its splits, adding a further element of randomness that prevents overfitting.

The above modifications help prevent the trees from being too highly correlated.

Polynomial Regression

Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as nth degree polynomial. The Polynomial Regression equation is given below: y= b0+b1x1+ b2x12+ b2x13+..... bnx1n

In polynomial regression, the relationship between the dependent variable and the independent variable is modeled as an nth-degree polynomial function. When the polynomial is of degree 2, it is called a quadratic model; when the degree of a polynomial is 3, it is called a cubic model, and so on.

The degree of order which to use is a Hyperparameter, and we need to choose it wisely. But using a high degree of polynomial tries to overfit the data, and for smaller values of degree, the model tries to underfit, so we need to find the optimum value of a degree. Polynomial Regression models are usually fitted with the method of least squares. The least square method minimizes the variance of the coefficients under the Gauss-Markov Theorem.

It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression. It is a linear model with some modification in order to increase the accuracy. The dataset used in Polynomial regression for training is of non-linear nature. It makes use of a linear regression model to fit the complicated and non-linear functions and datasets.

The main steps involved in Polynomial Regression are given below:

- Data Pre-processing
- Build a Linear Regression model and fit it to the dataset
- Build a Polynomial Regression model and fit it to the dataset
- Visualize the result for Linear Regression and Polynomial Regression model.
- Predicting the output.

Need For Polynomial Regression:

If we apply a linear model on a linear dataset, then it provides us a good result as we have seen in Simple Linear Regression, but if we apply the same model without any modification on a non-linear dataset, then it will produce a drastic output. Due to which loss function will increase, the error rate will be high, and accuracy will be decreased.So for such cases, where data points are arranged in a non-linear fashion, we need the Polynomial Regression model. We can understand it in a better way using the below comparison diagram of the linear dataset and non-linear dataset.

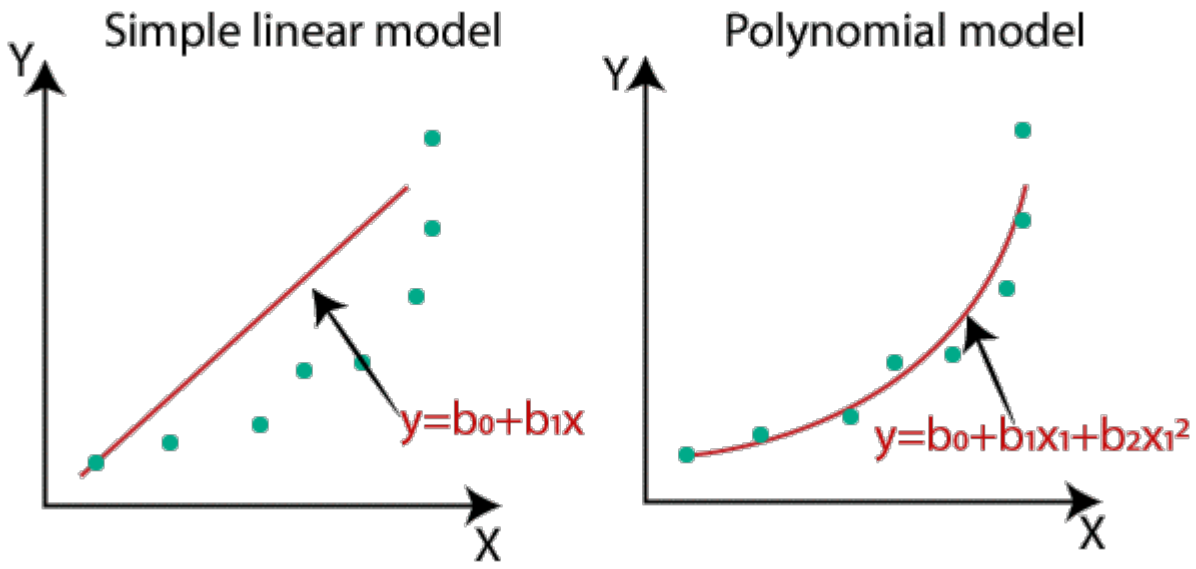

Figure 4.7: Polynomial regression graph

In the above image, we have taken a dataset which is arranged non-linearly. So if we try to cover it with a linear model, then we can clearly see that it hardly covers any data point. On the other hand, a curve is suitable to cover most of the data points, which is of the Polynomial model. Hence, if the datasets are arranged in a non-linear fashion, then we should use the Polynomial Regression model instead of Simple Linear Regression.

Support Vector Regression

Support Vector Regression (SVR) is a type of machine learning algorithm used for regression analysis. The goal of SVR is to find a function that approximates the relationship

between the input variables and a continuous target variable, while minimizing the prediction error. Unlike Support Vector Machines (SVMs) used for classification tasks, SVR seeks to find a hyperplane that best fits the data points in a continuous space. This is achieved by mapping the input variables to a high-dimensional feature space and finding the hyperplane that maximizes the margin (distance) between the hyperplane and the closest data points, while also minimizing the prediction error.

SVR can handle non-linear relationships between the input variables and the target variable by using a kernel function to map the data to a higher-dimensional space. This makes it a powerful tool for regression tasks where there may be complex relationships between the input variables and the target variable.

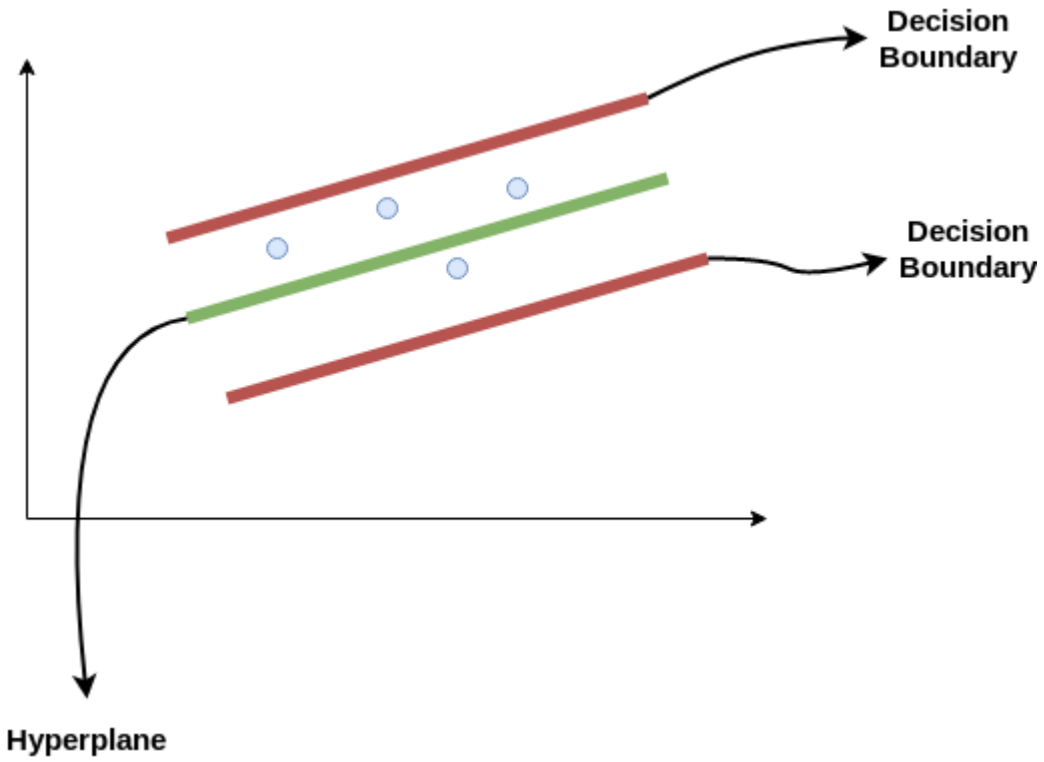

Figure 4.8: Support Vector Regression Graph

Consider these two red lines as the decision boundary and the green line as the hyperplane. Our objective, when we are moving on with SVR, is to basically consider the points that are within the decision boundary line. Our best fit line is the hyperplane that has a maximum number of points.

Our main aim here is to decide a decision boundary at ‘a’ distance from the original hyperplane such that data points closest to the hyperplane or the support vectors are within that boundary line. Consider these lines as being at any distance, say ‘a’, from the hyperplane. So, these are the lines that we draw at distance ‘+a’ and ‘-a’ from the hyperplane. This ‘a’ in the text is basically referred to as epsilon.

Consider these lines as being at any distance, say ‘a’, from the hyperplane. So, these are the lines that we draw at distance ‘+a’ and ‘-a’ from the hyperplane. This ‘a’ in the text is basically referred to as epsilon.

Y = wx+b (equation of hyperplane)

Then the equations of decision boundary become:

wx+b = +a

wx+b = -a

Thus, any hyperplane that satisfies our SVR should satisfy:

-a ¡ Y- wx+b ¡ +a

Chapter 5

System Design

5.1 Architecture Diagram

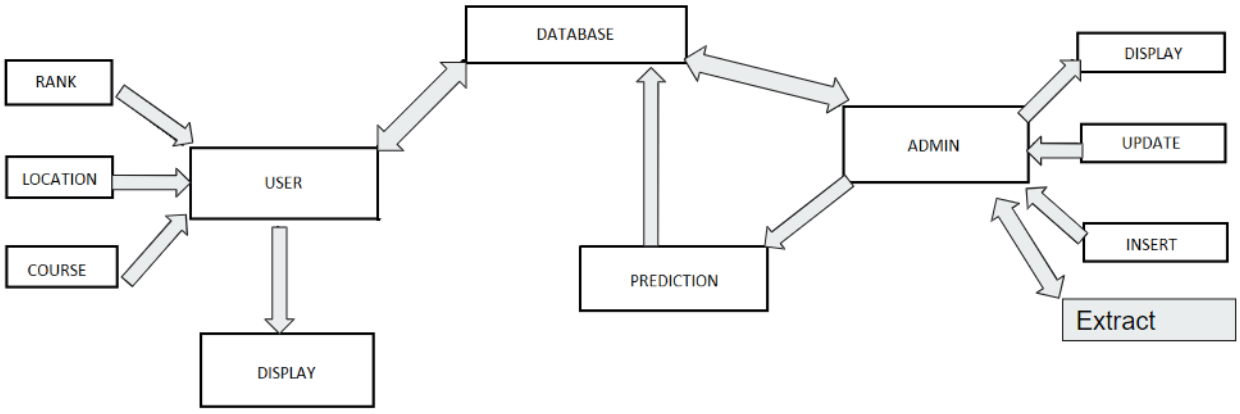

Figure 5.1: Architecture diagram

5.2 Flow diagram

5.2.1 User Interface

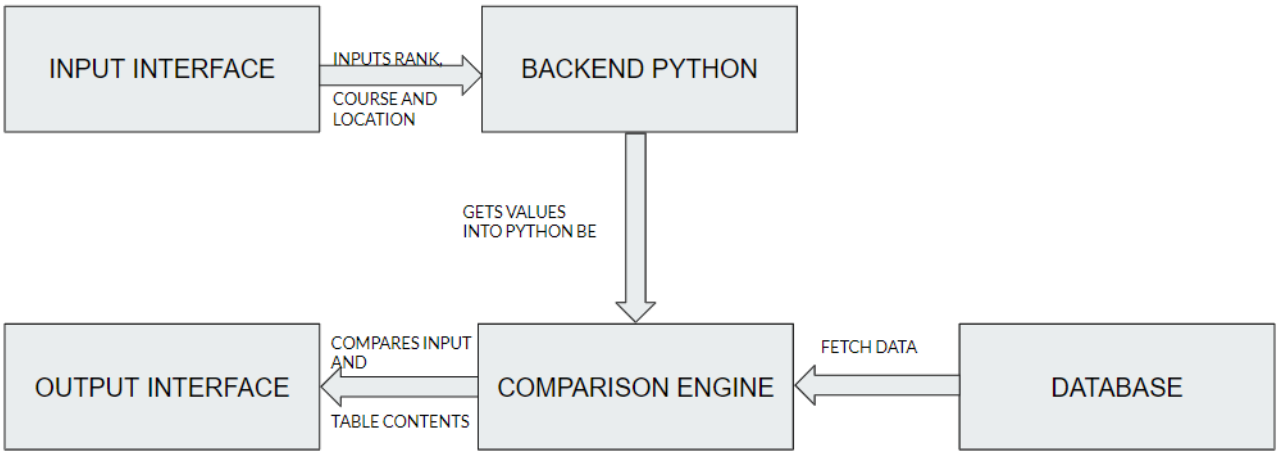

Figure 5.2: User Interface

5.2.2 Admin Interface

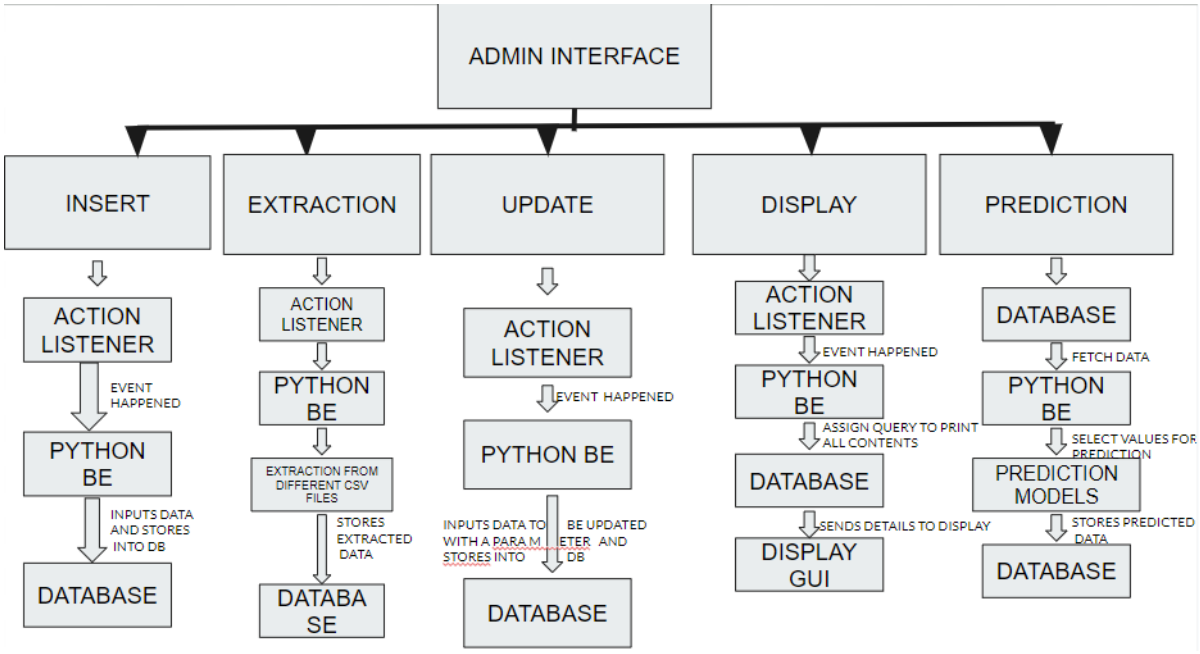

Figure 5.3: Admin interface

5.3 UML diagram

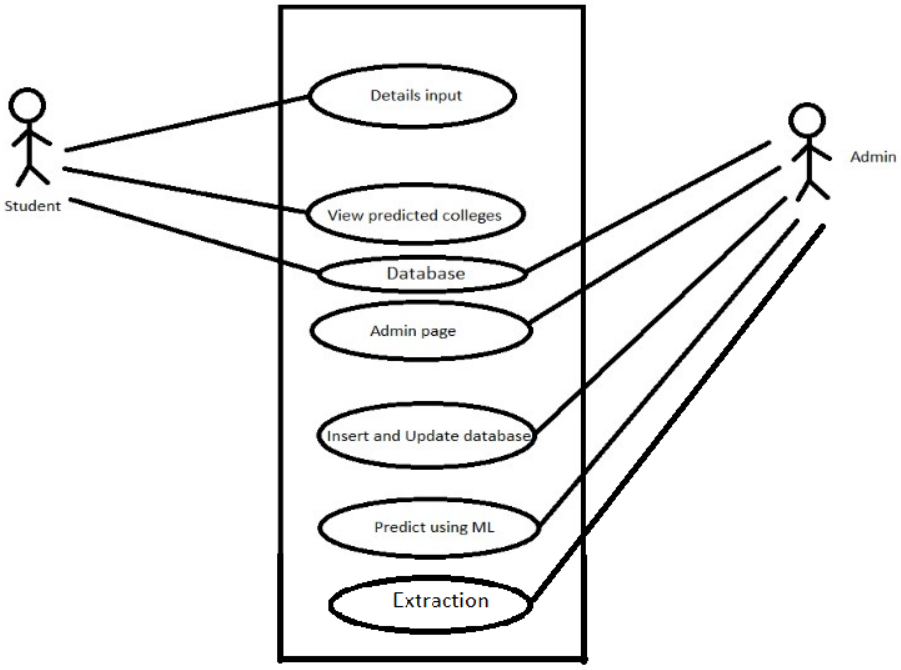

Figure 5.4: UML diagram

The UML diagram depicts a system that comprises two primary actors: User and Admin. Users are granted access to user details input and can view predicted data. On the other hand, administrators possess extended privileges, allowing them to interact with the database admin GUI, database itself, and utilize both prediction and extraction algorithms. The User and Admin classes contain relevant attributes such as UserID, Username, and Password to distinguish their roles, and a Database class functions as the central data repository.

Furthermore, the system includes two essential components: the Prediction Algorithm class, which facilitates data prediction based on specific inputs or patterns, and the Extraction Algorithm class, responsible for extracting relevant information from the stored data. Admins can efficiently manage the database using the Database Admin GUI class, which provides a user-friendly interface for their interactions. This comprehensive UML diagram visually represents the interconnections and functionalities of the system, streamlining the implementation process and fostering a clear understanding of how users and admins interact with the various components.

Chapter 6

System Implementation

6.1 User Interface

The user interface (UI) is designed using HTML to create a user-friendly web form. The form includes input fields where the user can enter their rank, desired course name, and preferred location. Additionally, there is a "Submit" button at the bottom of the form. When the user enters the required information and clicks the "Submit" button, the form triggers a Python program running in the backend.

The Python program is responsible for handling the user's input and querying the database. It retrieves the entered rank and searches for a corresponding match in the "predicted ranks" column of the database. The database contains information about various colleges, including their predicted ranks for different courses and locations. Once the Python program finds the matches based on the entered rank, it fetches the corresponding colleges and their details from the database. Finally, the Python program generates a response, and the user interface displays the list of colleges that match the entered criteria, providing the user with valuable insights on the colleges they are eligible for based on their rank and preferences. This interactive and efficient system helps students make informed decisions about their college choices.

6.2 Database

The college database is a comprehensive repository that holds crucial information about various colleges. It includes essential details such as the college name, college code, course offerings, and location of each college. Additionally, the database records the National Institutional Ranking Framework (NIRF) ranking of each college, providing insights into their overall academic and research excellence. For students seeking accommodation, the database also indicates the availability of hostel facilities at each college, ensuring that

they can make informed decisions about their living arrangements.

Moreover, the database captures the cutoff scores for the past five years, giving prospective students an understanding of the competitiveness of admission to specific courses at each college. This historical data is invaluable in helping students gauge their chances of securing admission to their desired courses. To further enhance the recommendation system, the database includes four additional columns that contain the predicted values obtained through machine learning algorithms. These algorithms use historical data and various parameters to predict the expected cutoff scores for the current year. When a user inputs their rank through the user interface, the system efficiently compares this rank with the predicted values in the respective columns. By doing so, the recommendation system can suggest colleges that align with the user's rank and preferences, offering personalized and accurate college recommendations.

6.3 Admin gui

The Admin GUI is a powerful and intuitive interface designed to streamline various tasks with five main objectives: insert, extraction, update, display, and prediction. The "insert" function offers a user-friendly GUI that facilitates manual input of data using SQL commands, enabling administrators to seamlessly add new records to the database. On the other hand, the "extraction" GUI utilizes a custom code to extract data from an Excel sheet and efficiently populate the database, ensuring a smooth and automated process for data import.

The "update" function plays a vital role in maintaining data accuracy and integrity. It allows administrators to make necessary changes or corrections to existing records in the database, ensuring that any discrepancies or errors are promptly rectified. The "display" feature showcases the entire database in an organized and visually appealing manner, providing a comprehensive overview of all college-related information at a glance.

One of the most innovative aspects of the Admin GUI is the "prediction" function. Leveraging the power of machine learning, this feature employs four algorithms – linear regression, MLP regression, SVR regression, and random forest – to predict cutoff scores based on historical data from the last five years. The predicted values are then stored in a dedicated column within the database, empowering administrators with valuable insights

for future college admissions. With its user-friendly design and sophisticated functionalities, the Admin GUI serves as an invaluable tool for college administrators, streamlining data management and analysis while enabling accurate predictions for improved decision-making.

6.4 Machine Learning Algorithms

Linear Regression is a fundamental and widely used supervised machine learning algorithm for predicting continuous values. In the context of predicting college cutoffs, it takes into account the historical cutoff data from the last five years as input and fits a straight line to the data points. The linear regression model then uses this line to predict the cutoff for the next year based on the input features. It is a simple and interpretable model, making it an excellent starting point for prediction tasks.

Multilayer Perceptron (MLP) Regression is a type of artificial neural network with multiple layers of interconnected nodes. It excels at capturing complex patterns and relationships within the data. In the context of college cutoff prediction, the MLP regression algorithm leverages the cutoff data of the last five years as input and learns the underlying patterns between the features and the target cutoff values. This powerful algorithm can handle non-linear relationships, making it suitable for more intricate prediction tasks where the cutoffs may exhibit non-linear trends.

Support Vector Regression (SVR) is a regression algorithm that is particularly effective for datasets with non-linear relationships. It works by finding a hyperplane that best fits the data while allowing some tolerance for errors. In the college cutoff prediction scenario, SVR takes the cutoff data from the last five years and constructs a model that can predict the cutoff for the next year while considering the patterns and fluctuations in the historical data. SVR is well-suited for scenarios where the data may not follow a linear trend, providing more accurate predictions in such cases.

Random Forest Regression is an ensemble learning method that combines multiple decision tree regressors to create a robust and accurate prediction model. In the context of predicting college cutoffs, the algorithm employs the historical cutoff data from the last five years to build a collection of decision trees. Each tree contributes to the final prediction, and the algorithm averages the outputs to provide a reliable estimate for the

cutoff of the next year. Random Forest is highly effective in handling noisy or complex data and mitigating overfitting, making it an excellent choice for college cutoff prediction tasks where the data may have uncertainties or variations.

In summary, these four machine learning algorithms – Linear Regression, MLP Regression, SVR Regression, and Random Forest Regression – each offer unique strengths and capabilities for predicting college cutoffs based on historical data. By leveraging their respective approaches, the college administration system can provide more accurate and reliable predictions to assist students in making well-informed decisions about their college choices.

6.5 Extraction of data

The program is designed to efficiently extract relevant details from an Excel sheet and subsequently process and store the data into a database while adhering to predefined conditions. To achieve this, the program utilizes the pandas library to read the data from the Excel sheet into a DataFrame, enabling seamless data manipulation and analysis. The data is carefully processed and filtered based on the given conditions, ensuring that only the necessary information is retained for storage.

Upon processing the data, the program leverages the sqlite3 library to interact with the database. The data is then inserted into the appropriate tables within the database, ensuring proper organization and storage. The program may also include additional logic to validate the data and handle any potential errors during the extraction, processing, and storage phases. This robust and reliable solution offers a streamlined approach to managing data from Excel sheets, making it a valuable tool for various applications, such as data migration, data analysis, and maintaining structured databases.

Chapter 7

Testing

7.1 Machine Learning

The test conducted on the four machine learning algorithms - linear regression, MLP regression, SVR regression, and random forest regression - aimed to gauge their predictive performance for college cutoffs. To carry out the evaluation, we collected historical cutoff data spanning the last four years and utilized each algorithm to forecast the cutoff for the upcoming fifth year. Following the predictions, we rigorously compared the projected cutoff values against the actual cutoff values of the fifth year to determine the accuracy of each algorithm.

For accuracy assessment, we adopted the absolute mean error (MAE) method, a widely-used metric for regression tasks. The MAE represents the average absolute difference between the predicted cutoffs and the actual cutoffs for the fifth year across all colleges in the dataset. A lower MAE signifies better predictive performance, as it indicates smaller discrepancies between the predicted and actual values. By employing this evaluation metric, we gained valuable insights into the effectiveness of each algorithm in predicting college cutoffs, enabling us to identify the most reliable algorithm for the college recommendation system. This test played a pivotal role in ensuring the robustness and accuracy of the system, providing students with personalized and dependable college recommendations based on the latest cutoff predictions.

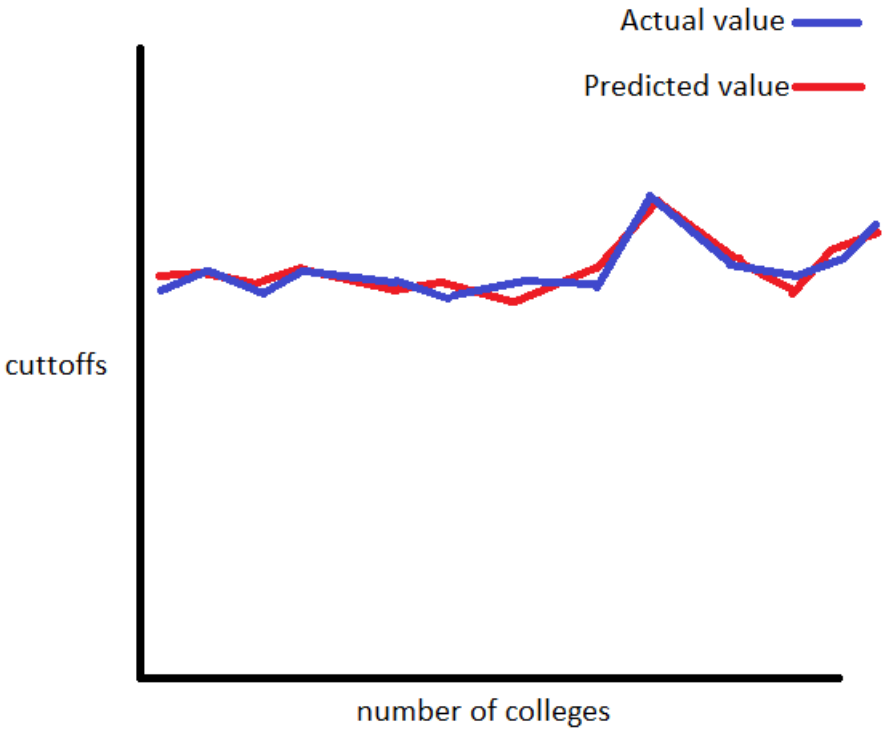

Figure 7.1: Comparison between predicted value and actual value

Chapter 8

Results

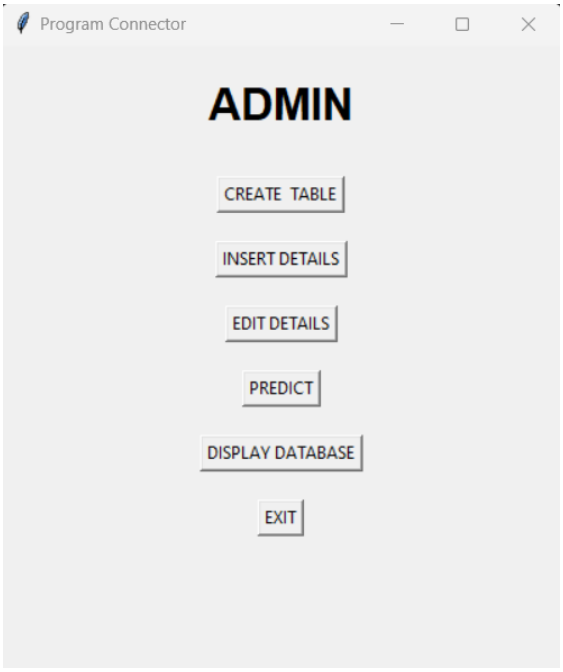

Figure 8.1: Admin GUI

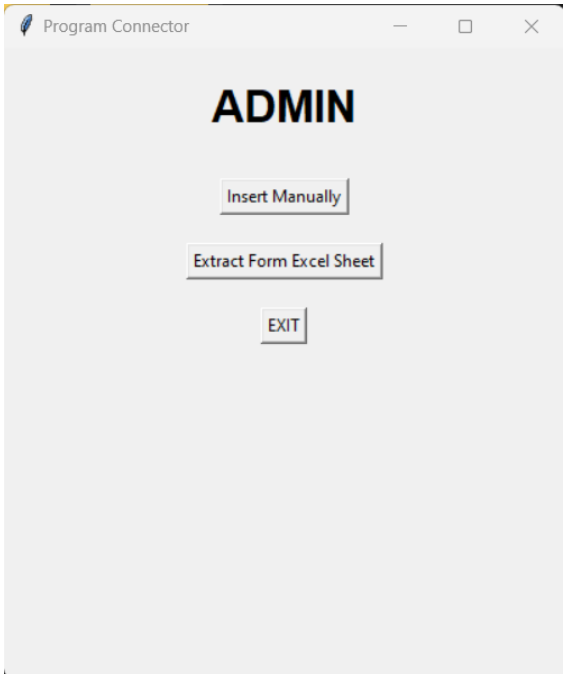

Figure 8.2: Insert GUI

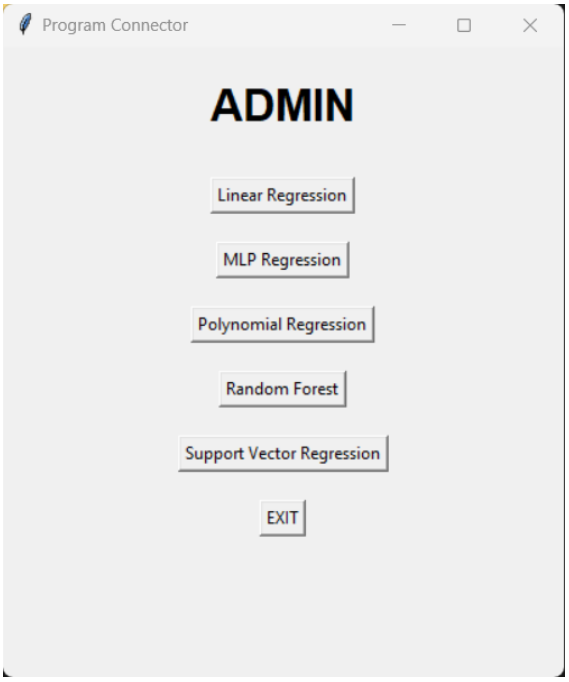

Figure 8.3: Prediction GUI

Figure 8.4: Extraction GUI

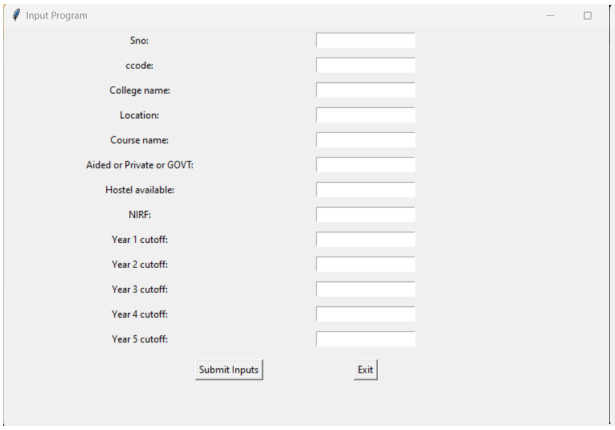

Figure 8.5: Updation

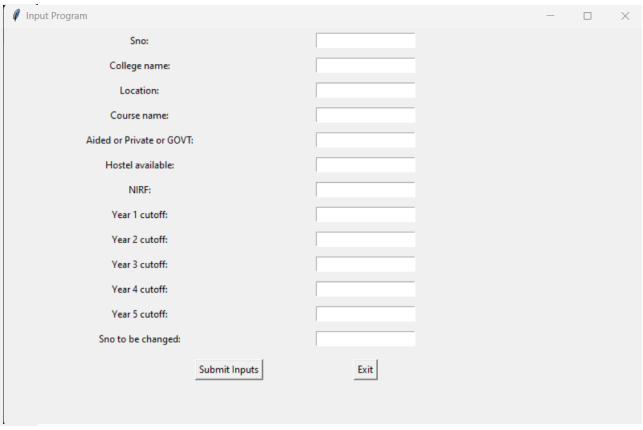

Figure 8.6: Insertion

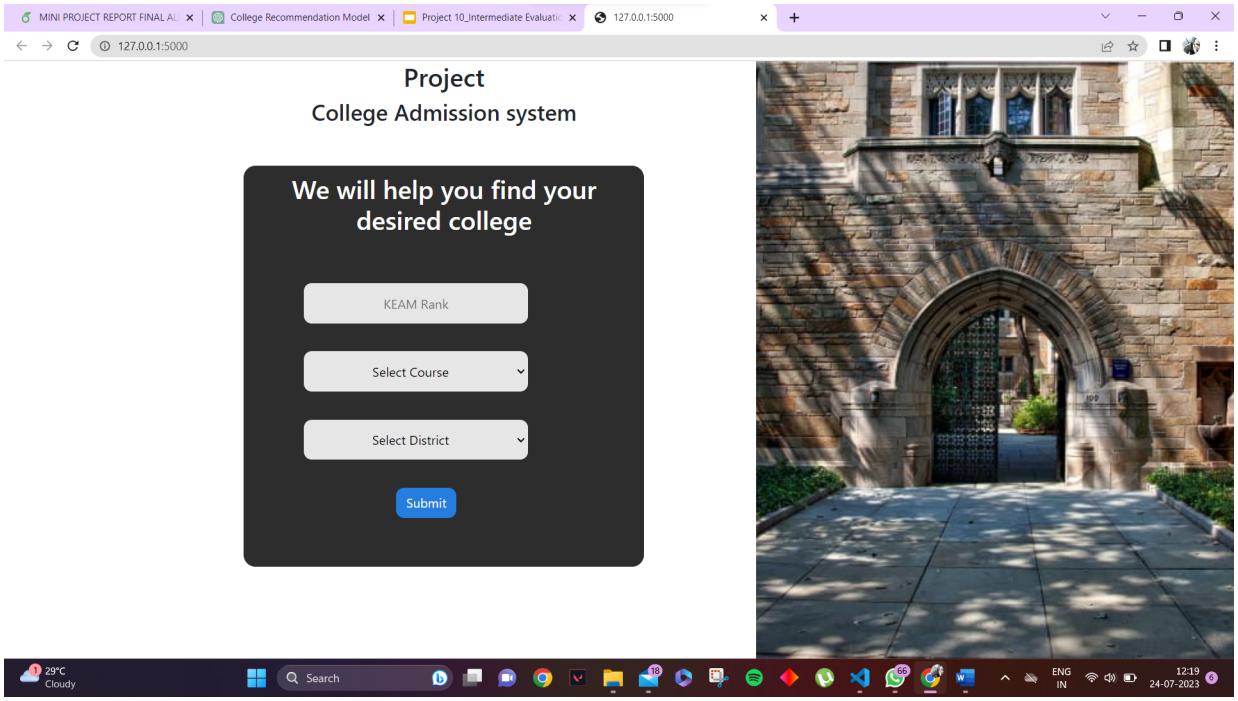

Figure 8.7: Home Page

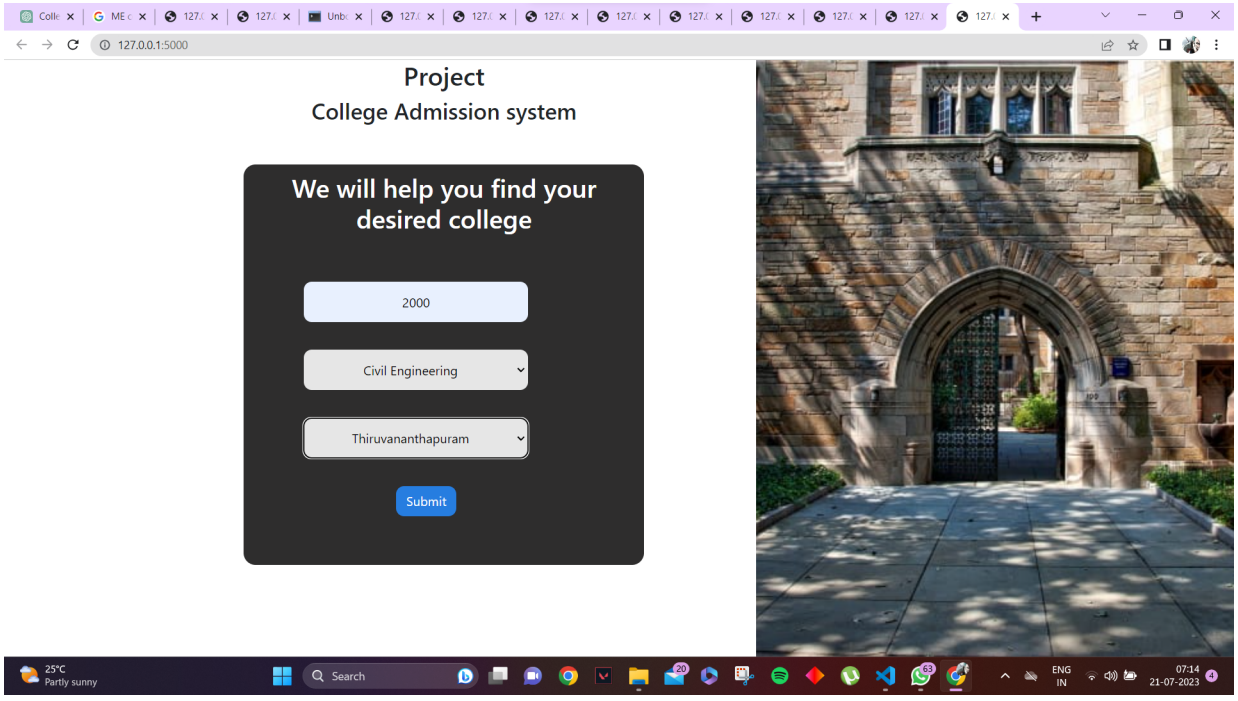

Figure 8.8: Data Insertion

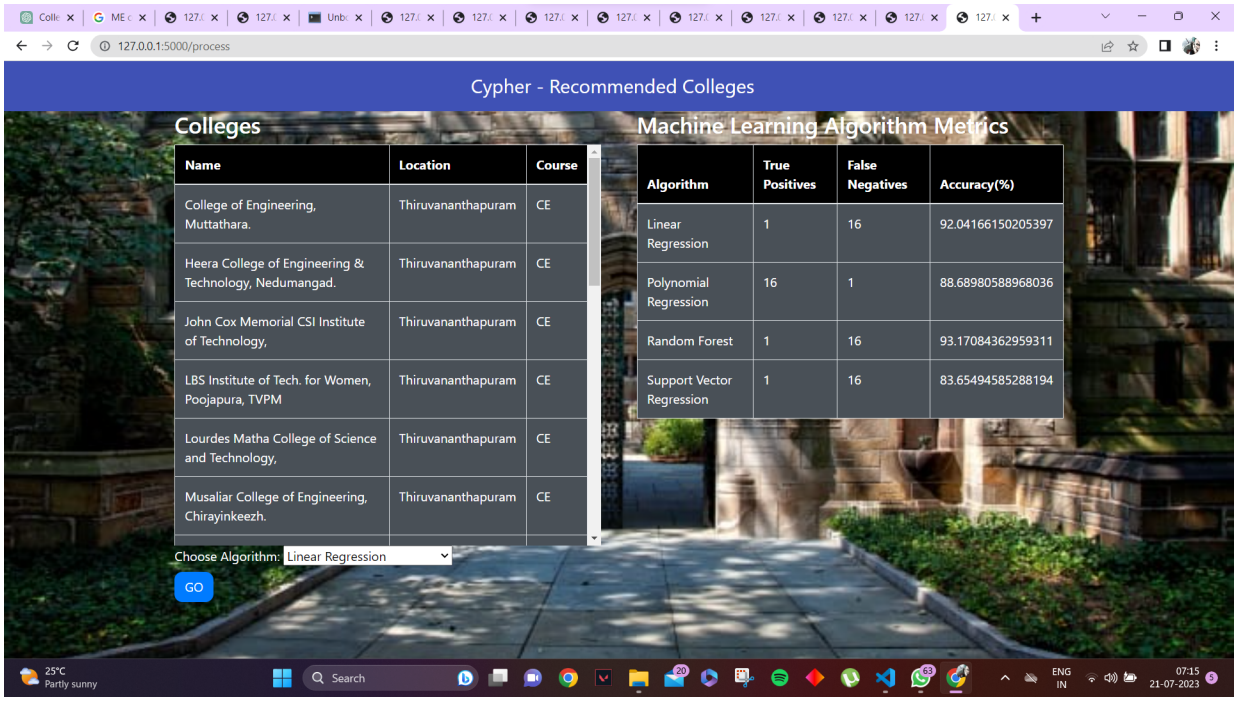

Figure 8.9: Reccomended colleges with accuracy table

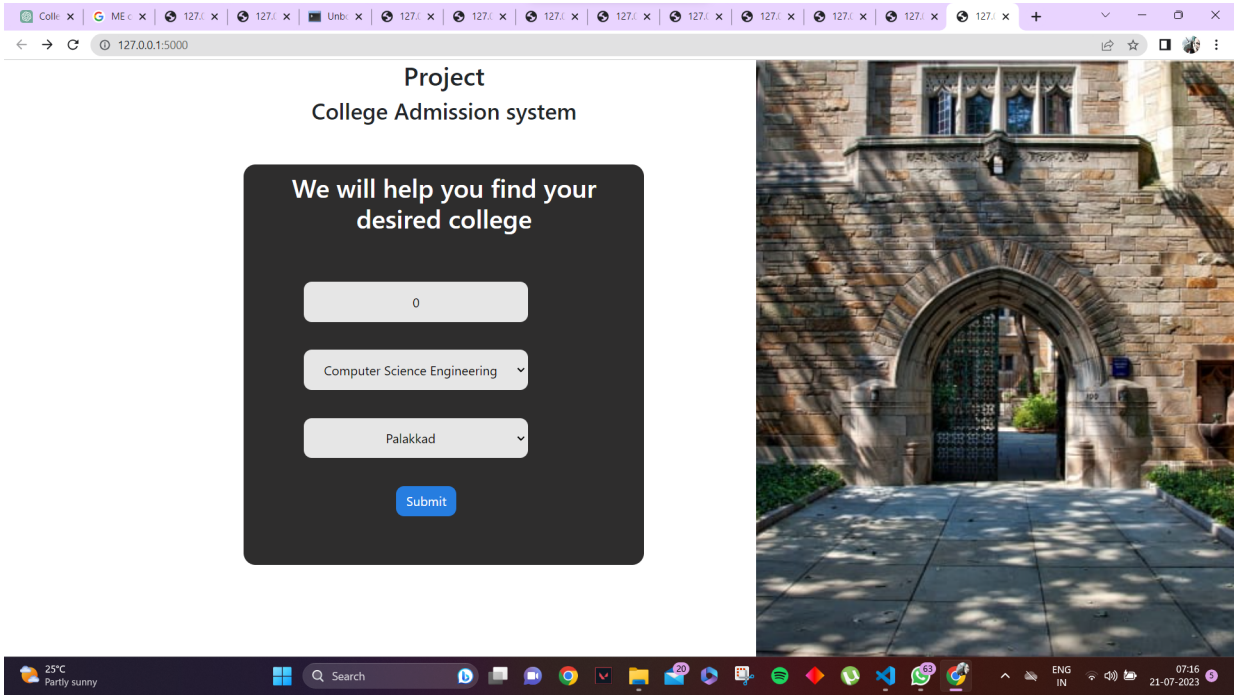

Figure 8.10: Invalid rank insertion

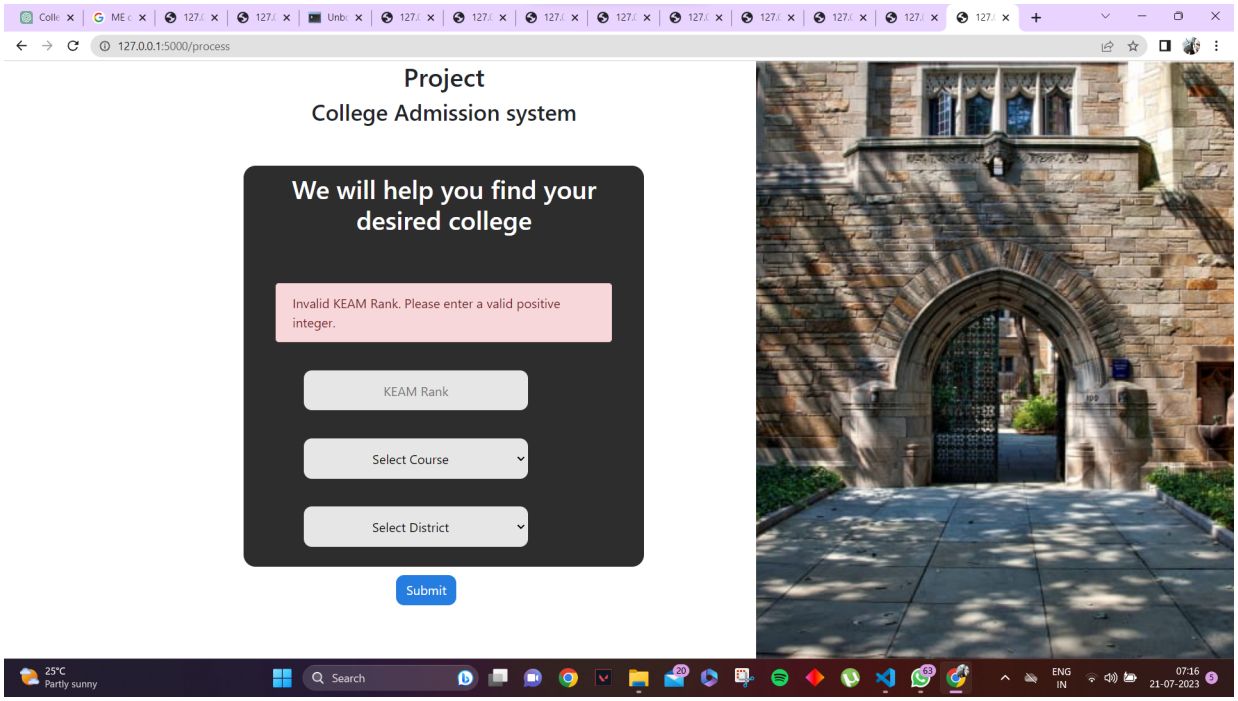

Figure 8.11: Error message

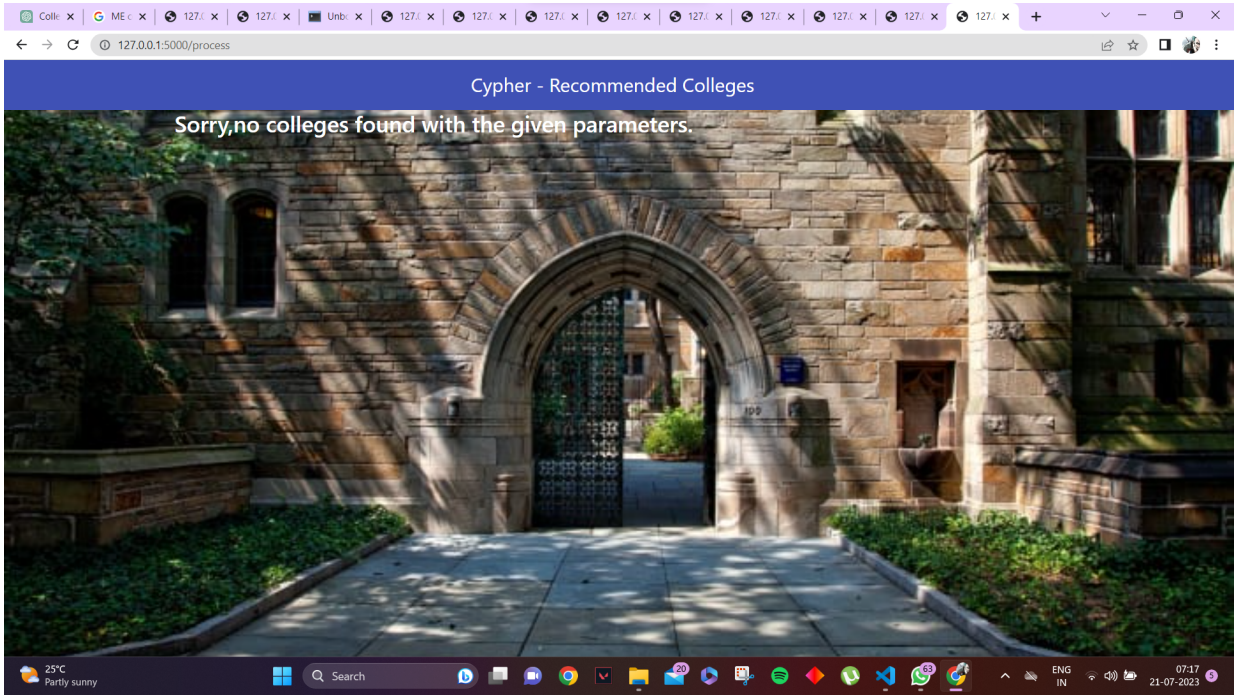

Figure 8.12: No colleges found

Chapter 9

Risks and Challenges

1. The quality and availability of the data can pose a significant challenge. The dataset may have missing or inconsistent information, outdated records, or limited coverage for certain colleges or regions.

2. Colleges and universities frequently update their admission criteria, policies, and programs. We have to keep the system up-to-date.

3. As the user base grows, the system should be able to handle increased traffic and deliver recommendations efficiently.

4. The machine learning algorithm predicts the output with upto 70-80 percent accuracy.

5. Efficiency increases as historical data increases in number.

Chapter 10

Conclusion

Our college recommendation system is a groundbreaking tool that aims to transform the way students navigate the complex landscape of college selection. With the ever-increasing number of colleges and courses available, students often find themselves overwhelmed and confused during this critical decision-making process. Our user-friendly HTML-based interface ensures that students can easily input their rank, course preferences, and desired location. The seamless integration with a Python backend empowers the system to access a vast database of historical data, enabling us to accurately predict the admission cutoffs for the current year. This innovative use of machine learning technology ensures that students receive personalized recommendations tailored to their unique needs and academic goals.

At the core of our system's success is the comprehensive database that houses a wealth of information about colleges, their admission requirements, and past cutoff trends. By offering these personalized college recommendations, we aim to guide students towards institutions that best align with their academic strengths and career aspirations. Additionally, our system provides valuable insights into each recommended college, presenting information about the available courses, faculty, infrastructure, and extracurricular opportunities, empowering students to make well-rounded decisions that encompass both academic and holistic growth.

By providing students with a reliable, data-driven tool, we aim to level the playing field and ensure that every aspiring student has equal opportunities to explore and pursue their dream colleges. The system's ease of use and transparent approach make it accessible to students from various backgrounds and regions. We believe that by simplifying the college selection process and delivering accurate and relevant recommendations, we can help students embark on their educational journey with confidence and enthusiasm, setting them up for a successful and fulfilling future.

References

[1] https://www.geeksforgeeks.org/ml-linear-regression/

[2] https://docs.python.org/3/library/tkinter.html

[3] https://www.shiksha.com/engineering/keam-exam-cutoff

[4] Pedregosa, F., et al. (2011). "Scikit-learn: Machine Learning in Python." Journal of Machine Learning Research, 12, 2825-2830

[5] Breiman, L. (2001). "Random forests." Machine Learning, 45(1), 5-32.

[6] Smola, A. J., and Schölkopf, B. (2004). "A tutorial on support vector regression." Statistics and Computing, 14(3), 199-222.

Appendix A: Sample Code

LINEAR REGRESSION MODEL

```
import numpy as np
from sklearn.linear_model import LinearRegression
import sqlite3

conn = sqlite3.connect('db1.db')
cursor = conn.cursor()

a=0
m=1
x1=1
count1=0

table_name = 'college'
cursor.execute(f"SELECT COUNT(*) FROM {table_name}")
result = cursor.fetchone()
count = result[0]
count1=int(count)


for a in range(count1):
 print(x1)
 cursor.execute("SELECT sno,y1, co1, y2, co2, y3, co3, y4, co4, y5, co5 FROM college where sno =
"+str(x1)+";")
 data = cursor.fetchone()
 sno0,y01, co01, y02, co02, y03, co03, y04, co04, y05, co05 = data
 past_ranks = np.array([[y01,co01], [y02,co02], [y03,co03], [y04,co04]])
 target_year = 2023
 X = past_ranks[:, 0].reshape(-1, 1)
 y = past_ranks[:, 1]
 model = LinearRegression()
 model.fit(X, y)
 predicted_rank = model.predict([[target_year]])
 p=predicted_rank
 q=int(p)
 cursor.execute ("UPDATE college SET lr = "+str(q)+" where sno = "+str(x1)+";")
 conn.commit()
 x1=x1+1
print("LR Prediction Done")
```

EXTRACTION ALGORITHM

```
import csv
import sqlite3

with open('main1.csv', 'r') as file:

    reader = csv.reader(file)


    conn = sqlite3.connect('db1.db')
    cursor = conn.cursor()
    sno1=1
    x1=0
    for row in reader:
        print(x1)
        col1 = row[0]
        col2 = row[1]
        col3 = row[2]
        col4 = row[3]
        col5 = row[4]
        col6 = row[5]
        col7 = row[6]
        #col8 = row[7]
        sql = "INSERT OR REPLACE INTO college (sno, ccode, cname ,loc,coname,
nirf,hos,apg,co1,co2,co3,co4,co5,y1,y2,y3,y4,y5) VALUES ( ?,?,?,?,?,?, ?, ?,?, ?, ?, ?, ?,?,?,?,?,?)"
        cursor.execute(sql, (sno1,col1,col2,col3,col4,col5,col6,col7,0,0,0,0,0,2017,2018,2019,2021,2022))
        sno1 = sno1+1
        conn.commit()
        x1=x1+1


    conn.close()
print("Main extraction done.")


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


import csv
import sqlite3

with open('main1.csv', 'r') as file:

    reader = csv.reader(file)
```

```
    conn = sqlite3.connect('db1.db')
    cursor = conn.cursor()
    sno1=1
    x1=0
    for row in reader:
        print(x1)
        col1 = row[0]
        col2 = row[1]
        col3 = row[2]
        col4 = row[3]
        col5 = row[4]
        col6 = row[5]
        col7 = row[6]
        #col8 = row[7]
        sql = "INSERT OR REPLACE INTO college (sno, ccode, cname ,loc,coname,
nirf,hos,apg,co1,co2,co3,co4,co5,y1,y2,y3,y4,y5) VALUES ( ?,?,?,?,?,?, ?, ?,?, ?, ?, ?, ?,?,?,?,?,?)"
        cursor.execute(sql, (sno1,col1,col2,col3,col4,col5,col6,col7,0,0,0,0,0,2017,2018,2019,2021,2022))
        sno1 = sno1+1
        conn.commit()
        x1=x1+1


    conn.close()
print("Main extraction done.")
```

ADMIN INTERFACE

```
import tkinter as tk
import subprocess
from PIL import Image, ImageTk

def run_program0():
    subprocess.run(["python", "database.py"])

def run_program1():
    subprocess.run(["python", "inputoptionsgui.py"])

def run_program2():
    subprocess.run(["python", "predictgui.py"])

def run_program3():
    subprocess.run(["python", "displaygui.py"])

def run_program4():
    subprocess.run(["python", "updategui.py"])
```

```
def exit_program():
    window.destroy()

window = tk.Tk()
window.title("Program Connector")



screen_width = window.winfo_screenwidth()
screen_height = window.winfo_screenheight()


center_x = screen_width // 2
center_y = screen_height // 2


window_width = 400
window_height = 450
window.geometry(f"{window_width}x{window_height}+{center_x - (window_width // 2)}+{center_y -
(window_height // 2)}")


label = tk.Label(window, text="ADMIN", font=("Arial", 24, "bold"))
label.pack(pady=20)


button1 = tk.Button(window, text="CREATE  TABLE", command=run_program0)
button1.pack(pady=10)

button1 = tk.Button(window, text="INSERT DETAILS", command=run_program1)
button1.pack(pady=10)


button1 = tk.Button(window, text="EDIT DETAILS", command=run_program4)
button1.pack(pady=10)

button2 = tk.Button(window, text="PREDICT", command=run_program2)
button2.pack(pady=10)

button3 = tk.Button(window, text="DISPLAY DATABASE", command=run_program3)
button3.pack(pady=10)

button3 = tk.Button(window, text="EXIT", command=exit_program)
button3.pack(pady=10)

window.mainloop()
```

USER INTERFACE HTML

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
integrity="sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z"
crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-
9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN"
crossorigin="anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js" integrity="sha384-
B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"
crossorigin="anonymous"></script>
    <style>
        body {
            margin: 0px;
        }
        .background {
            background-image: url("https://res-
console.cloudinary.com/ddncpqawo/thumbnails/v1/image/upload/v1687934576/Y29sZ18xX2Q1MXZjeA==/preview");
            background-size: cover;
            background-repeat: no-repeat;
            background-position: center;
            height: 100vh;
            width: 80vh;
        }
        .white-box {
            padding-inline: 100px;
        }
        .heading {
            text-align:center;
            color: white;
            margin-block: 50px;
            padding-block: 10px;
            text-align: center;

        }
        .quest {
            height: 50px;
            width: 280px;
            border: none;
            padding-inline: 20px;
            border-width: 2px;
```

```
            margin: 35px;
            background-color: rgb(230, 230, 230);
            border-radius: 10px;
            display: block;
            align-items: right;
            text-align: center;
        }



        .details-box {
            background-color: rgb(46, 45, 45);
            height: 500px;
            border-radius: 15px;
            width: 500px ;
            margin: 40px;
            padding-left: 40px;
            padding-right: 40px;
            align-items: center;
        }
        .head {
            text-align: center;
        }
        .btn {
            color: white;
            background-color: rgb(38, 125, 224);
            border-radius: 10px;
            margin-left: 150px;
        }

    </style>
    <script>
        function getinputs() {
            var rank = document.getElementById('keam-rank').value;
            var course = document.getElementById('course').value;
            var location = document.getElementById('location').value;
            window.location.href = "/templates/output.html";
        }
    </script>
</head>
<body>
    <div class="d-flex flex-row justify-content-end">
        <div class="white-box">
            <h2 class="head">Project</h2>
            <h3 class="head">College Admission system</h3>
            <div class="details-box">

                <h2 class="heading">
                    We will help you find your desired college
```

```
            </h2>

            <form action="{{ url_for('process') }}" method="POST">

                <input type="text" class="quest" placeholder="KEAM Rank" name="keam-rank" required>
                <select class="quest" name="course" required>
                    <option value="" selected disabled>Select Course</option>
                    <option value="AEI">AEI</option>
                    <option value="AIDS">AIDS</option>
                    <option value="CE">CE</option>
                    <option value="CHE">CHE</option>
                    <option value="CSE">CSE</option>
                    <option value="ECE">ECE</option>
                    <option value="EEE">EEE</option>
                    <option value="IT">IT</option>
                    <option value="ME">ME</option>
                </select>
                <!--<input type="text" class="quest" placeholder="Location" name="location" required>-->
                <select class="quest" name="location" required>
                    <option value="" selected disabled>Select District</option>
                    <option value="All Kerala">All Kerala</option>
                    <option value="Alappuzha">Alappuzha</option>
                    <option value="Ernakulam">Ernakulam</option>
                    <option value="Idukki">Idukki</option>
                    <option value="Kannur">Kannur</option>
                    <option value="Kasaragod">Kasaragod</option>
                    <option value="Kollam">Kollam</option>
                    <option value="Kottayam">Kottayam</option>
                    <option value="Kozhikode">Kozhikode</option>
                    <option value="Malappuram">Malappuram</option>
                    <option value="Palakkad">Palakkad</option>
                    <option value="Pathanamthitta">Pathanamthitta</option>
                    <option value="Thiruvananthapuram">Thiruvananthapuram</option>
                    <option value="Thrissur">Thrissur</option>
                    <option value="Wayanad">Wayanad</option>
                </select>
                <button type="submit" onclick="getinputs()"  class="btn">Submit</button>
            </form>
        </div>
    </div>
    <div class="background">

    </div>
</div>

</body>
</html>
```

PYTHON BACKEND

```
from flask import Flask, render_template, request, session
from sklearn.metrics import mean_absolute_error
import sqlite3

app = Flask(__name__, static_folder='static')
app.secret_key = "babu"

default_algorithm = "lr"


algorithms = {
    "lr": "Linear Regression",
    "pr": "Polynomial Regression",
    "rf": "Random Forest",
    "svr": "Support Vector Regression"
}

@app.route('/')
def home():
    error = request.args.get('error')
    return render_template('input.html', algorithms=algorithms)

@app.route('/process', methods=['POST'])
def process():

    rank = request.form.get('keam-rank', '')
    course = request.form.get('course', '')
    location = request.form.get('location', '')

    try:
        rank = int(rank)
        if rank <= 0:
            raise ValueError
    except ValueError:
        error = "Invalid KEAM Rank. Please enter a valid positive integer."
        return render_template('input.html', algorithms=algorithms, error=error)


    session['rank'] = rank
    session['course'] = course
    session['location'] = location

    conn = sqlite3.connect('db1.db')
    cursor = conn.cursor()
```

```
    cutoff_column = default_algorithm

    if location == 'All Kerala':
        query = f"SELECT cname, loc, coname, apg, hos FROM college WHERE {cutoff_column} >= ? AND
coname = ?"
        cursor.execute(query, (rank, course))
    else:
        query = f"SELECT cname, loc, coname, apg, hos FROM college WHERE {cutoff_column} >= ? AND
coname = ? AND loc = ?"
        cursor.execute(query, (rank, course, location))
    results = cursor.fetchall()

    colleges = []
    for row in results:
        college = {
            'name': row[0],
            'location': row[1],
            'course': row[2],
            'category': row[3],
            'hostel': row[4]
        }
        colleges.append(college)

    colleges_found = bool(colleges)

    conn.close()

    conn = sqlite3.connect('db2.db')
    cursor = conn.cursor()

    metrics = {}
    if colleges_found:
        for algorithm in algorithms:
            if location == 'All Kerala':
                query = f"SELECT {algorithm}, Co5 FROM college WHERE {algorithm} >= ? AND coname = ?"
                cursor.execute(query, (rank, course))
            else:
                query = f"SELECT {algorithm}, Co5 FROM college WHERE {algorithm} >= ? AND coname = ?
AND loc = ?"
                cursor.execute(query, (rank, course, location))
            results = cursor.fetchall()

            predictions = []
            actuals = []
            for row in results:
                predicted_cutoff = row[0]
                actual_cutoff = row[1]
                predictions.append(predicted_cutoff)
```

```
            actuals.append(actual_cutoff)

        mae = mean_absolute_error(actuals, predictions)
        accuracy = 1 - (mae / max(actuals))  # Calculating accuracy as a ratio
        accuracy = accuracy * 100
        true_positive = sum(predicted > actual for predicted, actual in zip(predictions, actuals))
        false_negative = sum(predicted < actual for predicted, actual in zip(predictions, actuals))

        metrics[algorithm] = {
            'accuracy': accuracy,
            'true_positive': true_positive,
            'false_negative': false_negative
        }
    else:
        for algorithm in algorithms:
            metrics[algorithm] = {
                'accuracy': -1,
                'true_positive': -1,
                'false_negative': -1
            }

    conn.close()

    return render_template('output.html', colleges=colleges, colleges_found=colleges_found,
algorithms=algorithms, metrics=metrics, selected_algorithm=default_algorithm)

@app.route('/train', methods=['POST'])
def train():
    global default_algorithm

    algorithm = request.form.get('algorithm', '')

    if algorithm:
        default_algorithm = algorithm

    rank = session.get('rank', '')
    course = session.get('course', '')
    location = session.get('location', '')

    conn = sqlite3.connect('db1.db')
    cursor = conn.cursor()

    cutoff_column = default_algorithm

    if location == 'All Kerala':
        query = f"SELECT cname, loc, coname, apg, hos FROM college WHERE {cutoff_column} >= ? AND
coname = ?"
        cursor.execute(query, (rank, course))
    else:
```

```
        query = f"SELECT cname, loc, coname, apg, hos FROM college WHERE {cutoff_column} >= ? AND coname = ? AND loc = ?"
        cursor.execute(query, (rank, course, location))
    results = cursor.fetchall()

    colleges = []
    for row in results:
        college = {
            'name': row[0],
            'location': row[1],
            'course': row[2],
            'category': row[3],
            'hostel': row[4]
        }
        colleges.append(college)

    colleges_found = bool(colleges)

    conn.close()

    conn = sqlite3.connect('db2.db')
    cursor = conn.cursor()

    metrics = {}
    if colleges_found:
        for algorithm in algorithms:
            if location == 'All Kerala':
                query = f"SELECT {algorithm}, Co5 FROM college WHERE {algorithm} >= ? AND coname = ?"
                cursor.execute(query, (rank, course))
            else:
                query = f"SELECT {algorithm}, Co5 FROM college WHERE {algorithm} >= ? AND coname = ? AND loc = ?"
                cursor.execute(query, (rank, course, location))
            results = cursor.fetchall()

            predictions = []
            actuals = []
            for row in results:
                predicted_cutoff = row[0]
                actual_cutoff = row[1]
                predictions.append(predicted_cutoff)
                actuals.append(actual_cutoff)

            mae = mean_absolute_error(actuals, predictions)
            accuracy = 1 - (mae / max(actuals))
            accuracy = accuracy * 100
            true_positive = sum(predicted > actual for predicted, actual in zip(predictions, actuals))
            false_negative = sum(predicted < actual for predicted, actual in zip(predictions, actuals))
```

```
            metrics[algorithm] = {
                'accuracy': accuracy,
                'true_positive': true_positive,
                'false_negative': false_negative
            }
    else:
        for algorithm in algorithms:
            metrics[algorithm] = {
                'accuracy': -1,
                'true_positive': -1,
                'false_negative': -1
            }



    conn.close()


    return render_template('output.html', colleges=colleges, colleges_found=colleges_found,
algorithms=algorithms, metrics=metrics, selected_algorithm=default_algorithm)

if __name__ == '__main__':
    app.run(debug=True)
```

Appendix B: CO-PO And CO-PSO Mapping

COURSE OUTCOMES:

After completion of the course the student will be able to

SL. NO	DESCRIPTION	Blooms' Taxonomy Level
CO1	Identify technically and economically feasible problems (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO2	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO3	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions of minimal complexity by using modern tools & advanced programming techniques (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO4	Prepare technical report and deliver presentation (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO5	Apply engineering and management principles to achieve the goal of the project (Cognitive Knowledge Level: Apply)	Level 3: Apply

CO-PO AND CO-PSO MAPPING

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO3
CO1	3	3	3	3		2	2	3	2	2	2	3	2	2	2
CO2	3	3	3	3	3	2		3	2	3	2	3	2	2	2
CO3	3	3	3	3	3	2	2	3	2	2	2	3			2
CO4	2	3	2	2	2			3	3	3	2	3	2	2	2
CO5	3	3	3	2	2	2	2	3	2		2	3	2	2	2

3/2/1: high/medium/low

JUSTIFICATIONS FOR CO-PO MAPPING

MAPPING	LOW/ MEDIUM/ HIGH	JUSTIFICATION
100003/CS622T.1-PO1	**HIGH**	Identify technically and economically feasible problems by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS622T.1-PO2	**HIGH**	Identify technically and economically feasible problems by analysing complex engineering problems reaching substantiated conclusions using first principles of mathematics.
100003/CS622T.1-PO3	**HIGH**	Design solutions for complex engineering problems by identifying technically and economically feasible problems.
100003/CS622T.1-PO4	**HIGH**	Identify technically and economically feasible problems by analysis and interpretation of data.
100003/CS622T.1-PO6	**MEDIUM**	Responsibilities relevant to the professional engineering practice by identifying the problem.
100003/CS622T.1-PO7	**MEDIUM**	Identify technically and economically feasible problems by understanding the impact of the professional engineering solutions.
100003/CS622T.1-PO8	**HIGH**	Apply ethical principles and commit to professional ethics to identify technically and economically feasible problems.
100003/CS622T.1-PO9	**MEDIUM**	Identify technically and economically feasible problems by working as a team.
100003/CS622T.1-PO10	**MEDIUM**	Communicate effectively with the engineering community by identifying technically and economically feasible problems.
100003/CS622T.1-P011	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles by selecting the technically and economically feasible problems.
100003/CS622T.1-PO12	**HIGH**	Identify technically and economically feasible problems for long term learning.
100003/CS622T.1-PSO1	**MEDIUM**	Ability to identify, analyze and design solutions to identify technically and economically feasible problems.
100003/CS622T.1-PSO2	**MEDIUM**	By designing algorithms and applying standard practices in software project development and Identifying technically and economically feasible problems.
100003/CS622T.1-PSO3	**MEDIUM**	Fundamentals of computer science in competitive research can be applied to Identify technically and economically feasible problems.
100003/CS622T.2-PO1	**HIGH**	Identify and survey the relevant by applying the knowledge of mathematics, science, engineering fundamentals.

100003/CS622T.2-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems get familiarized with software development processes.
100003/CS622T.2-PO3	**HIGH**	Design solutions for complex engineering problems and design based on the relevant literature.
100003/CS622T.2-PO4	**HIGH**	Use research-based knowledge including design of experiments based on relevant literature.
100003/CS622T.2-PO5	**HIGH**	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes by using modern tools.
100003/CS622T.2-PO6	**MEDIUM**	Create, select, and apply appropriate techniques, resources, by identifying and surveying the relevant literature.
100003/CS622T.2-PO8	**HIGH**	Apply ethical principles and commit to professional ethics based on the relevant literature.
100003/CS622T.2-PO9	**MEDIUM**	Identify and survey the relevant literature as a team.
100003/CS622T.2-PO10	**HIGH**	Identify and survey the relevant literature for a good communication to the engineering fraternity.
100003/CS622T.2-PO11	**MEDIUM**	Identify and survey the relevant literature to demonstrate knowledge and understanding of engineering and management principles.
100003/CS622T.2-PO12	**HIGH**	Identify and survey the relevant literature for independent and lifelong learning.
100003/CS622T.2-PSO1	**MEDIUM**	Design solutions for complex engineering problems by Identifying and survey the relevant literature.
100003/CS622T.2-PSO2	**MEDIUM**	Identify and survey the relevant literature for acquiring programming efficiency by designing algorithms and applying standard practices.
100003/CS622T.2-PSO3	**MEDIUM**	Identify and survey the relevant literature to apply the fundamentals of computer science in competitive research.
100003/CS622T.3-PO1	**HIGH**	Perform requirement analysis, identify design methodologies by using modern tools & advanced programming techniques and by applying the knowledge of mathematics, science, engineering fundamentals.
100003/CS622T.3-PO2	**HIGH**	Identify, formulate, review research literature for requirement analysis, identify design methodologies and develop adaptable & reusable solutions.

100003/CS622T.3-PO3	**HIGH**	Design solutions for complex engineering problems and perform requirement analysis, identify design methodologies.
100003/CS622T.3-PO4	**HIGH**	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
100003/CS622T.3-PO5	**HIGH**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools.
100003/CS622T.3-PO6	**MEDIUM**	Perform requirement analysis, identify design methodologies and assess societal, health, safety, legal, and cultural issues.
100003/CS622T.3-PO7	**MEDIUM**	Understand the impact of the professional engineering solutions in societal and environmental contexts and Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions.
100003/CS622T.3-PO8	**HIGH**	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions by applying ethical principles and commit to professional ethics.
100003/CS622T.3-PO9	**MEDIUM**	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
100003/CS622T.3-PO10	**MEDIUM**	Communicate effectively with the engineering community and with society at large to perform requirement analysis, identify design methodologies.
100003/CS622T.3-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering requirement analysis by identifying design methodologies.
100003/CS622T.3-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by analysis, identify design methodologies and develop adaptable & reusable solutions.
100003/CS622T.3-PSO3	**MEDIUM**	The ability to apply the fundamentals of computer science in competitive research and prior to that perform requirement analysis, identify design methodologies.
100003/CS622T.4-PO1	**MEDIUM**	Prepare technical report and deliver presentation by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS622T.4-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems by preparing technical report and deliver presentation.

100003/CS622T.4-PO3	**MEDIUM**	Prepare Design solutions for complex engineering problems and create technical report and deliver presentation.
100003/CS622T.4-PO4	**MEDIUM**	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions and prepare technical report and deliver presentation.
100003/CS622T.4-PO5	**MEDIUM**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and Prepare technical report and deliver presentation.
100003/CS622T.4-PO8	**HIGH**	Prepare technical report and deliver presentation by applying ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
100003/CS622T.4-PO9	**HIGH**	Prepare technical report and deliver presentation effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
100003/CS622T.4-PO10	**HIGH**	Communicate effectively with the engineering community and with society at large by prepare technical report and deliver presentation.
100003/CS622T.4-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work by prepare technical report and deliver presentation.
100003/CS622T.4-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by prepare technical report and deliver presentation.
100003/CS622T.4-PSO1	**MEDIUM**	Prepare a technical report and deliver presentation to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas.
100003/CS622T.4-PSO2	**MEDIUM**	To acquire programming efficiency by designing algorithms and applying standard practices in software project development and to prepare technical report and deliver presentation.
100003/CS622T.4-PSO3	**MEDIUM**	To apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs by preparing technical report and deliver presentation.
100003/CS622T.5-PO1	**HIGH**	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
100003/CS622T.5-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems by applying engineering and management principles to achieve the goal of the project.

100003/CS622T.5-PO3	**HIGH**	Apply engineering and management principles to achieve the goal of the project and to design solutions for complex engineering problems and design system components or processes that meet the specified needs.
100003/CS622T.5-PO4	**MEDIUM**	Apply engineering and management principles to achieve the goal of the project and use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
100003/CS622T.5-PO5	**MEDIUM**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO6	**MEDIUM**	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities by applying engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO7	**MEDIUM**	Understand the impact of the professional engineering solutions in societal and environmental contexts, and apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO8	**HIGH**	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice and to use the engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO9	**MEDIUM**	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PSO1	**MEDIUM**	The ability to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas. Apply engineering and management principles to achieve the goal of the project.

100003/CS622T.5-PSO2	**MEDIUM**	The ability to acquire programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products meeting the demands of the industry and to apply engineering and management principles to achieve the goal of the project.
100003/CS622T.5-PSO3	**MEDIUM**	The ability to apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur and apply engineering and management principles to achieve the goal of the project.