Chapter-I

**INTRODUCTION**

**1.1 OBJECTIVE:**

Today, all higher education institutions, especially science and engineering colleges, are facing challenges in the admissions process. Each university should strive for an admissions system based on valid and reliable admissions criteria that select candidates likely to succeed in its programs. In addition, each university should use the best possible techniques for predicting applicant’s future academic performance before admitting them. This would support university decision makers as they set efficient admissions criteria.

However, most higher education institutions face challenges when they analyze their large educational databases to predict student’s performance. This is because they use only conventional statistical methods rather than new and efficient predictive techniques such as Educational Data Mining (EDM), which is the most popular technique to evaluate and predict student’s performance.

The primary objective of this project is to leverage data mining techniques to develop predictive models for student performance in the context of university admission systems. The project aims to address the following key objectives:

* **Predictive Modeling:**

Create accurate and robust predictive models using data mining algorithms that analyze historical data on student’s academic records, standardized test scores, extracurricular activities, and other relevant factors. These models will forecast the academic performance of prospective students during their university education.

* **Feature Engineering:**

Identify and select the most influential features and attributes that significantly contribute to the prediction of student performance. This involves preprocessing and transforming raw data to enhance the quality and effectiveness of the predictive models.

* **Early Identification:**

Develop predictive models that can identify students at risk of underperforming or dropping out early in their academic journey. Early intervention can improve retention rates and student success.

* 1. **PROBLEM STATEMENT:**

University admission systems play a pivotal role in selecting candidates who are not only academically qualified but also likely to thrive in their chosen programs. However, the traditional admission process often relies on standardized test scores and academic transcripts, which may not fully capture a student's potential or readiness for a particular program. The problem at hand is the need for a more data-driven and predictive approach to university admissions that goes beyond conventional criteria.

This project aims to address this challenge by leveraging data mining techniques to predict student performance and suitability for specific programs. The problem lies in developing a predictive model that can analyze a comprehensive range of data points, including academic records, extracurricular activities, letters of recommendation, and personal statements, to assess a student's potential to succeed in a particular academic program. The model should not only provide accurate predictions but also be interpretable, fair, and adaptable to the dynamic nature of student data.

* 1. **EXISTING SYSTEM:**

In today's data-driven educational landscape, universities are increasingly turning to data mining techniques to predict student performance and enhance decision-making processes in their admission systems. By leveraging advanced algorithms and analytics, these systems analyze vast amounts of historical data on student demographics, academic records, standardized test scores, extracurricular activities, and other relevant factors. Through this analysis, universities can identify patterns and trends that correlate with student success and failure, allowing them to make more informed decisions when evaluating prospective applicants.

By predicting student performance, these systems assist admission committees in selecting candidates who are not only academically qualified but also likely to thrive within the university's academic environment. Moreover, such predictive models enable universities to allocate resources more efficiently, personalize support services, and ultimately enhance the overall educational experience for both students and faculty. As technology continues to evolve, these data mining techniques are poised to play an increasingly integral role in shaping the future of university admission systems.

Existing systems use methods like Decision Trees, Support Vector Machines, and Naïve Bayes to classify the applicant into two categories “accepted” and “rejected”. Each of these techniques offers unique advantages that contribute to the effectiveness of the admission decision-making process. Decision Trees excel in their ability to represent and visualize decision-making processes, making them intuitive for admission committees to interpret and understand. Support Vector Machines, on the other hand, are adept at handling high-dimensional data and can effectively delineate complex decision boundaries between accepted and rejected applicants. Naïve Bayes classifiers leverage probabilistic reasoning to assess the likelihood of an applicant being accepted based on their attributes, making them particularly suitable for handling large datasets with categorical variables.

**1.3.1 DRAWBACKS:**

* **Data Quality Issues**: One of the primary challenges is the quality of the data itself. The data used for predictive analytics might be incomplete, outdated, or inaccurate, leading to biased predictions and flawed decision-making.
* **Bias and Fairness Concerns**: Data mining models can inherit biases present in historical data, leading to unfair treatment of certain groups of students, such as those from underrepresented backgrounds or minority groups. This can perpetuate existing inequalities in education and access to opportunities.
* **Overfitting**: Data mining models might be overfitted to the training data, meaning they perform well on the data they were trained on but fail to generalize to new data. This can lead to unreliable predictions and inaccurate decision-making.
* **Lack of Contextual Understanding**: Data mining techniques often rely solely on quantitative data, overlooking important qualitative factors that could influence student performance, such as socio-economic background, personal circumstances, or non-academic achievements.
* **Ethical Concerns**: There are ethical implications associated with using predictive analytics in university admissions, such as privacy concerns and the potential for discrimination. Students may feel uncomfortable with their personal data being used to make high-stakes decisions about their future.
* **Dynamic Nature of Education**: Student performance can be influenced by various dynamic factors, such as changes in teaching methodologies, curriculum, or external factors like socio-political events. Data mining models may struggle to adapt to these changes effectively.
* **Incompleteness of Data**: Not all relevant data about a student may be available or accessible for analysis. This can limit the effectiveness of data mining techniques in accurately predicting student performance and making informed decisions.
  1. **PROPOSED SYSTEM:**

Proposed system uses Random Forest Classifier and Artificial Neural Network (ANN) algorithms for leveraging data mining techniques to predict student performance, thereby enhancing decision-making processes in university admission systems.

Random Forest Classifier operates by constructing multiple decision trees during training and outputting the mode of the classes as the prediction, providing robustness against overfitting and handling large datasets efficiently.

On the other hand, ANN, inspired by the human brain's neural structure, employs interconnected layers of nodes to process complex patterns in data. Its ability to adapt and learn from input-output relationships makes it adept at capturing intricate relationships within student performance data. By harnessing these algorithms, admission systems can analyze diverse student attributes, such as academic records, extracurricular activities, and demographic information, to forecast performance accurately.

Such predictive insights empower universities to make informed decisions regarding admissions, aiding in selecting candidates who are likely to excel academically and contribute positively to the university community. Furthermore, these systems can facilitate personalized interventions and support mechanisms for students at risk of underperformance, fostering a conducive learning environment and bolstering overall educational outcomes.

**1.4.1 ADVANTAGES:**

* **High Accuracy**: RFC and ANN are known for their ability to handle complex datasets and make accurate predictions. By analyzing historical student performance data, these algorithms can identify patterns and correlations that might not be immediately apparent to human evaluators, resulting in more accurate predictions of future student performance.
* **Scalability**: These algorithms are highly scalable, meaning they can effectively handle large volumes of data without sacrificing performance. This scalability is crucial for university admission systems, which often deal with thousands of applicants each year.
* **Flexibility**: Both RFC and ANN are flexible algorithms that can handle various types of data, including numerical, categorical, and textual data. This flexibility allows admission systems to incorporate a wide range of factors into their decision-making process, such as academic records, standardized test scores, extracurricular activities, and personal statements.
* **Feature Importance Analysis**: RFC can provide insights into the importance of different features in predicting student performance. By understanding which factors are most influential, universities can tailor their admission criteria to prioritize the attributes that are most predictive of success.
* **Non-linearity Handling**: ANN excels at capturing non-linear relationships between input features and output labels. This is particularly beneficial in admission systems where the relationship between various factors and student performance may not be straightforward.
* **Handling Missing Data**: Both RFC and ANN have mechanisms to handle missing data effectively. In admission systems, where not all applicants may provide complete information, this capability ensures that predictions can still be made even with incomplete datasets.
* **Adaptability**: These algorithms can be trained and retrained with new data periodically, allowing admission systems to adapt to changing trends and preferences over time. This adaptability ensures that the predictive models remain relevant and effective in supporting decision-making processes.

Chapter-II

**SYSTEM ANALYSIS**

* 1. **LITERATURE SURVEY:**

**2.1.1 REFERENCE PAPERS:**

* **TITLE: A New Student Performance Analyzing System Using Knowledge Discovery in Higher Educational Databases by H. Guruler, A. Istanbullu, and M. Karahasan.**

This paper states that- Knowledge discovery is a wide ranged process including data mining, which is used to find out meaningful and useful patterns in large amounts of data. In order to explore the factors having impact on the success of university students, knowledge discovery software, called MUSKUP, has been developed and tested on student data. In this system a decision tree classification is employed as a data mining technique. With this software system all this approach is to have access to all the functionalities of SQL server and Analysis Services through single software. The study was carried out on the data from university students. According to results of the study, the types of registration to the university and the income levels of the student’s family were found to be associated with student success.

This paper concludes that- The development and testing of the MUSKUP knowledge discovery software, utilizing decision tree classification as a data mining technique, have provided valuable insights into factors impacting university student success. By employing this system, researchers were able to analyze student data effectively and identify significant correlations. Specifically, the study revealed that the types of registration to the university and the income levels of student’s families are closely linked to student success. This underscores the importance of considering socio-economic factors alongside educational practices in supporting student achievement. Overall, the findings highlight the potential of data mining techniques facilitated by software like MUSKUP to inform strategies for enhancing educational outcomes.

* **TITLE: Educational Data Mining: A Review by S. K. Mohamad and Z. Tasir.**

This paper states that- Data Mining is very useful in the field of education especially when examining students learning behavior in online learning environment. This is due to the potential of data mining in analyzing and uncovering the hidden information of the data itself which is hard and very time consuming if to be done manually. The purpose of this review is to look into how the data mining was tackled by previous scholars and the latest trends on data mining in educational research. Several limitations of existing research are discussed and some directions for future research are suggested.

This paper concludes that- Data mining emerges as a powerful tool for delving into the intricacies of student learning behavior within online learning environments. Its capability to unveil hidden insights within data streamlines analysis processes that would otherwise be laborious and time-intensive if done manually. This review comprehensively examines how previous scholars have leveraged data mining in educational research, while also highlighting the latest trends in this field. Despite the advancements, several limitations in existing research are identified, prompting suggestions for future directions. By synthesizing past approaches and addressing current challenges, this review paves the way for more nuanced and effective applications of data mining in educational contexts. Looking ahead, the future of data mining in educational research holds promising opportunities. Emerging trends such as deep learning, natural language processing, and sentiment analysis offer new avenues for exploring educational data in more depth. Additionally, interdisciplinary collaborations between data scientists, educators, and psychologists can lead to innovative approaches for leveraging data mining techniques to enhance learning outcomes.

* **TITLE: A Comparative Analysis On The Evaluation Of Classification Algorithms In The Prediction Of Student’s Performance by C. Anuradha and T. Velmurugan.**

This paper states that- Data mining techniques are implemented in many organizations as a standard procedure for analyzing the large volume of available data, extracting useful information and knowledge to support the major decision-making processes. Educational data mining is rapidly developing as a key technique in the analysis of data generated in the educational domain. The primary objective of this research work is to apply the classification techniques to the prediction of the performance of students in end semester university examinations. Particularly, the decision tree algorithm C4.5 (J48), Bayesian classifiers, k Nearest Neighbor algorithm and two rule learner’s algorithms namely OneR and JRip are used for classifying the performance of students as well as to develop a model of student performance predictors. The result of this study reveals that overall accuracy of the tested classifiers is above 60%. In addition, classification accuracy for the different classes reveals that the predictions are worst for distinction class and fairly good for the first class. The study can be extended to draw the performance of other classification techniques on an expanded data set with more distinct attributes to get more accurate results.

This paper concludes that- The application of classification techniques, including decision trees, Bayesian classifiers, k Nearest Neighbor algorithm, and rule learners, in predicting students' performance in end semester university examinations demonstrates promising results in educational data mining. The study reveals an overall accuracy of above 60% for the tested classifiers, indicating their effectiveness in predicting student outcomes. While the classification accuracy varies across different performance categories, with lower accuracy observed for the distinction class and higher accuracy for the first class, the study highlights the potential of these techniques to provide valuable insights for educational institutions. These insights can inform decision-making processes, such as identifying at-risk students who may require additional support or interventions to improve their performance. By continuing to refine and extend these methodologies, educational data mining can play an increasingly vital role in supporting educators and administrators in their efforts to optimize student learning outcomes and educational experiences.

**2.1.2 DATA MINING:**

Data mining is defined as a process used to extract usable data from a larger set of any raw data. It implies analyzing data patterns in large batches of data. This helps businesses be closer to their objective and make better decisions. Data mining is also known as Knowledge Discovery in Data (KDD).

It is a multi-disciplinary skill that uses machine learning, statistics, AI and database technology. The insights derived via Data Mining can be used for marketing, fraud detection, and scientific discovery, etc. Data mining is also called as Knowledge discovery, Knowledge extraction, data/pattern analysis, information harvesting, etc.

Data mining is the process of searching and analyzing a large batch of raw data in order to identify patterns and extract useful information. It is the process of discovering patterns, trends, and valuable insights from large datasets. It involves using various techniques, including statistical analysis, machine learning, and artificial intelligence, to extract meaningful information from data.

**2.1.2.1 Data Mining Techniques:**

* **Classification:** This analysis is used to retrieve important and relevant information about data, and metadata. This data mining method helps to classify data in different classes.
* **Clustering**: Clustering analysis is a data mining technique to identify data that are like each other. This process helps to understand the differences and similarities between the data.
* **Regression**: Regression analysis is the data mining method of identifying and analyzing the relationship between variables. It is used to identify the likelihood of a specific variable, given the presence of other variables.
* **Association Rules:** This data mining technique helps to find the association between two or more Items. It discovers a hidden pattern in the data set.
* **Outer detection:** This type of data mining technique refers to observation of data items in the dataset which do not match an expected pattern or expected behavior. This technique can be used in a variety of domains, such as intrusion, detection, fraud or fault detection, etc. Outer detection is also called Outlier Analysis or Outlier mining.
* **Sequential Patterns:** This data mining technique helps to discover or identify similar patterns or trends in transaction data for certain period.

Each of these techniques has its strengths and is applicable to different types of data mining tasks. The choice of technique depends on the specific problem which trying to solve and the nature of dataset.

* + - 1. **Overview of Data Mining:**
* **Data Collection:** The first step in data mining is collecting relevant data. This data can come from various sources, including databases, spreadsheets, text documents, sensor data, social media, and more.
* **Data Preprocessing:** Raw data often needs to be cleaned and preprocessed. This involves handling missing values, removing duplicates, and transforming data into a suitable format for analysis.
* **Data Integration:** In some cases, data from multiple sources need to be integrated into a single dataset for comprehensive analysis.
* **Data Transformation:** Data might be transformed to make it more suitable for mining. This can include normalization, discretization, or feature engineering to create new attributes.
* **Data Reduction:** Large datasets can be reduced in size to speed up the mining process while still preserving essential information. Techniques like sampling, dimensionality reduction, and aggregation can be used.
* **Model Evaluation:** After applying data mining algorithms, models need to be evaluated for their accuracy and reliability. This may involve techniques such as cross-validation.
* **Visualization:** Data mining results are often visualized to make them more understandable. Graphs, charts, and reports can help convey insights to stakeholders.
* **Interpretation and Application:** The patterns and insights discovered through data mining are interpreted and used for decision-making. This can have applications in various fields, such as business, healthcare, finance, and more.
* **Deployment:** Once a data mining model proves effective, it can be deployed for real-world use. This could involve integrating it into business processes or software applications.
* **Ongoing Monitoring:** Data mining models may need to be continuously monitored and updated to adapt to changing data patterns.
  + - 1. **Advantages of Data Mining:**
* Data-Driven Decision Making
* Increased Efficiency and Productivity:
* Real-time Insights and Actionability
* Cost Reduction and Resource Optimization
  + - 1. **Applications of Data Mining:**

Data mining, the process of discovering patterns, trends, and insights from large datasets, finds applications across numerous domains. Here are some common applications of data mining:

* Market Basket Analysis
* Customer Segmentation
* Fraud Detection
* Predictive Maintenance
* Healthcare Analytics
* Social Media Analysis
* Recommendation Systems

**2.1.3 MACHINE LEARNING:**

Machine learning is a subset of artificial intelligence (AI) that focuses on developing algorithms and models that enable computers to learn from and make predictions or decisions based on data. Instead of being explicitly programmed to perform a specific task, a machine learning system uses data to improve its performance over time.

**2.1.3.1 Methods and Techniques:**

* **Supervised Learning:** In supervised learning, the algorithm is trained on a labeled dataset, where each input is paired with the correct output. Common algorithms include:
* Regression: Predicting a continuous value, like predicting house prices.
* Classification: Predicting a discrete label, like spam detection in emails.
* **Unsupervised Learning:** Here, the algorithm works with unlabeled data to discover patterns or structures. Techniques include:
* Clustering: Grouping similar data points together, like customer segmentation.
* Dimensionality Reduction: Reducing the number of features while preservingimportant information, like principal component analysis (PCA).
* **Semi-Supervised Learning:** This combines elements of supervised and unsupervised learning, useful when labeled data is scarce but unlabeled data is plentiful.
* **Reinforcement Learning:** This involves training an agent to make decisions within an environment to maximize cumulative rewards. It's commonly used in gaming, robotics, and optimization problems.
* **Neural Networks:** A neural network is a computational model inspired by the structure and function of the human brain. Techniques include:
* Convolutional Neural Networks (CNNs): For image processing tasks.
* Recurrent Neural Networks (RNNs): Handling sequential data, like time series or natural language.

**2.1.3.2 Need for Machine Learning:**

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven’t surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. The data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that humans can’t do without human intelligence, but other aspect is that everyone need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

* + - 1. **Advantages of Machine Learning:**
* Easily identifies trends and patterns
* No human intervention needed (automation)
* Handling multi-dimensional and multi-variety data

**2.1.3.4 Applications of Machine Learning:**

* Image and Speech Recognition
* Natural Language Processing (NLP)
* Recommendation Systems
* Healthcare
* Finance
* Manufacturing
* Gaming
* Agriculture
* Energy Management
* Social media
* Customer Service
* Anomaly Detection
* Human Resources
* Environmental Monitoring

**2.1.4 DECISION TREE:**

A Decision Tree is a popular and interpretable machine learning algorithm used for both classification and regression tasks. It's a tree-like structure where each internal node represents a decision, each branch represents an outcome of the decision, and each leaf node represents a class label (in classification) or a continuous value (in regression).

**2.1.5 SVM:**

Support Vector Machines (SVM) are a type of machine learning algorithm used for classification and regression tasks. SVMs work by finding a hyperplane in a high-dimensional space that best separates different classes of data points. This hyperplane is determined by maximizing the margin between the classes, and data points closest to the hyperplane are called support vectors.

**2.1.6 NAÏVE BAYES:**

Naive Bayes is a simple probabilistic technique based on applying Bayes theorem with independent assumptions between variables. It assigns probabilities to each object for each possible class. In this study, this technique is used due to its simplicity, very good performance for real-world problems, computational efficiency.

**2.1.7 RANDOM FOREST:**

The Random Forest Classifier is a powerful machine learning algorithm commonly used for classification tasks. It belongs to the ensemble learning family, which combines the predictions of multiple individual models to produce a more accurate and robust final prediction. Here's how it works:

* **Ensemble of Decision Trees**: The Random Forest algorithm creates a collection of decision trees during the training phase. Each decision tree is trained on a random subset of the training data and uses a random subset of features for making decisions at each node.
* **Bootstrap Aggregating (Bagging)**: Random Forest employs a technique known as bagging, where each decision tree is trained on a bootstrap sample of the original training data. This involves randomly selecting a subset of data points with replacement, ensuring that each tree sees a slightly different subset of the data.
* **Random Feature Selection**: At each node of the decision tree, instead of considering all features to make a split, Random Forest selects a random subset of features. This randomness helps in decorrelating the trees, making the ensemble more robust and less prone to overfitting.
* **Voting Mechanism**: During the prediction phase, each decision tree in the forest independently predicts the class label of the input data point. The final prediction is determined by a majority vote (for classification tasks) or by averaging (for regression tasks) the predictions of all individual trees.
  + - 1. **Advantages of Random Forest Classifier:**
* **High Accuracy**: Random Forest tends to produce highly accurate predictions, often outperforming single decision trees and other traditional machine learning algorithms.
* **Robustness to Overfitting**: By averaging the predictions of multiple trees, Random Forest mitigates overfitting, making it less sensitive to noise and outliers in the data.
* **Feature Importance**: The algorithm provides a measure of feature importance, allowing users to identify the most influential features in the dataset.
* **Handles Missing Data**: Random Forest can handle missing values in the dataset without the need for imputation, simplifying the preprocessing step.
  + - 1. **Applications of Random Forest Classifier:**
* **Classification Tasks**: Random Forest Classifier is widely used for classification tasks in fields such as finance, healthcare, marketing, and more. It can classify data into multiple classes based on a set of features, making it useful for predicting customer churn, disease diagnosis, sentiment analysis, and fraud detection.
* **Feature Selection**: Random Forests can be used to identify the most important features in a dataset. By analyzing feature importance scores generated by the algorithm, researchers and data scientists can prioritize which features have the most significant impact on the outcome, aiding in feature selection and dimensionality reduction.
* **Anomaly Detection**: Random Forests are effective for anomaly detection in datasets where the majority of data points are normal, but anomalies or outliers exist. By training the algorithm on normal instances, it can identify deviations from the norm, making it useful for detecting fraudulent transactions, network intrusions, and manufacturing defects.
* **Regression Analysis**: While Random Forests are primarily known for classification tasks, they can also be adapted for regression analysis. Random Forest Regression models can predict continuous outcomes, making them suitable for forecasting sales, stock prices, housing prices, and other numerical variables.
* **Customer Segmentation**: Random Forests can be utilized for customer segmentation, where the goal is to divide customers into distinct groups based on their characteristics or behaviors. This segmentation helps businesses tailor marketing strategies, personalize product offerings, and improve customer satisfaction.
* **Image Recognition**: In computer vision applications, Random Forests can be used for image classification and object detection tasks. By extracting features from images and training the algorithm on labeled data, it can accurately classify images into predefined categories or detect objects within images.
* **Biomedical Research**: Random Forest Classifier is employed in biomedical research for tasks such as disease diagnosis, drug discovery, and genomic analysis. It can analyze large-scale biological datasets to identify biomarkers, predict patient outcomes, and uncover patterns in complex biological systems.
* **Environmental Monitoring**: Random Forests are applied in environmental science for tasks such as land cover classification, species distribution modeling, and climate change analysis. They can provide insights into ecosystem dynamics and inform conservation efforts.

Overall, the Random Forest Classifier is a versatile and widely used algorithm suitable for various classification tasks, including but not limited to, predicting student performance in educational settings, medical diagnosis, and financial forecasting.Top of Form

* + 1. **ANN:**

ANN is a popular technique in EDM, and is designed to mimic the structure of the human brain to solve complex problems. It consists of a set of units that accept a weighted set of inputs and responds with an output. Many published papers have used ANN to predict student’s performance. Also, used it because of its ability to detect all possible interactions among variables and its ability to learn from a limited set of examples.

In addition, ANN models were found in a previous study to outperform classification techniques in correctly classifying admitted applicant who will accepted and not accepted. In this study, the usage of Multilayer Perception topology for the ANN model due to the nature of the datasets, which were not large enough to require more complex topologies. Artificial Neural Networks (ANNs) are a class of machine learning models inspired by the structure and functioning of the human brain. ANNs, often referred to as neural networks or simply "deep learning" models, have gained significant attention due to their ability to solve complex problems across various domains.

* + - 1. **Methods and Techniques of ANNs:**

Artificial Neural Networks (ANNs) are a class of machine learning models inspired by the structure and functioning of the human brain. ANNs consist of interconnected nodes, called neurons, organized in layers. Here are some common methods and techniques used in training and optimizing artificial neural networks:

* **Feedforward Neural Networks (FNNs):** These are the simplest form of neural networks, where information flows in one direction, from input nodes through hidden layers to output nodes. They are used for tasks such as classification and regression.
* **Backpropagation:** Backpropagation is the primary algorithm used to train feedforward neural networks. It works by iteratively adjusting the weights of the connections between neurons to minimize the error between the predicted output and the actual output. This is achieved by computing the gradient of the error with respect to the network's parameters and updating the weights accordingly.
* **Convolutional Neural Networks (CNNs):** CNNs are specialized neural networks designed for processing structured grid-like data, such as images. They use convolutional layers to automatically learn spatial hierarchies of features from the input data, making them highly effective for tasks like image classification, object detection, and image segmentation.
* **Recurrent Neural Networks (RNNs):** RNNs are neural networks designed for sequential data processing, such as time series data or natural language text. They use recurrent connections to incorporate information from previous time steps, enabling them to model temporal dependencies and sequential patterns.
* **Long Short-Term Memory (LSTM) Networks:** LSTMs are a type of RNN that addresses the vanishing gradient problem by introducing gated units, allowing the network to learn long-term dependencies in sequential data more effectively. LSTMs are commonly used in tasks such as speech recognition, language translation, and sentiment analysis.
  + - 1. **Overview of ANNs:**
* **Basic Structure of an Artificial Neural Network:** Neurons are the fundamental units of an ANN, each one representing a mathematical function.

Neurons are organized into multiple layers:

1. Input Layer: Receives the initial data.
2. Hidden Layers: Process data through a series of transformations.
3. Output Layer: Produces the final result, such as a classification or regression prediction.
4. Weights and Biases: Each connection between neurons has an associated weight, which determines the strength of the connection. Biases are added to neurons to adjust their input/output relationship.

* **Activation Functions:** Activation functions introduce non-linearity into the network, allowing it to model complex relationships. Common activation functions include ReLU (Rectified Linear Unit), Sigmoid.
* **Feedforward Process:** During the feedforward process, data is passed through the network from the input layer to the output layer, with each neuron applying its activation function to the weighted sum of its inputs.
* **Training an ANN:** ANNs are trained through a process called backpropagation. It involves the following steps:
  1. Forward Pass: The input data is fed forward through the network to make predictions.
  2. Calculate Error: The error (the difference between predictions and actual values) is computed.
  3. Backward Pass: The error is propagated backward through the network, adjusting weights and biases through gradient descent to minimize the error.
  4. Iterative Process: This process is repeated for multiple epochs until the network's performance converges.
* **Loss Functions:** Loss functions quantify the error between predictions and true values. The choice of loss function depends on the specific task, such as mean squared error (MSE) Optimizers are algorithms used during training to adjust the model's weights and minimize the loss function. Examples include stochastic gradient descent (SGD), Adam, and RMSprop.
* **Applications of ANNs:** ANNs have been successful in a wide range of applications, including image recognition, natural language processing, autonomous driving, recommendation systems, and more.
* **Challenges:** Deep learning models can require large amounts of data and computational resources. Overfitting is a common concern, and finding the right architecture and hyper parameters can be challenging.
  1. **REQUIREMENTS:**

The following are the hardware and software requirements considered to implement the proposed system:

**2.2.1 HARDWARE REQUIREMENTS:**

Processor : Intel I5 11g @ 2.40GHZ

RAM : 8 GB

Hard disk : 1 TB

**2.2.2 SOFTWARE REQUIREMENTS:**

Operating System : Windows 11

Programming Language : Python 3.11.5

Framework : Django 4.2

Database : Local File System

Design Tools : STAR UML 5

**2.3 FEASIBILITY STUDY:**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company.

For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are

* Economical Feasibility
* Technical Feasibility
* Social Feasibility

**2.3.1 ECONOMICAL FEASIBILITY:**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### 2.3.2 TECHNICAL FEASIBILITY:

### This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement; as only minimal or null changes are required for implementing this system.

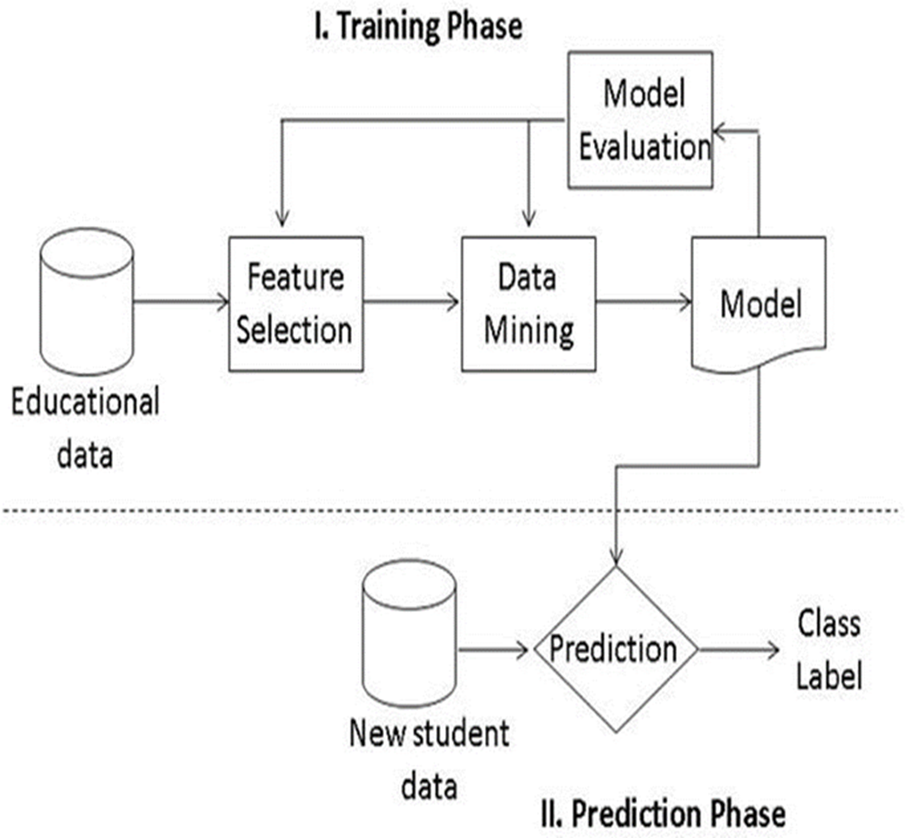
### 2.3.3 SOCIAL FEASIBILITY:

This aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it.

Chapter-III

**SYSTEM DESIGN**

**3.1 ARCHITECTURE:**

****

**Figure 3.1:** SystemArchitecture

The training phase for using data mining techniques to predict student performance in university admission systems involves several key steps:

* **Educational Data:** The educational data need to be collected from the datasets and need to be processed.
* **Data Collection**: The first step is to gather relevant data from various sources, including student demographics, academic records, standardized test scores, extracurricular activities, and any other factors that may influence student performance. This data may include attributes like previous academic performance, extracurricular activities, personal statements, and other factors believed to influence student success. This data is typically stored in a structured format, such as a database, to facilitate analysis.
* **Data Preprocessing**: Once the data is collected, it needs to be preprocessed to ensure its quality and suitability for analysis. This involves tasks such as handling missing values, removing duplicates, normalizing or scaling numerical features, and encoding categorical variables into numerical representations.
* **Feature Selection**: In this step, relevant features that are likely to have an impact on student performance are selected from the dataset. Feature selection techniques help to reduce dimensionality and focus on the most informative variables, improving the efficiency and accuracy of the predictive model.
* **Model Selection**: Next, an appropriate data mining model is selected for predicting student performance. Common models used in this context include decision trees, random forests, logistic regression, support vector machines, and neural networks. The choice of model depends on the specific characteristics of the dataset and the desired predictive performance.
* **Model Training**: The selected model is trained on the preprocessed dataset using a portion of the data known as the training set. During the training phase, the model learns the underlying patterns and relationships between the input features (student attributes) and the target variable (student performance). This is achieved by optimizing the model parameters to minimize prediction errors.
* **Model Evaluation**: After training the model, it is evaluated using a separate portion of the data called the validation set or test set. The performance of the model is assessed using appropriate evaluation metrics, such as accuracy, precision, recall, F1-score, or area under the receiver operating characteristic (ROC) curve. This step helps to determine how well the model generalizes to unseen data and whether the model's performance on the testing set provides insights into its effectiveness in predicting student performance.
* **New Student Data**: The process begins with collecting relevant data about new students who are applying for admission to the university. This data can include various factors such as academic records, standardized test scores, extracurricular activities, demographic information, and socio-economic background. Now, by predicting previous existing student’s data accuracies this new student data performances will be compared and analyzed for process of decision making.
* **Class Label Prediction:** In the context of predicting student performance, the class label typically represents the outcome or category of interest, such as whether a student is likely to succeed or fail academically. Based on the predictions generated by the model, each new student is assigned a class label indicating their predicted performance level, which can then be used to support decision-making in university admission systems.

**3.2 MODULES:**

The proposed system has 2 modules:

* + User
  + System
    1. **USER:**

User is the college admission officer who uses the system to analyze the candidate’s past academic performance.The admission officer consists of few features to analyze the candidate’s past academic performance. They include:

* **Upload Data:** User can upload dataset to the system for predicting the student performance by this upload data functionality using this feature of upload data.
* **Open Filename:** Thisfeature of User module provides users to select a particular type of file dataset from various types of files for predicting the student’s performance.
  + 1. **SYSTEM:**

System is a predictor which processes the candidate’s academic data and helps user to take decision. It can process the data by using algorithms and can predict the results. The key features that such a system module might include:

* **Data import:** Data import is a fundamental feature within the system module function for uploading data into the university admission system for analysis and prediction of student performance using data mining techniques. This feature also ensures smooth and efficient transfer of data into the university admission system.
* **Data Preprocessing:** The data preprocessing feature within the system module is essential for optimizing data quality and structure before employing data mining techniques to predict student performance. This process involves cleaning the data by handling missing values and outliers, transforming variables, and possibly creating new features.
* **Train and Test Model:** This feature of system module involves splitting the dataset into training and testing subsets. The training subset is used to build predictive models, while the testing subset evaluates model performance. By iteratively refining models based on testing results, this feature ensures the accuracy and reliability of predictions.
* **Run Support Vector Machine:** This feature of system module enables users to utilize Support Vector Machine (SVM) algorithms, a powerful tool for classification and regression tasks, to analyze historical student data and forecast future academic outcomes. By leveraging SVM, users can effectively model complex relationships within the data.
* **Run Random Forest:** This feature allows users to train and deploy random forest models using uploaded student data. Random forest is effective for classification tasks, making it suitable for predicting various aspects of student performance, aiding universities in making informed decisions.
* **Run Naïve Bayes:** The run Naive Bayes feature within the system module provides efficient and probabilistic predictions, aiding in the selection process by assessing the likelihood of a student's success based on available data.
* **Run Decision Tree:** The run Decision Tree feature within the system module enables in utilizing decision tree algorithms to analyze student data and generate predictive models. By employing decision trees, the system can identify key factors influencing student outcomes, facilitating informed decisions regarding admissions
* **Run ANN:** The run ANN feature in the system module enables the application of artificial neural networks (ANN) for predicting student performance. ANN, a type of machine learning algorithm inspired by the structure of the human brain, analyzes student data to provide insights crucial for admissions decisions.
* **Accuracy graph:** This feature within the system module provides a visual representation of the predictive model's performance in forecasting student’s performance. This graph displays how the model's accuracy evolves over time or across different iterations of training and validation.

**3.3 DESIGN REPRESENTATION:**

The proposed system design models are created using UML tool. UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

UML consists an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems.

Unified Modeling Language is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

The goal of UML is to provide a standard notation that can be used by all object-oriented methods and to select and integrate the best elements of precursor notations. UML has been designed for a broad range of applications. Hence, it provides constructs for a broad range of systems and activities (e.g., distributed systems, analysis, system design and deployment).

* **StarUML:**

StarUML is a [UML tool](https://en.wikipedia.org/wiki/UML_tool) by MK Lab StarUML™ is a software modeling platform that supports UML (Unified Modeling Language). The user can freely create and manage different diagrams as needed. It is based on UML version 1.4 and provides eleven different types of diagrams, and it accepts UML 2.0 notation.

StarUML 2.0 uses its own file format, with the. uml extension. StarUML™ supports various UML diagram types. The user can freely create and manage different diagrams as needed.

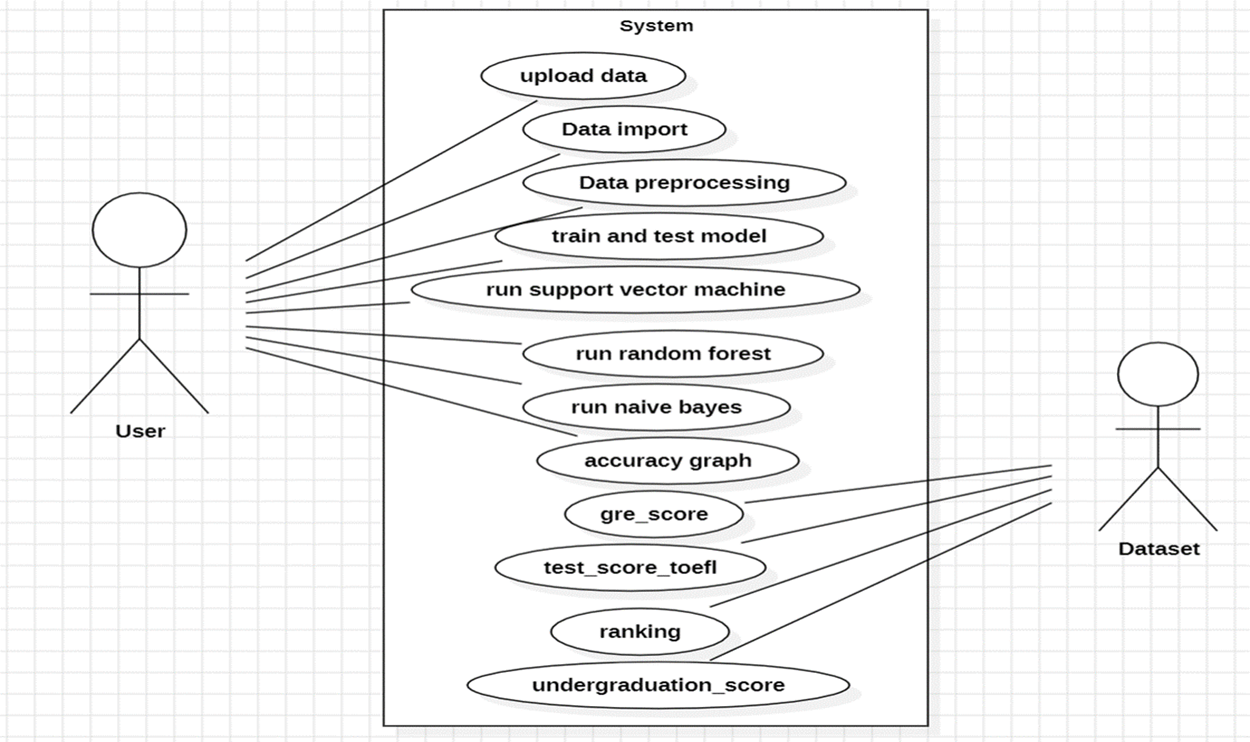
|  |
| --- |
| 1. Class Diagram 2. Object Diagram |
| 1. Use Case Diagram |
| 1. Sequence Diagram |
| 1. Collaboration Diagram |
|  |
| 1. State chart Diagram |
| 1. Activity Diagram |
| 1. Component Diagram 2. Deployment Diagram |

****

**Figure 3.2:** StarUML 5 Interface

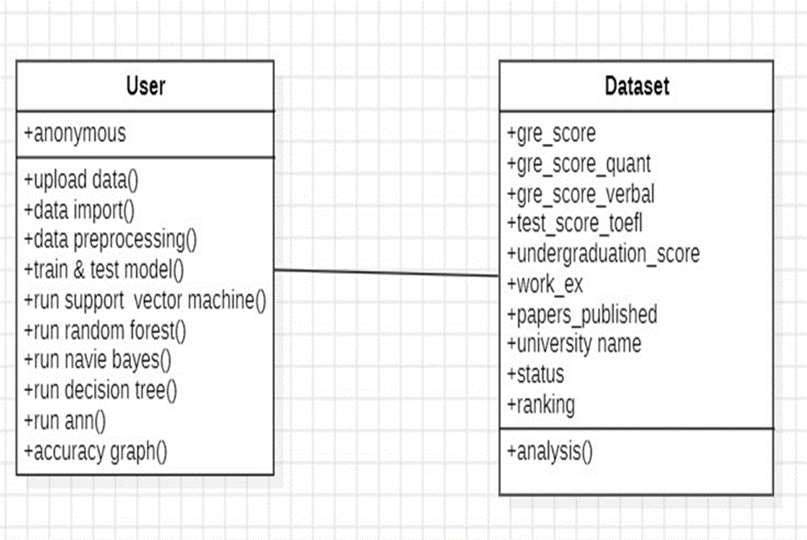
**3.3.1 USE CASE DIAGRAM:**

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.



**Figure 3.3:** Use Case Diagram

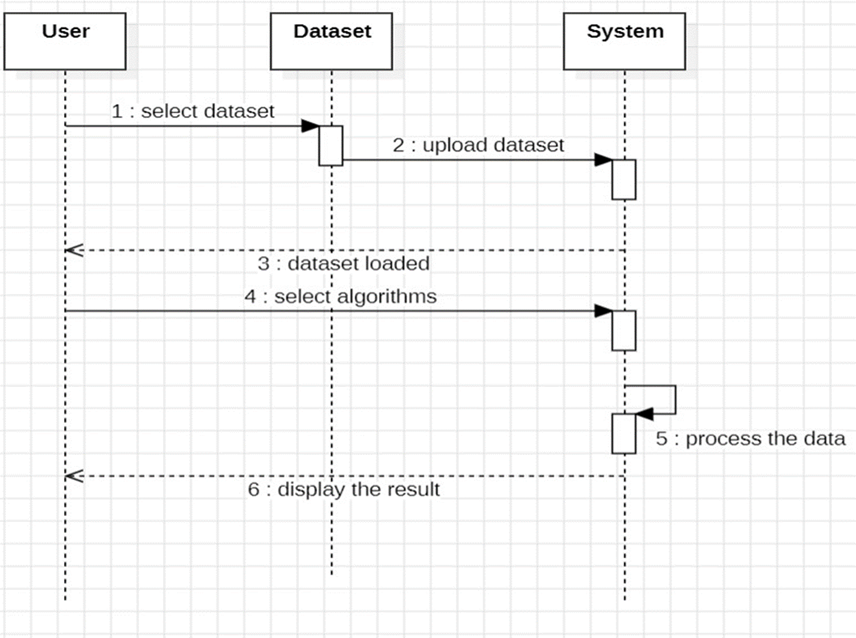
**3.3.2 CLASS DIAGRAM:**

 In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

**Figure 3.4:** Class Diagram

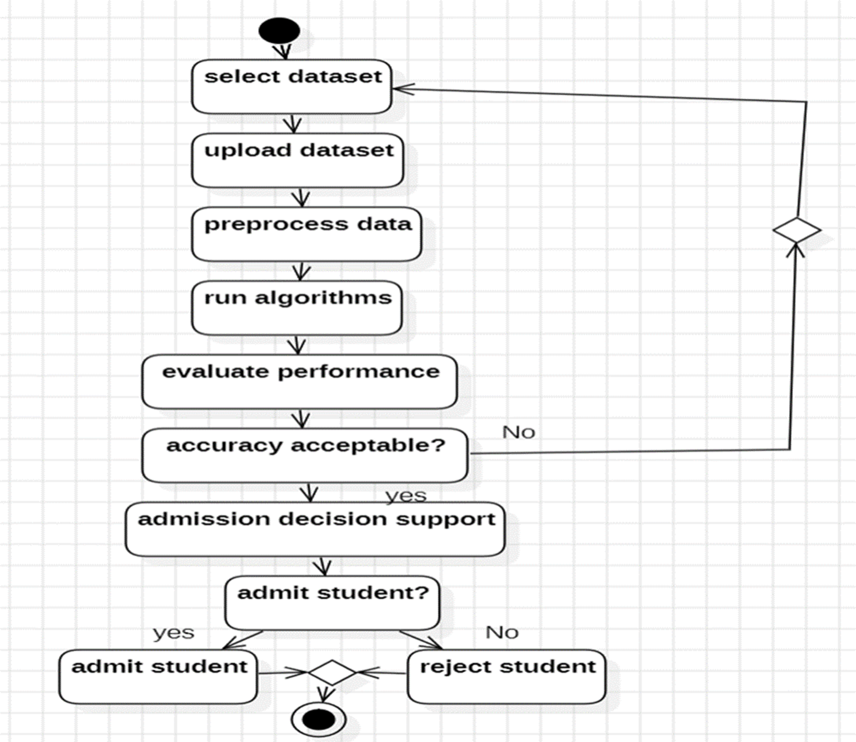
**3.3.3 SEQUENCE DIAGRAM:**

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.



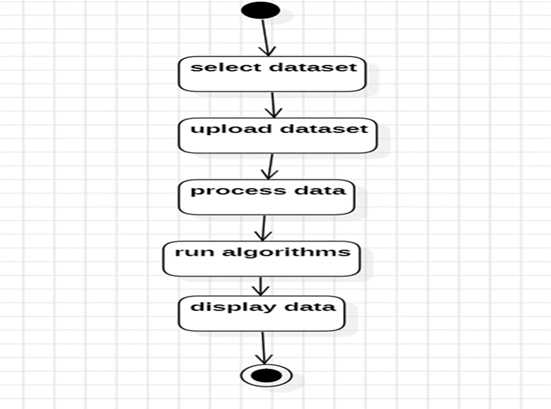
**Figure 3.5:** Sequence Diagram

**3.3.4 ACTIVITY DIAGRAM:**

 Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

**Figure 3.6:** Activity Diagram

**3.3.5 STATECHART DIAGRAM:**

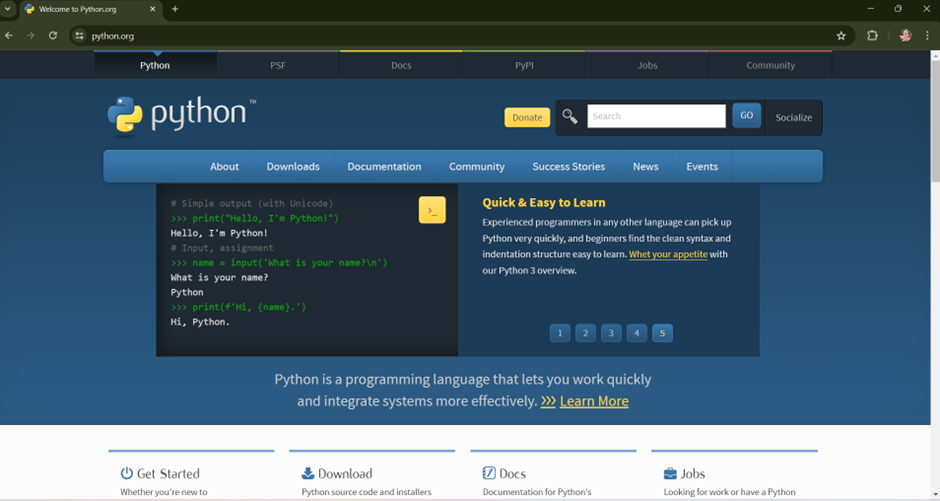
**** In Unified Modeling Language (UML), a statechart diagram depicts about the various states and transitions of an object or system. They are particularly useful for modeling the behavior of complex systems or entities with multiple states.

**Figure 3.7:** State Chart Diagram

Chapter-IV

**IMPLEMENTATION**

**4.1 TECHNOLOGIES:**

**4.1.1 PYTHON:**

**Figure 4.1:** Python Websource

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python is Interpreted − Python is processed at runtime by the interpreter. Python need to compile your program before executing it. This is similar to PERL and PHP. Python is Interactive – everyone can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc. The biggest strength of Python is huge collection of standard libraries which can be used for the following –

* Machine Learning
* GUI Applications (like Kivy, Tkinter, PyQt etc.)
* Web frameworks like Django (used by YouTube, Instagram, Dropbox)
* Image processing (like OpenCV, Pillow)
* Web scraping (like Scrapy, BeautifulSoup, Selenium)
* Test frameworks
* Multimedia

Advantages of Python:

* Extensive Libraries
* Extensible
* Readable
* Object-Oriented
* Free and Open-Source
* Portable
* Interpreted

Applications of Python

* Web Development
* Data Science and Machine Learning
* Scientific Computing
* Artificial Intelligence and Natural Language Processing
* Automation and Scripting
* Education and Training
* Finance and Quantitative Analysis

**4.1.2 LIBRARIES:**

* **Tkinter:**

TKinter is a Python library used for creating graphical user interfaces (GUIs). It provides a simple and efficient way to develop user interfaces for applications running on Windows, macOS, and Linux. TKinter is based on the Tk GUI toolkit, which is a cross-platform library that originated as part of the Tcl scripting language.

* **Pandas:**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem.

* **NumPy:**

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.It is the fundamental package for scientific computing with Python.

* **Matplotlib:**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

* **Seaborn:**

Seaborn is a Python data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. It is built on top of Matplotlib and integrates tightly with the PyData stack, including Pandas and NumPy.

* **Plotly:**

Plotly is a Python graphing library known for its interactivity and versatility. It enables the creation of beautiful and dynamic visualizations with ease. Plotly offers a range of charts, including line, scatter, bar, heatmaps, 3D surfaces plots, and many others. It allows for interactive exploration of data, with support for zooming, panning, and hover tooltips.

* **Scikit – learn:**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. Scikit-learn, also known as sklearn, is a popular Python library for machine learning.

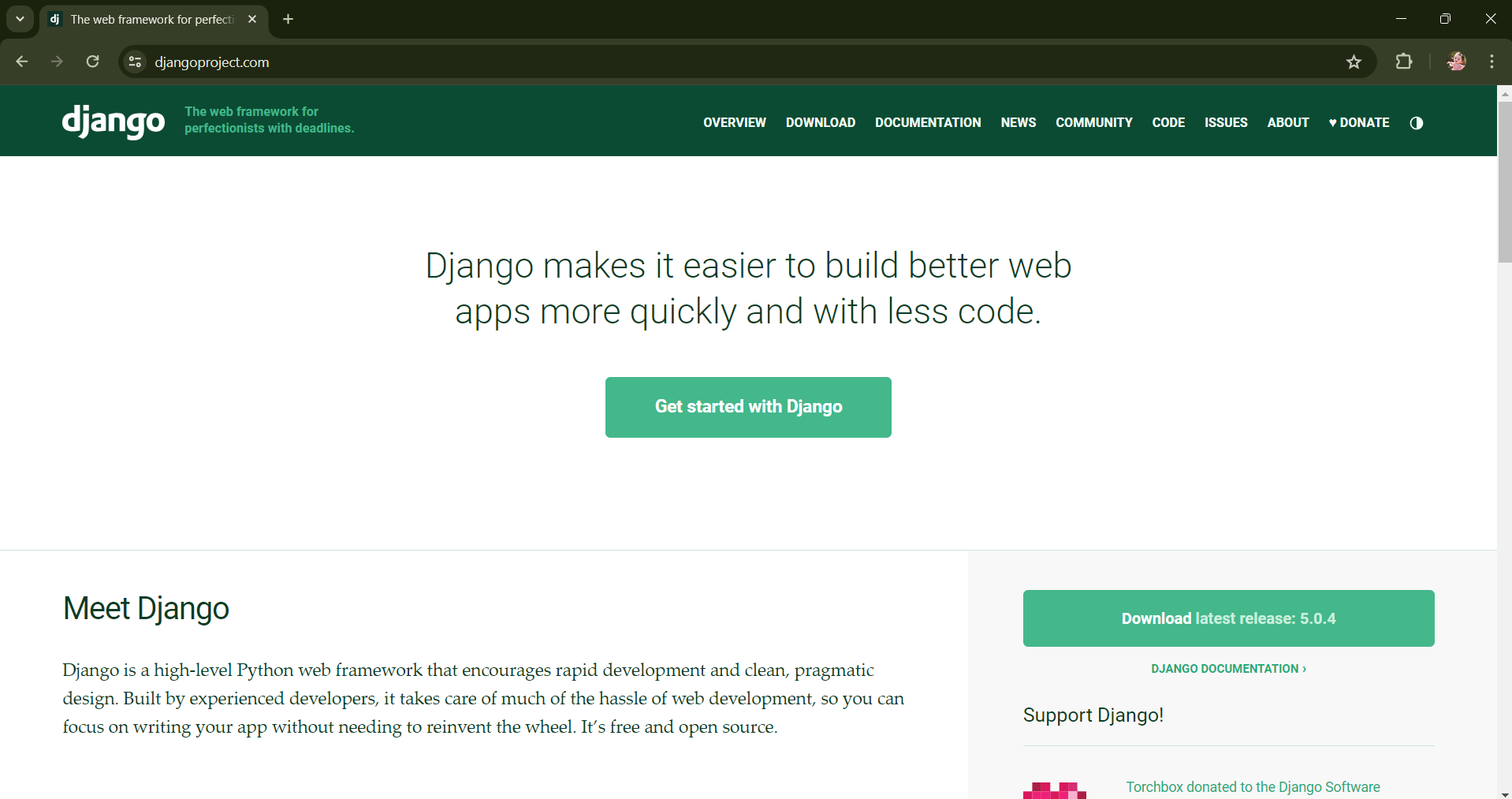
* **TensorFlow:**

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

* **Keras:**

Keras is a high-level neural networks API, originally developed as part of the TensorFlow library, that facilitates building, training, and deploying deep learning models with ease. It provides a user-friendly interface for creating neural networks, allowing developers to focus more on model architecture and less on low-level implementation details.

**4.1.3 DJANGO:**

****

**Figure 4.2:** Django Websource

#### Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Developed by Django Software Foundation, it follows the Model-View-Controller (MVC) architectural pattern, although in Django, it's commonly referred to as Model-View-Template (MVT).

Key Features of Django:

* **Object-Relational Mapping (ORM):** Django provides a powerful ORM system that allows developers to interact with the database using Python objects, without needing to write SQL queries directly.
* **Admin Interface:** Django comes with a built-in administration interface that can be easily customized and extended, providing CRUD (Create, Read, Update, Delete) operations for your models.
* **URL Routing:** Django uses a URL dispatcher to map URL patterns to views, allowing for clean and flexible URL routing.
* **Template Engine:** Django includes its own template engine, which allows developers to create dynamic HTML templates with template tags and filters.
* **Form Handling:** Django provides a comprehensive form handling library, making it easy to create and process HTML forms.
* **Security:** Django has built-in protection against many common security threats such as SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), and clickjacking.

Django is a versatile web framework that can be used to build a wide range of web applications across various domains. Here are some common types of applications that can be built using Django:

* **Content Management Systems (CMS):** Django's built-in admin interface and powerful ORM make it well-suited for building content management systems.
* **E-commerce Platforms:** Django can be used to build e-commerce platforms and online stores. Everyone can create product catalogs, shopping carts, checkout processes, and payment gateways using Django's models, views, and forms.
* **Social Networking Sites:** Django provides features for user authentication, authorization, and user-generated content, making it suitable for building social networking sites and online communities.
* **Online Marketplaces:** Django can be used to build online marketplaces where users can buy and sell goods or services. One can implement features such as user accounts, listings, search functionality, messaging between buyers and sellers, and transaction management using Django's framework.
* **Educational Platforms:** Django can be used to build educational platforms and learning management systems (LMS). Everyone can create courses, quizzes, assignments, and discussion forums using Django's models, views, and templates. Integration with e-learning standards such as SCORM and xAPI is also possible.
* **Job Boards and Freelance Platforms:** Django can be used to build job boards, freelance platforms, and recruitment portals. Everyone can create job listings, user profiles, job applications, and messaging functionality using Django's built-in components or third-party packages.
* **Real-Time Applications:** Django can be combined with technologies such as WebSockets and asynchronous task queues to build real-time applications such as chat applications, live updates, and collaborative editing tools. Django Channels is a third-party package that provides WebSocket support for Django applications.
* **Data Analytics Platforms:** Django can be used to build data analytics platforms and dashboards for visualizing and analyzing data. Everyone can integrate Django with data processing libraries such as Pandas, NumPy, and Matplotlib to perform data analysis and generate interactive visualizations.
* **API Backends:** Django can be used to build API backends for mobile applications, single-page applications (SPAs), and Internet of Things (IoT) devices. Everyone can use Django REST Framework (DRF), a powerful toolkit for building Web APIs, to create RESTful APIs with authentication, serialization, and validation.

**4.2 SAMPLE CODE:**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

from tkinter. filedialog import askopenfilename

import numpy as np

import pandas as pd

import matplotlib. pyplot as plt

import seaborn as sns

sns.set(style="darkgrid")

import plotly. offline as py

import plotly. figure\_factory as ff

import plotly. graph\_objs as go

from plotly import tools

from plotly. offline import download\_plotlyjs, init\_notebook\_mode, iplot, plot

from sklearn. model\_selection import cross\_val\_score, train\_test\_split

from sklearn. preprocessing import StandardScaler

from sklearn. metrics import accuracy\_score, f1\_score, confusion\_matrix

from sklearn. model\_selection import GridSearchCV

from sklearn. utils import resample

from sklearn. model\_selection import StratifiedKFold

import numpy as np

import numpy as np

from sklearn import metrics

from sklearn. metrics import roc\_curve,auc

import matplotlib. pyplot as plt

def upload ():

global filename

global data

text. delete ('1.0', END)

filename = askopenfilename (initialdir = ".")

pathlabel.config(text=filename)

text. insert (END,"Dataset loaded\n\n")

def importdata ():

global filename

global dataset

dataset = pd. read\_csv(filename)

text. insert (END,"Data Information:\n"+str (dataset. head ()) +"\n")

text. insert (END,"Columns Information:\n"+str (dataset. columns) +"\n")

accept\_list = dataset[dataset['status'] == 'accept']

reject\_list = dataset[dataset['status'] == 'reject']

plt. bar ('accept', len(accept\_list))

plt. bar ('reject', len(reject\_list))

plt. title ('Accept and Reject for Universities')

plt. show ()

print(dataset['status'].value\_counts ())

print ('===================================')

print ('Percentage of Accept and Reject Values in the dataset')

def preprocess ():

global X, y

global dataset

university\_list = dataset['university\_name']. unique (). tolist ()

text. insert (END,"Universities Information: \n"+str('universities') + '\n')

dataset. groupby(by=["university\_name"]). mean () ["ranking"]. sort\_values ()

dataset. groupby(['university\_name','ranking','status']) ['status']. count (). unstack (). plot (title = 'University\_name vs Admit/Reject’, fontsize = 30, figsize= (50,15), kind='bar', legend=False, color= ['g', 'r'])

dataset. groupby(['university\_name','status']) ['test\_score\_toefl']. mean (). unstack (). plot (legend=True, ylim = [85,120], title = 'Accept and Reject of universities on the basis on mean TOEFL Score’, fontsize = 30, figsize= (50,15), kind='bar', color=['g','r'])

dataset. university\_name.value\_counts ()

target\_universities=dataset. university\_name. unique (). tolist ()

resampled\_dfs= []

resampled\_df = pd. DataFrame ()

for each in target\_universities:

if dataset [(dataset. university\_name==each)]. shape [0]> 600:

resampled\_dfs. append (resample (dataset [(dataset. university\_name==each) & (dataset. status=='accept')], replace=True, n\_samples=300, random\_state=123))

resampled\_dfs. append (resample (dataset [(dataset. university\_name==each) & (dataset. status=='reject')], replace=True, n\_samples=300, random\_state=123))

elif dataset [(dataset. university\_name==each)]. shape [0] < 200:

resampled\_dfs. append (resample (dataset [(dataset. university\_name==each) & (dataset. status=='accept')], replace=True, n\_samples=125, random\_state=123))

resampled\_dfs. append (resample (dataset [(dataset. university\_name==each) & (dataset. status=='reject')], replace=True, n\_samples=125, random\_state=123))

else:

resampled\_dfs. append (dataset [(dataset. university\_name==each) & (dataset. status=='accept')])

resampled\_dfs. append (dataset [(dataset. university\_name==each) & (dataset. status=='reject')])

resampled\_df = pd. concat ([ f for f in resampled\_dfs])

resampled\_df. groupby(by='university\_name’) ['status'].value\_counts ()

dataset =resampled\_df. copy ()

text. insert (END,"Preprocess Done: \n")

plt. Figure (figsize = (10, 6))

ax =sns. Boxplot (x='university\_name', y='rescore', data=dataset[dataset['status’] =='accept'])

plt. setp (ax. artists, alpha=1, linewidth=1, edgecolor="k")

plt. title('GRE SCORE Box Plot - Accepts')

plt. xticks(rotation=90)

plt. figure (figsize = (10, 6))

ax =sns. boxplot (x='university\_name', y='gre\_score\_verbal', data=dataset[dataset['status’] =='accept'])

plt. show ()

Chapter-V

**TESTING**

**5.1 TEST STRATEGIES:**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

**5.1.1 UNIT TESTING:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**5.1.2 INTEGRATION TESTING:**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**5.1.3 VALIDATION TESTING:**

Validation/Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

* Valid Input : identified classes of valid input must be accepted.
* Invalid Input : identified classes of invalid input must be rejected.
* Functions : identified functions must be exercised.
* Output : identified classes of application outputs must be exercised.
* Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**5.1.4 SYSTEM TESTING:**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Test cases are generated using two techniques

**5.1.5 White Box Testing:**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It has a purpose. It is used to test areas that cannot be reached from a black box level.

**5.1.6 Black Box Testing:**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. one cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**5.2 TEST CASES:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Inputs** | **Expected Output** | **Actual Output** | **Result** |
| Entering student’s score details performances | Successful consideration | Dataset uploaded | Pass |

**Table 5.1:** Test Case for Uploading Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Inputs** | **Expected Output** | **Actual Output** | **Result** |
| Gre scores, toefl scores, under graduation scores of students | Data set uploaded Successfully and importing the student details | Data Imported Successfully | Pass |

**Table 5.2:** Test Case for Importing the Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Inputs** | **Expected Output** | **Actual Output** | **Result** |
| Updated Student detail performances | Dataset uploaded successfully | Updated Dataset not uploaded | Fail |

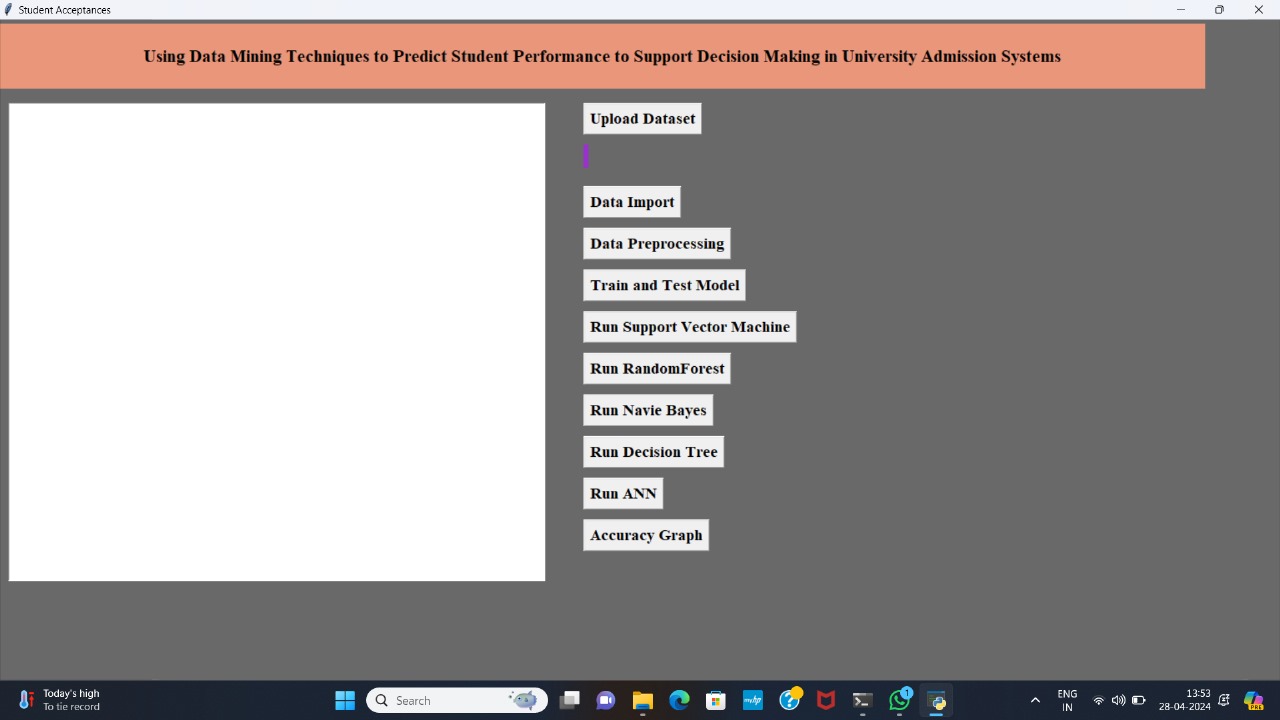
**Table 5.3:** Test Case for Updating Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Inputs** | **Expected Output** | **Actual Output** | **Result** |
| Student details | Accuracies of student displays | Accuracy graph displayed | Pass |

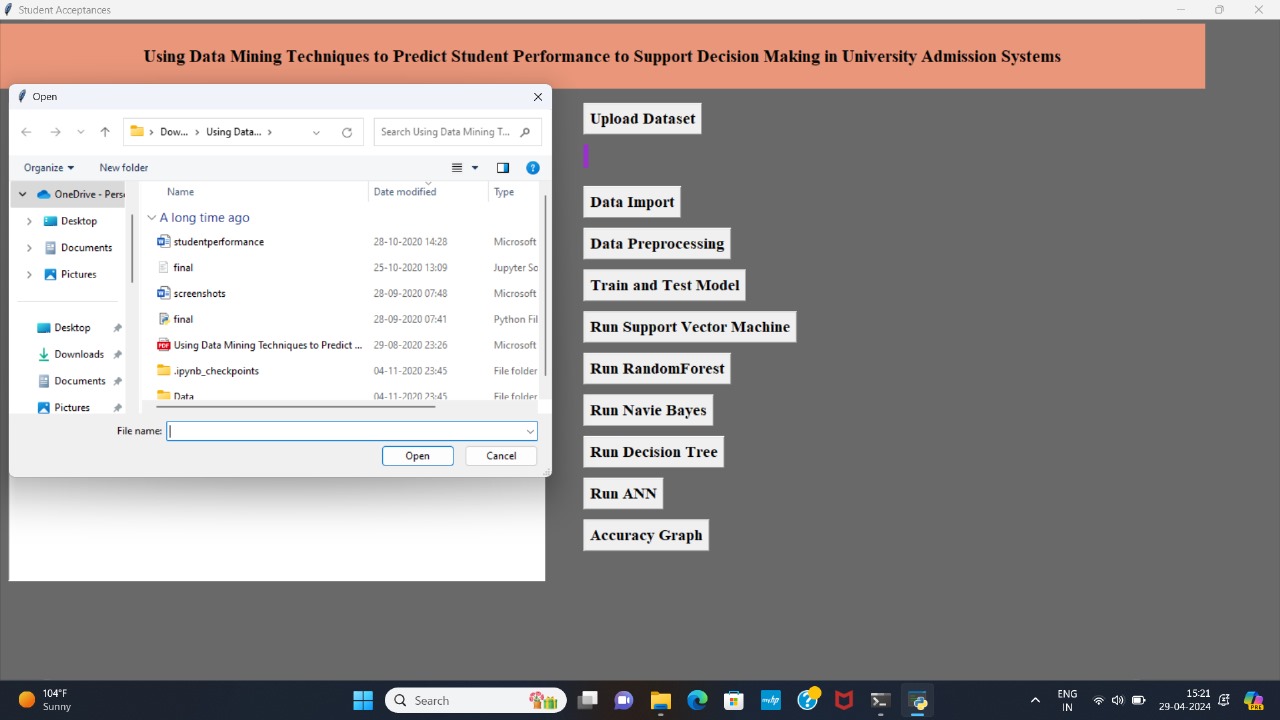
**Table 5.4:** Test Case for Accuracy

Chapter-VI

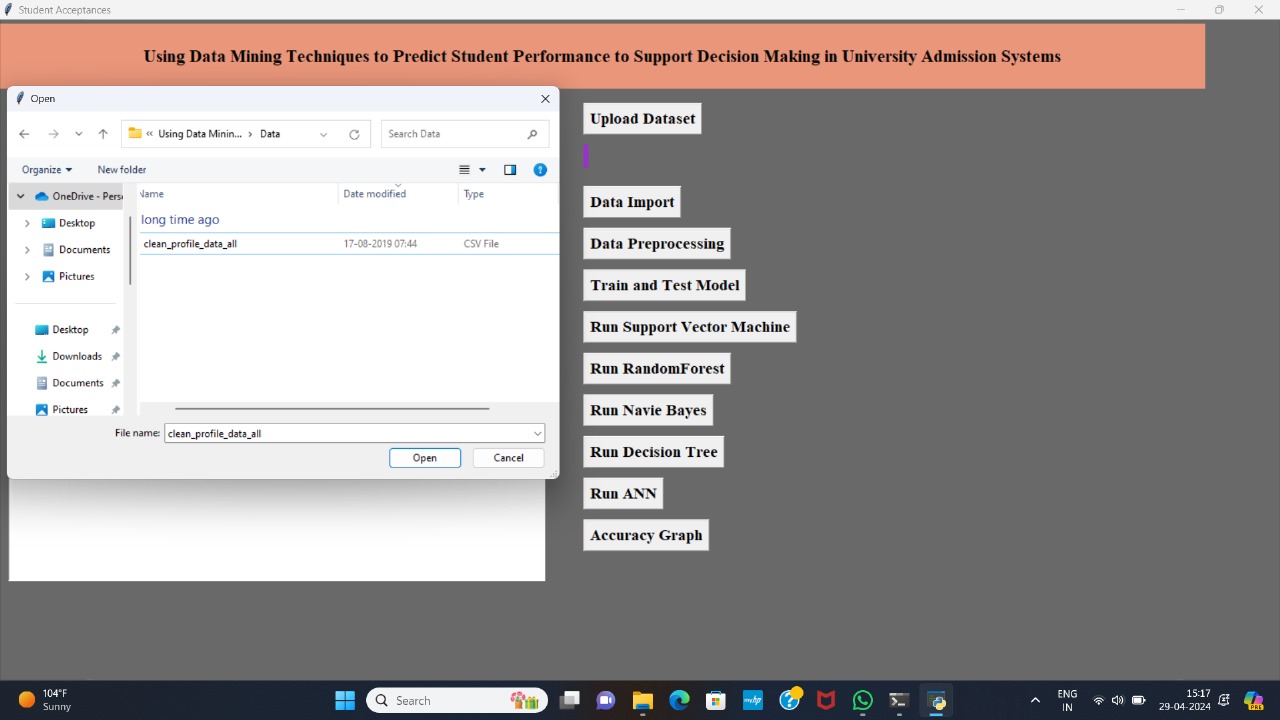
**RESULTS**

****

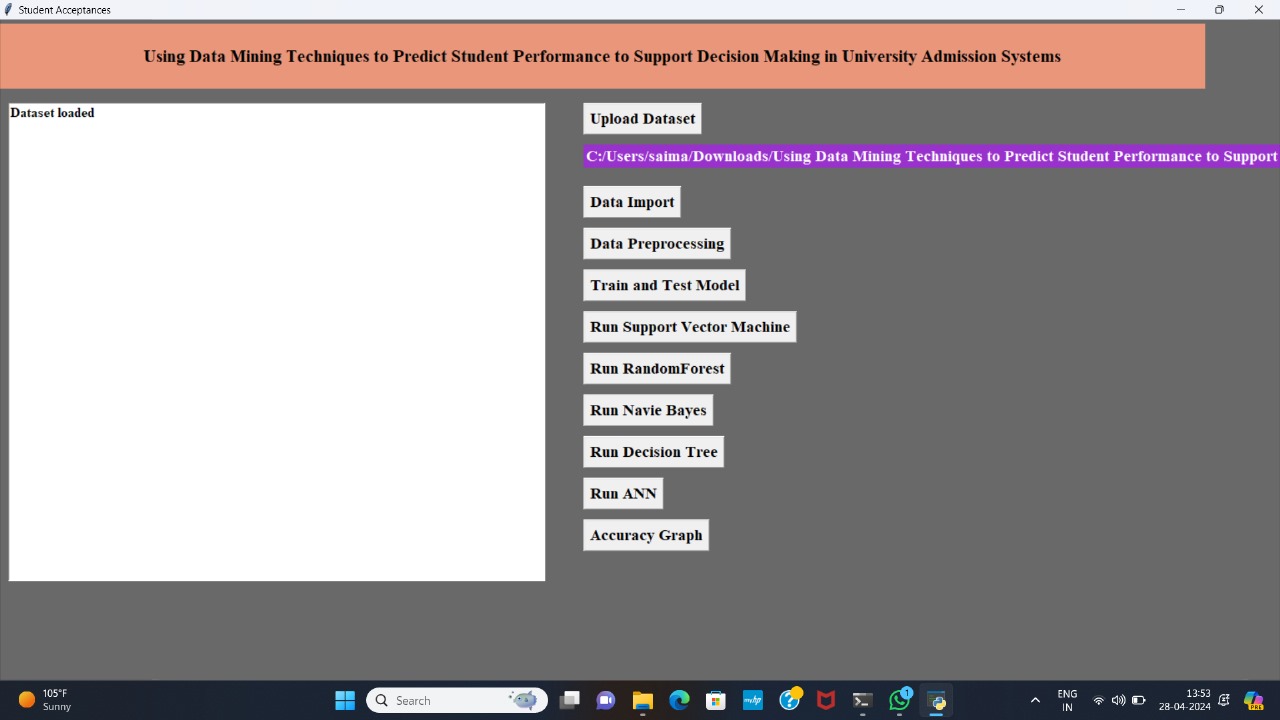
**Figure 6.1**: Home Screen



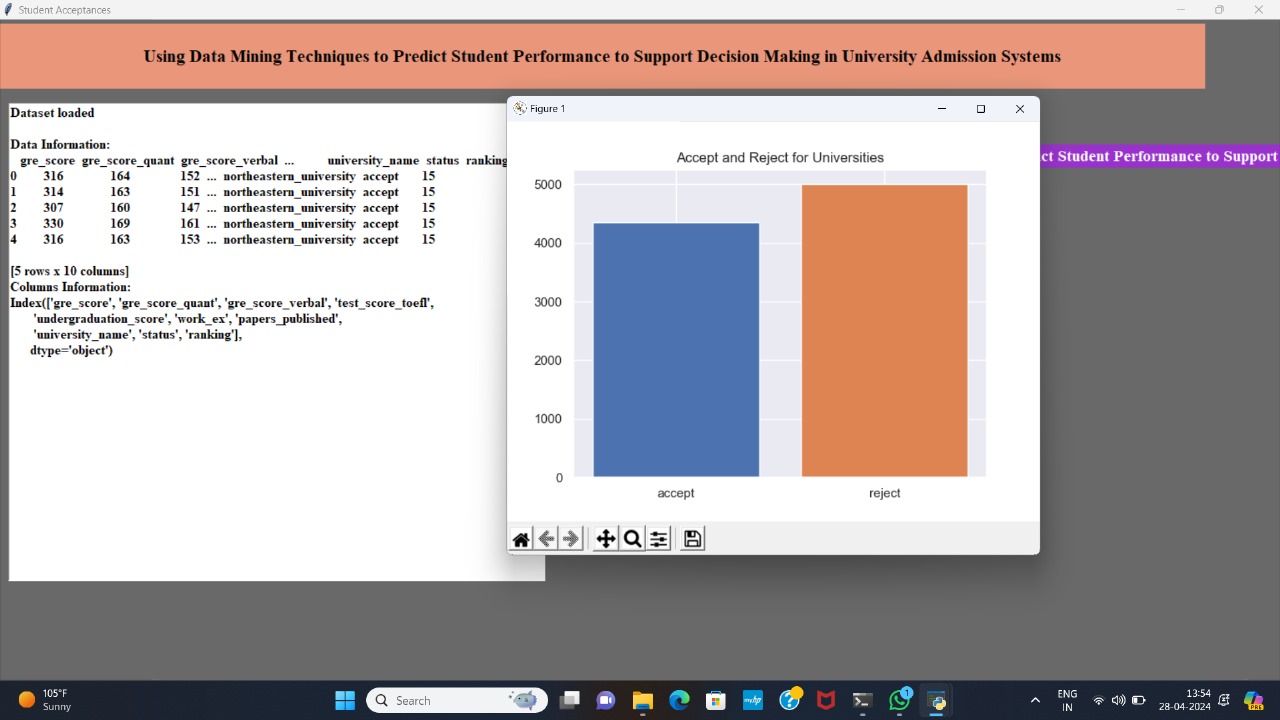
**Figure 6.2**: Choosing the Location of Dataset



**Figure 6.3**: Select Dataset



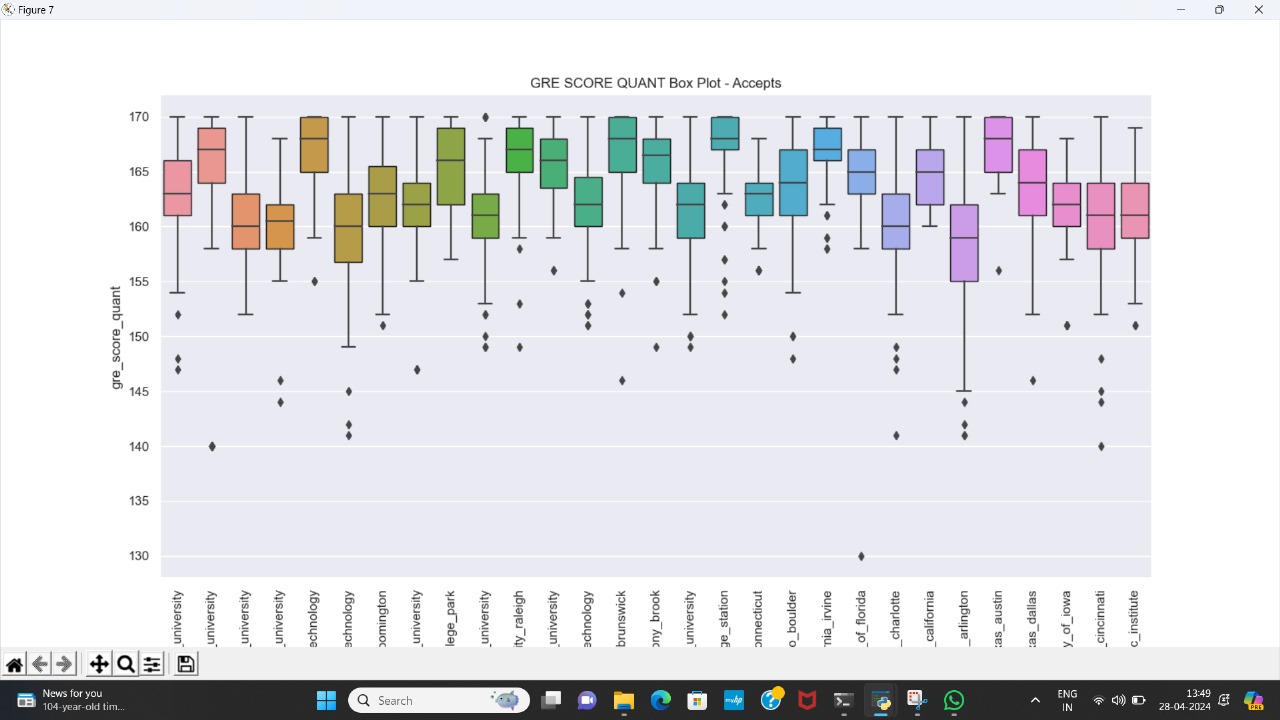
**Figure 6.4**: Dataset Uploaded

****

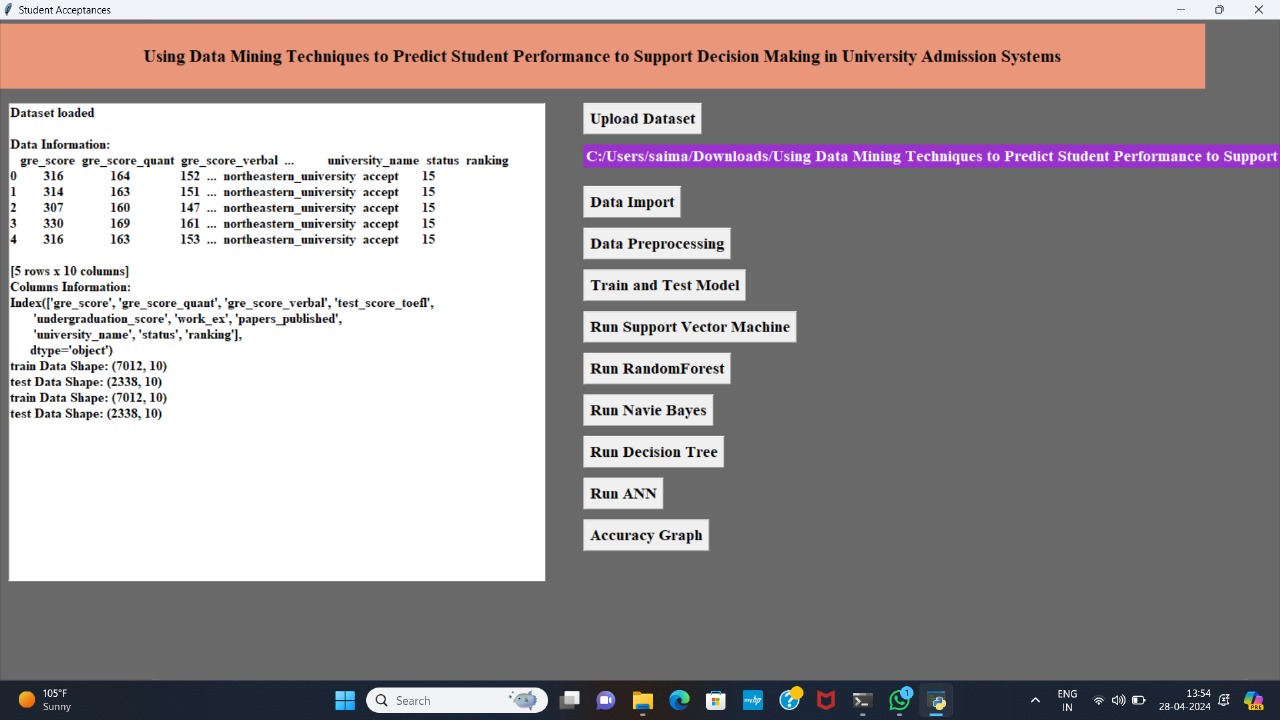
**Figure 6.5**: Importing the Data

****

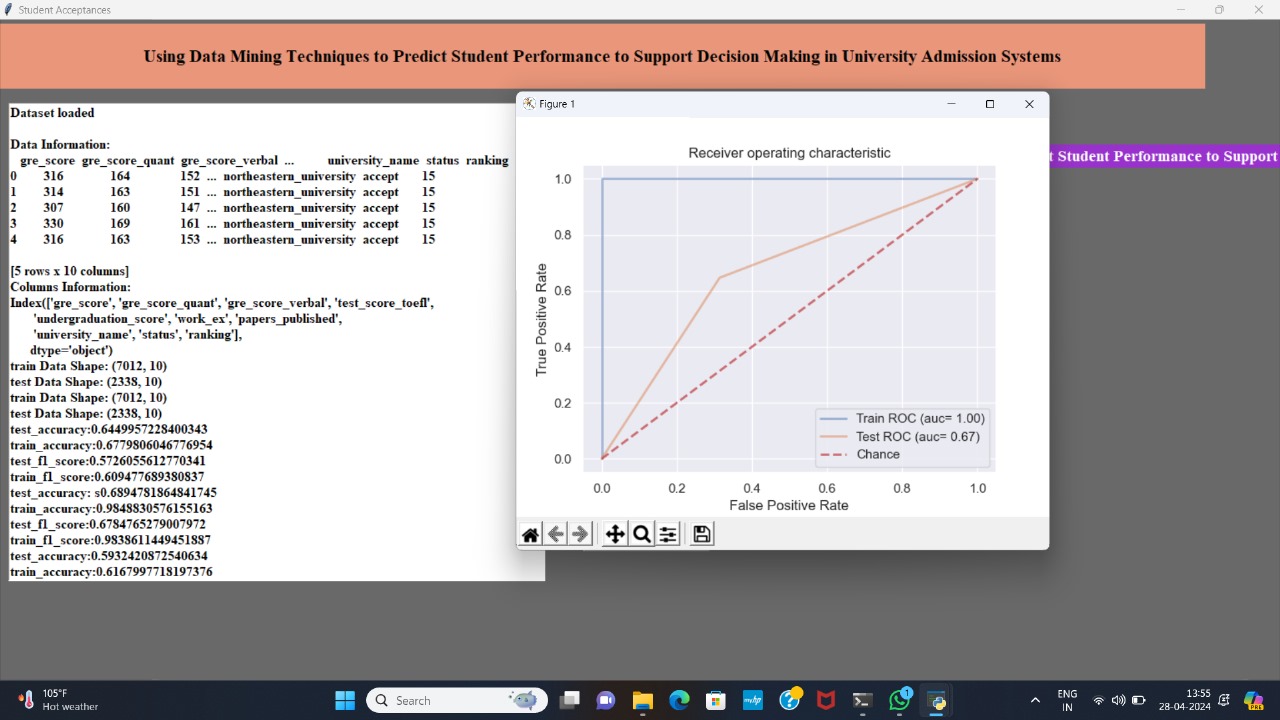
**Figure 6.6**: Data Preprocessing



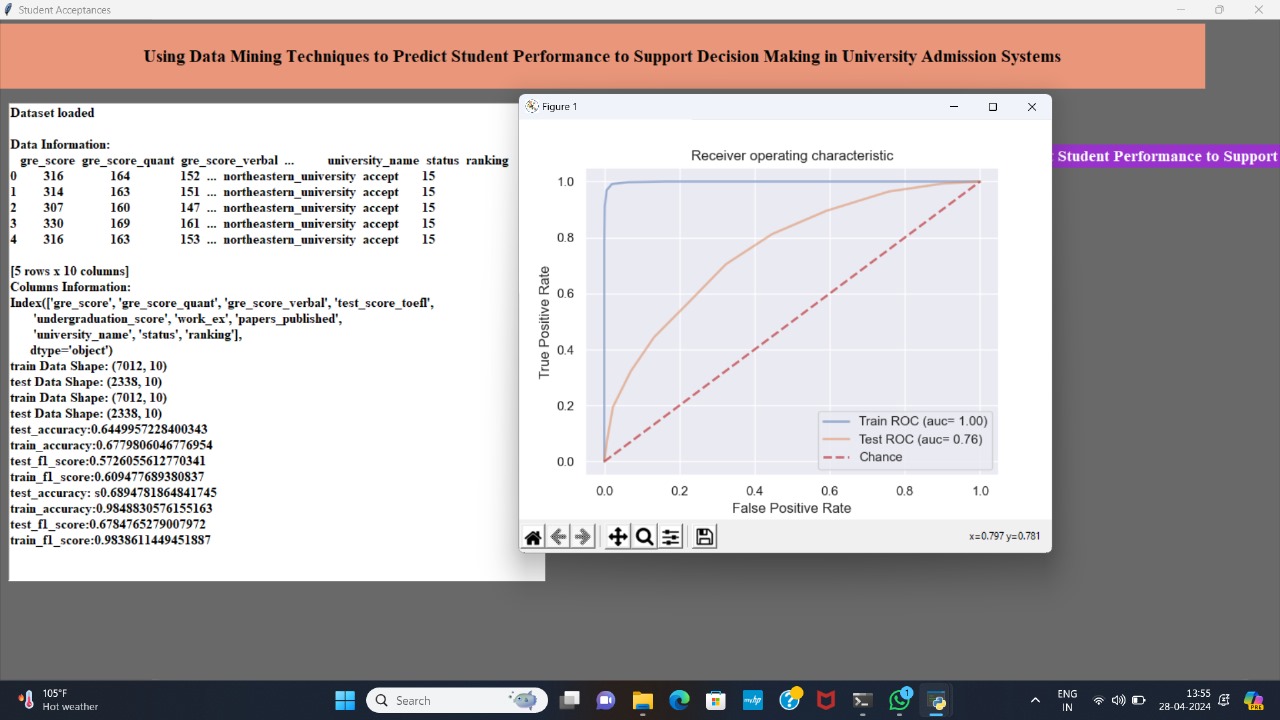
**Figure 6.7**: Data Preprocessing for GRE Score



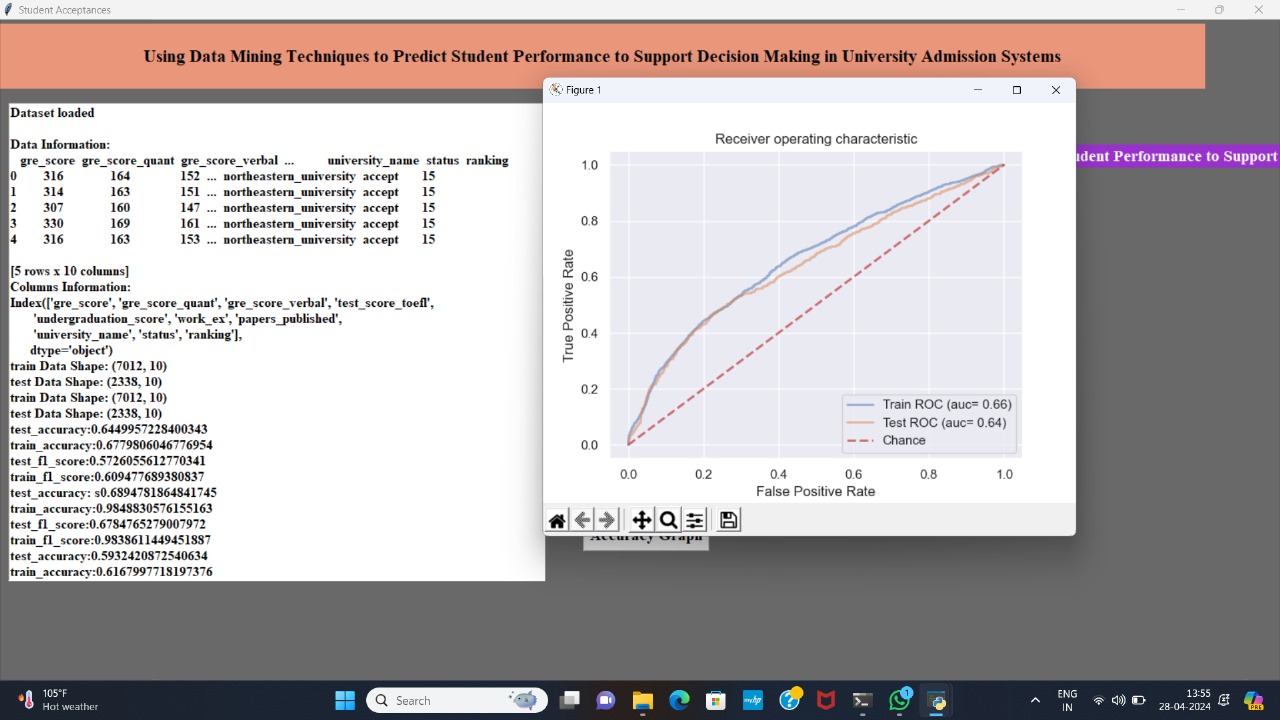
**Figure 6.8**: Train and Test Model Screen



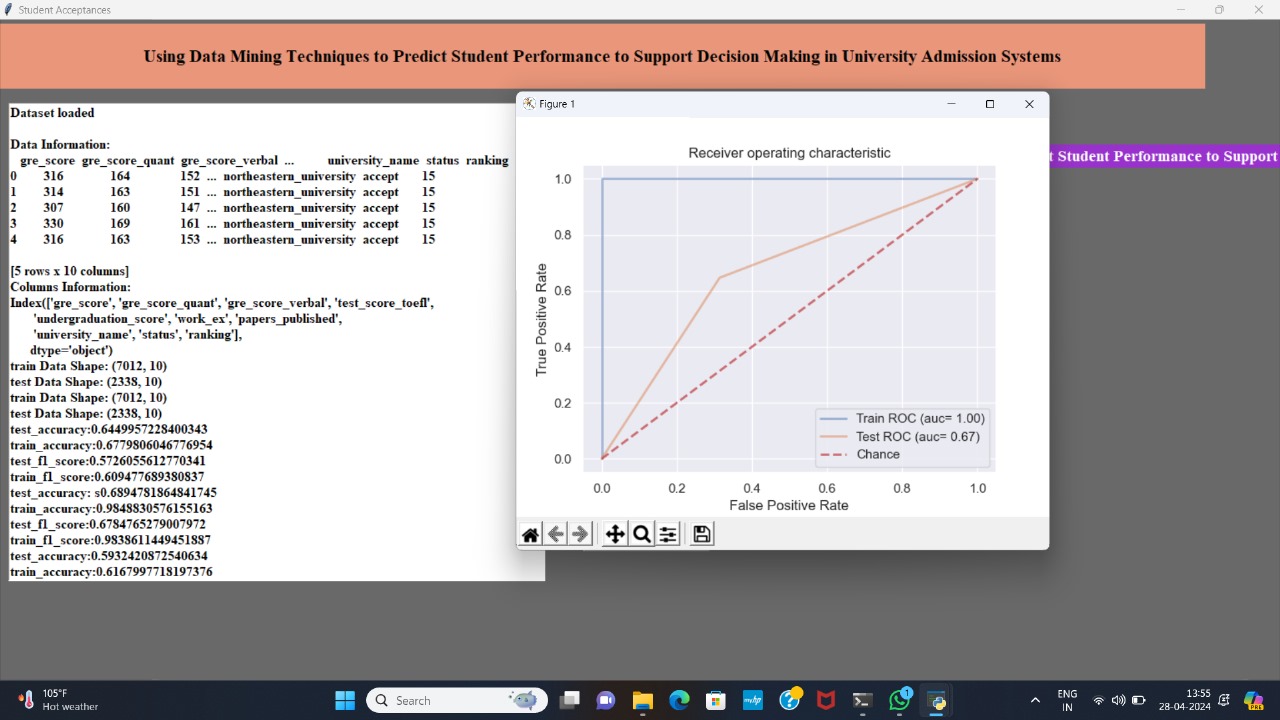
**Figure 6.9**: Support Vector Machine Screen



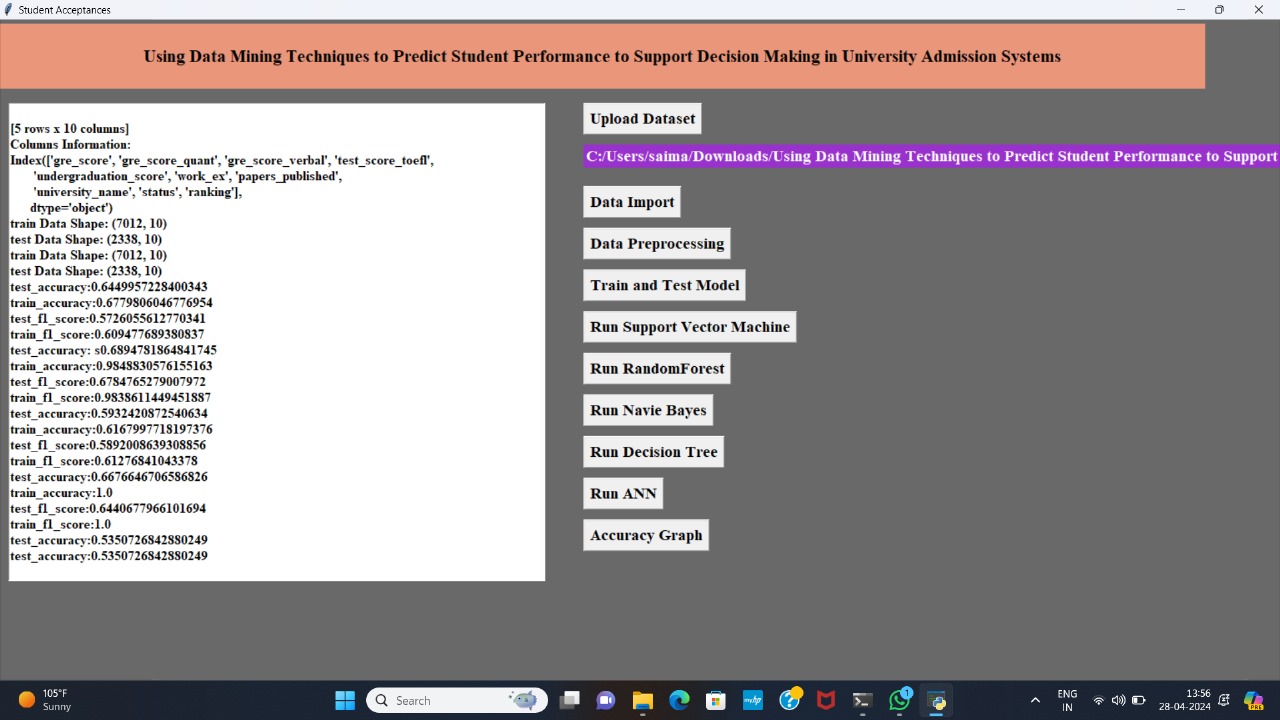
**Figure 6.10**: Random Forest Classifier Screen



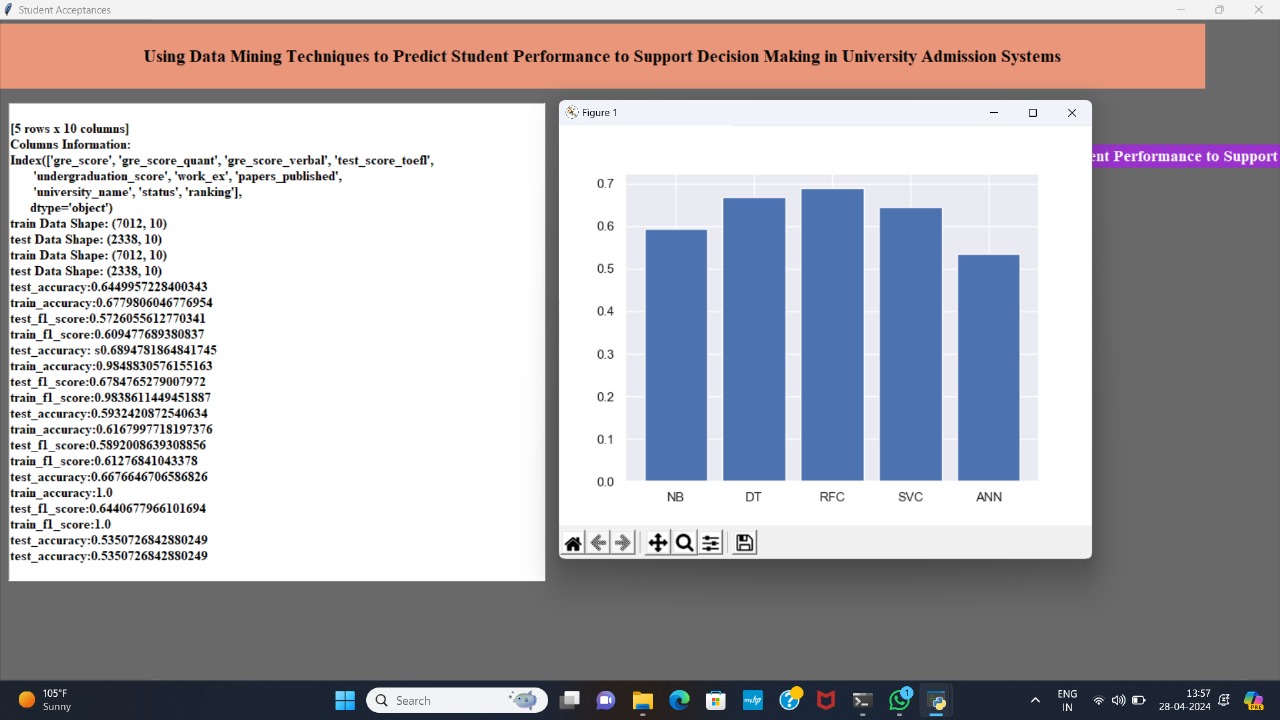
**Figure 6.11**: Naïve Bayes Screen



**Figure 6.12**: Decision Tree Screen



**Figure 6.13**: ANN Screen

****

**Figure 6.14**: Accuracy Graph

Chapter-VII

**CONCLUSION**

The “Admission Recommender System” has developed to address the issues regarding student admissions. The proposed system has implemented using data mining and machine learning algorithms that accepts applicants’ academic data and generates the recommendation as “Accept” and “Reject”. The developed system generates better results compared to existing techniques such as Decision Tree, Naïve Bayes, support vector machine. A data set named “Clean data profile all” with 9351 entries has processed by the system and the predictions along with accuracy graphs are generated as results.

Chapter-VIII

**FUTURE ENHANCEMENTS**

Even Though the proposed system provides the sufficient futures to assist the admission process, still there is scope for improvement the system can be further enhanced to include Natural Language Processing (NLP) techniques so that the process of decision making can become much more effective.

**BIBLIOGRAPHY**

**PUBLICATIONS:**

1. H. Guruler, A. Istanbullu, and M. Karahasan, ‘‘A new student performance analysing system using knowledge discovery in higher educational databases,’’ Comput. Edu., vol. 55, no. 1, pp. 247–254, Aug. 2010.
2. S. K. Mohamad and Z. Tasir, ‘‘Educational data mining: A review,’’ Procedia Social Behav. Sci., vol. 97, pp. 320–324, Nov. 2013.
3. C. Anuradha and T. Velmurugan, ‘‘A comparative analysis on the evaluation of classification algorithms in the prediction of student’s performance,’’ Indian J. Sci. Technol., vol. 8, no. 15, pp. 974–6846, Jan. 2015.

**BOOKS:**

1. "Educational Data Mining: Applications and Trends" edited by Alejandro Peña-Ayala
2. "Data Mining for Business Analytics: Concepts, Techniques, and Applications in Python" by Galit Shmueli, Peter C. Bruce, and Inbal Yahav
3. "Predictive Analytics for Human Resources" by Jac Fitz-enz and John Mattox II

**WEBSITES:**

1. https://www.semanticscholar.org/paper/Using-Data-Mining-Techniques-to-Predict-Student-to-Mengash/f88c9150f378a0fc28357e4daab17a8f48f7eae5
2. https://www.researchgate.net/publication/340046518\_Using\_Data\_Mining\_Techniques\_to\_Predict\_Student\_Performance\_to\_Support\_Decision\_Making\_in\_University\_Admission\_Systems
3. https://www.analyticsvidhya.com/blog/2023/04/student-performance-analysis-and-prediction/