

## Dr. D.Y. Patil School of MCA

***Charoli (BK), PUNE- 412105***

******

## SAVITRIBAI PHULE PUNE UNIVERSITY MASTER OF COMPUTER APPLICATION

Project Report on

## ADMISSION PREDICTION SYSTEM BASED ON GRE AND TOFEL SCORES

BY

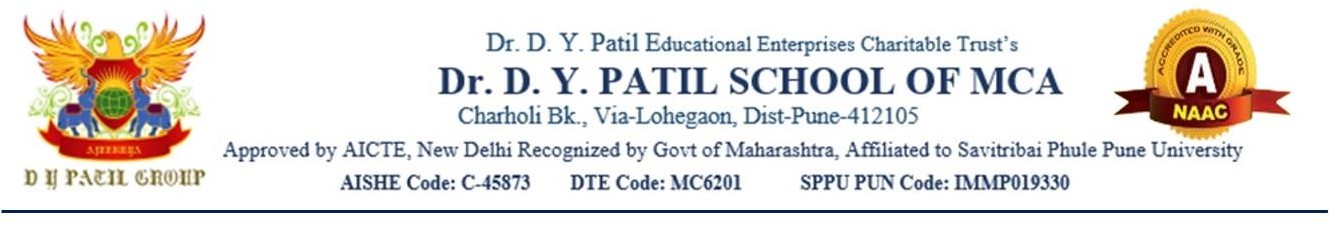
## NIKITA SINHA

**Seat No. :13185**

Under The Guidance Of

## PROF. SHUBHAM WADPALLIWAR

Class: MCA-II(Sem-III) Year: 2024-2025



## Certificate

This is to certify that Mr. /Ms. NIKITA SINHA, has successfully / partially completed his/her project work entitled “**ADMISSION PREDICTION SYSTEM BASED ON GRE AND TOFEL SCORES”** in partial fulfillment of MCA -II SEM-III Mini Project for the year 2024-25.

He / She has worked under our guidance and direction.

|  |  |  |
| --- | --- | --- |
| Prof. Shubham Wadpalliwar | Prof. Sapna Chavan | Dr. E. B. Khedkar |
| Project Guide | HOD | Director |

Examiner 1:

Examiner 2: -----------------

# Index

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Particulars** | **Page**  **Number** |
| **1** | **INTRODUCTION** | **5-8** |
| 1.1 | Company Profile | 5 |
| 1.2 | Abstract | 5-6 |
| 1.3 | Existing System and Need for system | 6-7 |
| 1.4 | Scope of System | 7 |
| 1.5 | Operating Environment - Hardware and Software | 8 |
| 1.6 | Brief Description of Technology Used   * + 1. Operating systems used (Windows or Unix)     2. RDBMS/No SQL used to build database (MySQL/ oracle, Teradata,   etc.) | 8 |
| **2** | **Proposed System** | **9-10** |
| 2.1 | Study of Similar Systems | 9 |
| 2.2 | Feasibility Study | 9 |
| 2.3 | Objectives of Proposed System | 9-10 |
| 2.4 | Users of System | 10 |
| **3** | **Analysis and Design** | **11-18** |
| 3.1 | System Requirements (Functional and Non-Functionalrequirements) | 11 |
| 3.2 | Entity Relationship Diagram (ERD) | 12 |
| 3.3 | Use Case Diagrams | 12 |
| 3.4 | Table Structure | 12-13 |
| 3.5 | Activity Diagram | 13-14 |
| 3.6 | Class Diagram | 15 |
| 3.7 | Module Hierarchy Diagram | 15 |
| 3.8 | Sample Input and Output Screens (Screens must have valid data.All  reports must have at-least 5 valid records.) | 16-18 |
| **4** | **Coding** | **19-24** |

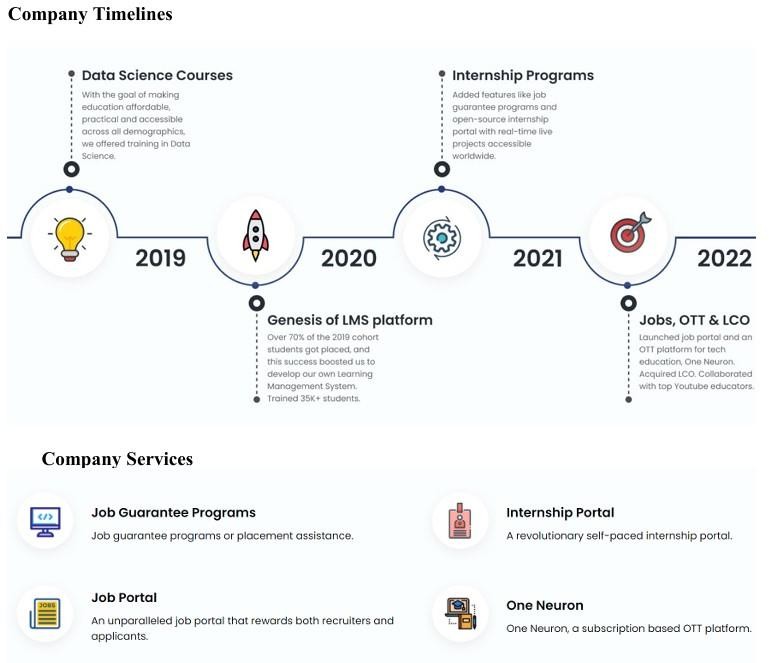
|  |  |  |
| --- | --- | --- |
| 4.1 | Algorithms | 19-20 |
| 4.2 | Code Snippets | 21-24 |
| **5** | **Testing** | **25-29** |
| 5.1 | Test Strategy | 25 |
| 5.2 | Unit Test Plan | 25 |
| 5.3 | Acceptance Test Plan | 25 |
| 5.4 | Test Case / Test Script | 26-28 |
| 5.5 | Defect report / Test Log | 28-29 |
| **6** | **Limitations of Proposed System** | **30** |
| **7** | **Proposed Enhancements** | **31** |
| **8** | **Conclusion** | **32** |
| **9** | **Bibliography** | **33** |
| **10** | **Appendix – Cost sheet, Data sheet** | **34-36** |

1. **INTRODUCTION**

## Company Profile

iNeuron started as a product development company, then launched its ed-tech division. We provide 360-degree solutions from learning to internship to finding a job, and the first ever educational OTT platform to upgrade your skill test.

We at iNeuron are a leading edtech company with our primary focus on providing education on emerging technologies while making tech education easily accessible, practical and affordable. Our 360-degree learning ecosystem ensures that you get all your learning requirements at one place.



**Company Services Mission:** Our goal is to make education and experiential skills affordable and accessible to everyone regardless of their disparate economic and educational backgrounds. We empower students to make demands unlike any other platform or institute because curiosity cannot be contained. Learning cannot be boxed in a book. So, let us step ahead and ‘build together’.

## Abstract

The Admission Prediction System is a machine learning-based solution designed to predict the likelihood of university admission based on key applicant attributes, such as GRE scores, TOEFL scores,

undergraduate GPA (CGPA), and other relevant factors. The system leverages a dataset of historical admission records to identify patterns and build predictive models that assist applicants and admissions committees in evaluating admission chances efficiently.

The project follows a systematic methodology comprising data preprocessing, exploratory data analysis (EDA), feature selection, and the training and evaluation of predictive models. Several machines learning algorithms, including Linear Regression, Decision Tree Regressor, and Random Forest Regressor, are implemented and evaluated using performance metrics like R-square, adjusted R-square, Mean Absolute Error, Mean Square Error, and Root Mean Square Error. Among these models, Linear Regression demonstrated the highest accuracy with an R-square value of 0.8163 and minimal error, making it the most reliable predictor.

This system enhances the transparency and efficiency of the admission process while providing actionable insights into the key factors influencing admission outcomes. Despite its strengths, the system is subject to limitations, such as dependence on historical data and exclusion qualitative attributes like recommendation letters or personal statements. With further development, this system can serve as a valuable tool for both applicants and educational institutions, improving decision making in the competitive admissions landscape.

## Existing System and Need for System Existing System

The current systems for predicting university admissions typically involve a combination of manual and automated processes. These systems often rely on a variety of factors, including standardized test scores (GRE and TOEFL), undergraduate GPA, letters of recommendation, personal statements, and relevant work experience. The admission committees at universities evaluate these criteria to make their decisions.

**University Admission Portals:** Most universities have their own online portals where applicants submit their documents and scores. These portals have basic features for sorting and filtering applications.

**Consultancy Firms:** Many students use educational consultancies that provide personalized guidance on the application process, helping t o optimize their chances of admission based on historical data and expert knowledge.

## Drawbacks Of Existing Systems:

**Lack of Standardization:** Different universities have varying criteria and weightages for GRE, TOEFL scores, and other application components. This lack of standardization makes it difficult to develop a universally accurate prediction model.

**Manual Bias:** The human element in the admission process can introduce biases. Admissions officers might unconsciously favour certain profiles over others, leading to inconsistent decision-making.

**Time-Consuming:** The process of manually reviewing each application is time-consuming and resource-intensive. This can delay decisions and reduce the overall efficiency of the admissions proce

## Need of System

The primary purpose of developing a prediction system for university admissions based on GRE and TOEFL scores is to enhance the efficiency, transparency, and accuracy of the admissions process for both applicants and university administrators. Below are the key reasons highlighting the need for such a system:

**Data-Driven Decision Making:** By leveraging machine learning algorithms, the proposed system can analyse vast amounts of historical admissions data to identify patterns and trends. This data-driven approach helps in making more informed and objective decisions, reducing the reliance on subjective judgment.

**Increased Transparency**: A predictive system can provide clear insights into how different components of an application (such as GRE and TOEFL scores) influence the likelihood of admission. This transparency helps applicants understand their chances better and enables them to improve their profiles accordingly.

**Efficiency and Time-Saving:** Manual review of thousands of applications is a labour-intensive process. An automated prediction system can quickly filter and rank applications based on their likelihood of success, allowing admissions committees to focus their time and efforts on the most promising candidates.

**Fairness and Consistency:** By standardizing the evaluation criteria and minimizing human biases, the system ensures a fairer and more consistent assessment of all applicants. This reduces the potential for favouritism and ensures that all candidates are judged based on the same standards.

**Personalized Feedback for Applicants:** The system can provide personalized feedback to applicants, highlighting their strengths and areas for improvement. This feedback can be invaluable for applicants in refining their applications or enhancing their profiles for future admissions cycles.

## Scope of System

The scope of this project encompasses the development, implementation, and deployment of a predictive system designed to assess the likelihood of university admissions based on key applicant attributes. The system leverages machine learning techniques to provide accurate and actionable predictions, offering significant benefits to both applicants and admissions committees.

The specific areas covered within the scope of the system include: Data Collection and Integration, Data Preprocessing and Analysis, Feature Selection and Engineering, Model Development, Model Optimization and Validation, Prediction and User Interaction.

## Operating Environment - Hardware and Software

* + 1. **HARDWARE SPECIFICATIONS**

**Client Side:** RAM Minimum 8GB and above

HARD DISK Minimum 500 GB and above PROCESSOR Intel I3 and above KEYBOARD

**Server Side:** RAM 16 GB and above

HARDDISK 512 GB and above PROCESSOR Intel I3 processor and above

## SOFTWARE SPECIFICATION

**Client-Side**: Operating System Windows 10 and above

Web Browser Google Chrome/Internet Explorer version 8.0 or higher

**Server-Side:** Operating System Windows 10 and above Database Used: MySQL

Programming Language: HTML, CSS, Python IDE: Visual Studio Code

Libraries used Pandas, Numpy, Seaborn, matplotlib, Sklearn, LinearRegression, train\_test\_split, RandomForestRegressor, DecisionTreeRegressor,

## Brief Description of Technology Used

The project solution proposed to take the required input of user from the created interface and process all the provided data to meet the requirements of the machine learning model and finally display the output saying so and so amount is the predicted cost.

The Technology used in the project is Data Science (Machine Learning Algorithms) to predict the fare flight based on the historical data.

Python 3.9 is used as the programming language and frame works like Numpy, pandas, Sklearn and other modules for building the model. For visualizations seaborn and parts of matplotlib are being used

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

# Proposed System

## Study of Similar Systems

These System have employed machine learning techniques to predict admission outcome. These systems provide valuable insights into the effectiveness of various approaches and highlight best practices for handling complex datasets in the domain of university admissions.

University Admission Predictors: Many predictive models for university admissions use datasets with applicant-specific attributes similar to those in this project. These systems often leverage machine learning algorithms such as Linear Regression, Decision Trees, and Support Vector Machines (SVMs). Machine Learning Techniques in Admission Systems: Ideal for initial benchmarking due to simplicity and interpretability. Decision Trees and Random Forests provide higher accuracy and resilience to outliers but may require careful tuning to avoid overfitting.

## Feasibility Study

The goal of this project is to develop a predictive model for university admissions based on key applicant attributes, such as GRE and TOEFL scores, undergraduate GPA, and other relevant factors. The system leverages machine learning techniques to provide accurate predictions of admission chances, enhancing both the efficiency and fairness of the admissions process.

## Methodology

The project utilizes a dataset from Kaggle containing 400 records with attributes such as GRE Score, TOEFL Score, University Rating, SOP, LOR, CGPA and Chance of Admit. The methodology involves several key steps:

**Data Cleaning and Preprocessing**: Handling missing values, converting datatypes and standardizing attributes to ensure a clean and usable dataset.

**Exploratory Data Analysis (EDA):** Analysing data distribution and relationships between attributes to gain insights into the factors influencing admissions.

**Feature Selection and Engineering:** Identifying and selecting the most relevant features for the predictive model.

**Model Training and Evaluation:** Using the Linear Regression, Ridge Regression, Decision Tree Regressor and Random Forest Regressor Model to train the model and evaluate its performance using metrics such as R-square and adjusted R-square.

**Model Optimization:** Tuning hyperparameters to improve the model's accuracy.

## Objectives of Proposed System

The proposed system aims to enhance the university admission prediction process by leveraging machine learning techniques to provide accurate, efficient, and fair assessments of applicants based on their GRE and TOEFL scores, along with other relevant attributes.

The primary objectives of the proposed system are:

**Accurate Prediction of Admission Chances**: To develop a machine learning model that accurately predicts the likelihood of admission to universities based on GRE and TOEFL scores, as well as other factors such as CGPA, university rating, SOP and LOR and to achieve a high level of prediction accuracy, with an aim to exceed 80% accuracy as indicated by R-square and adjusted R-square values.

**Streamlining the Admissions Process:** To create an automated system that reduces the time and effort required for university admissions committees to evaluate applications and to enable quick filtering and ranking of applicants, allowing committees to focus on the most promising candidates.

**Providing Transparency and Insight to Applicants:** (GRE, TOEFL scores) This impacts their chances of admission and to provide personalized feedback to applicants, helping them understand their strengths and areas for improvement.

**Ensuring Fairness and Consistency:** To minimize human biases in the admissions process by using standardized, data-driven evaluation criteria and to ensure that all applicants are assessed based on the same objective standards, thereby promoting fairness and consistency.

**Enhancing Decision-Making for Universities:** To assist universities in making more informed decisions by providing data- driven insights into applicant profiles and to enable strategic planning by predicting enrolment trends and managing resources accordingly.

## Users of System

The university admissions prediction system is designed to serve a diverse group of stakeholders involved in the admissions process. The Admission System is designed for the following types of users:

## Students

**Role:** Primary users.

## Responsibilities:

Enter academic details (GRE, TOEFL, CGPA, etc.). View admission predictions.

Apply to universities and check application status.

**University Administrators Role:** Secondary users.

## Responsibilities:

Review and process student applications.

Update application status (e.g., Accepted/Rejected).

**System Administrators Role:** Support users.

## Responsibilities:

Manage the system.

Resolve technical issues and maintain performance.

# Analysis and Design

## System Requirements (Functional and Non-Functional requirements) Functional Requirements

**Data Collection and Loading:**

The system shall load the dataset from a specified source (e.g., Kaggle).

The system shall verify the presence of required attributes: GRE Score, TOEFL Score, University Rating, SOP, LOR, CGPA, Research, and Chance of Admit.

## Data Cleaning and Preprocessing:

The system shall replace special characters in the dataset with null values. The system shall convert categorical attributes to appropriate data types.

The system shall handle missing values by imputing them with mean values for numerical attributes and mode values for categorical attributes.

The system shall standardize the dataset to ensure uniform data distribution.

## Exploratory Data Analysis (EDA):

The system shall generate statistical summaries for each attribute in the dataset.

The system shall create visualizations such as histograms, scatter plots, and box plots to depict data distributions and relationships.

The system shall generate a correlation heatmap to illustrate the relationships between different attributes.

## Feature Selection and Engineering

The system shall identify and select relevant features for the predictive model.

The system shall engineer new features if necessary to improve model performance.

The system shall evaluate the importance of each attribute in predicting admission chances.

## Model Training and Testing

The system shall split the dataset into training and testing subsets using stratified sampling. The system shall train the Linear Regression, Ridge Regression, Support Vector Regressor (SVR) algorithm, Decision Tree Regressor, on the training dataset.

The system shall evaluate the model's performance using the testing dataset and report metrics such as accuracy, R-square, and adjusted R-square.

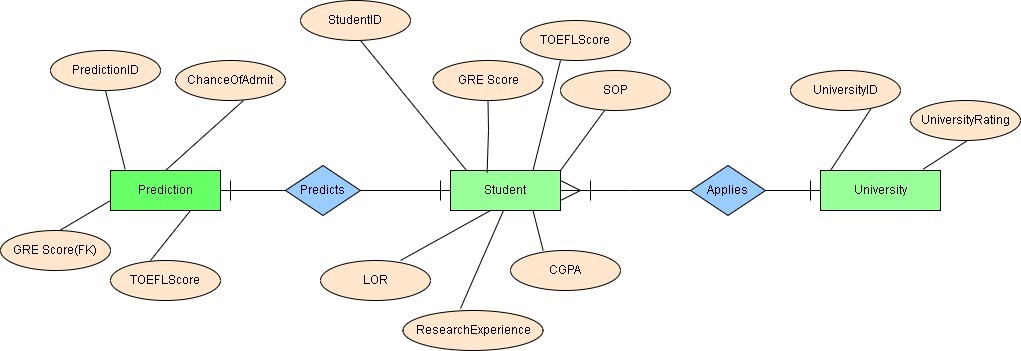
## Non-Functional Requirements

**Performance:** The system shall provide admission predictions within few seconds for a given set of input data.

**Reliability:** The system shall have an uptime of 99.9%, ensuring high availability. **Efficiency:** The system shall optimize resource usage to ensure cost-effective operation, minimizing the use of computational power and storage.

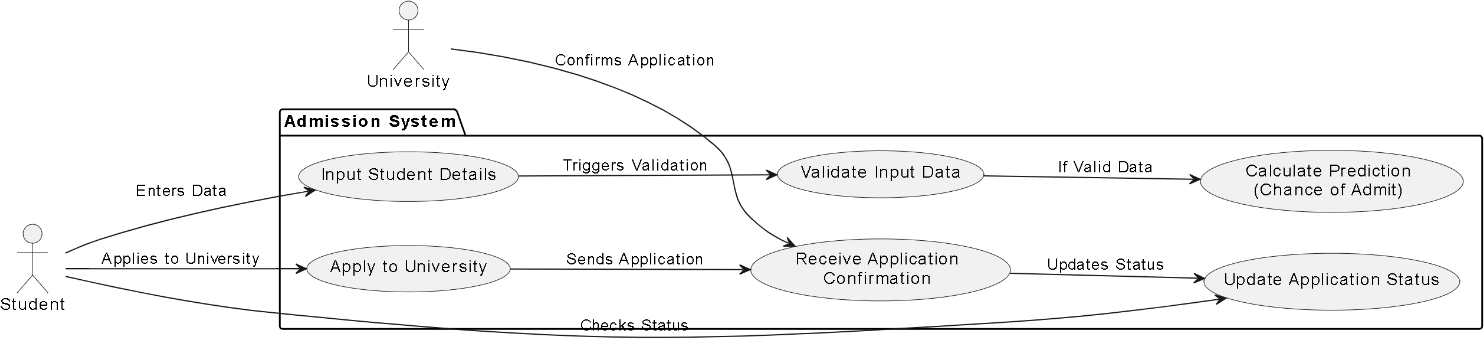
**Interoperability:** The system shall be capable of integrating with other systems and services via APIs to allow for data exchange and interoperability.

## Entity Relationship Diagram (ERD)

****

* 1. **Use Case Diagrams**

A Use Case Diagram for the Admission Prediction System based on GRE and TOEFL scores will illustrate the interaction between the system and its applicants.



* 1. **Table Structure**

**Table: Students (Applicants)**

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| Student\_id | INT | PRIMARY KEY, AUTO\_INCREMENT | Unique Identifier for Each Student |
| GRE Score | INT | CHECK (GRE Score BETWEEN 260 AND 340) | GRE (Graduate Record Examination) Score, Ranging From 260 To 340. |
| TOEFL Score | INT | CHECK (TOEFL Score BETWEEN 0 AND 120) | TOEFL (Test of English as a Foreign Language) score, ranging from 0 to 120 |
| SOP | FLOAT | CHECK (SOP BETWEEN 1.0 AND 5.0) | Strength of the Statement of Purpose, rated on a scale of 1 to 5 |

|  |  |  |  |
| --- | --- | --- | --- |
| LOR | FLOAT | CHECK (LOR BETWEEN 1.0 AND 5.0) | Strength of the Letter of Recommendation, rated on a scale of 1 to 5. |
| CGPA | FLOAT | CHECK (CGPA BETWEEN 0 AND 10.0) | Cumulative Grade Point Average, typically ranging from 0 to 10 |
| Research Experience | BOOLEAN | CHECK (Research Experience IN (0, 1)) | Binary value indicating research experience: 0 = No, 1 = Yes |

**Table: University**

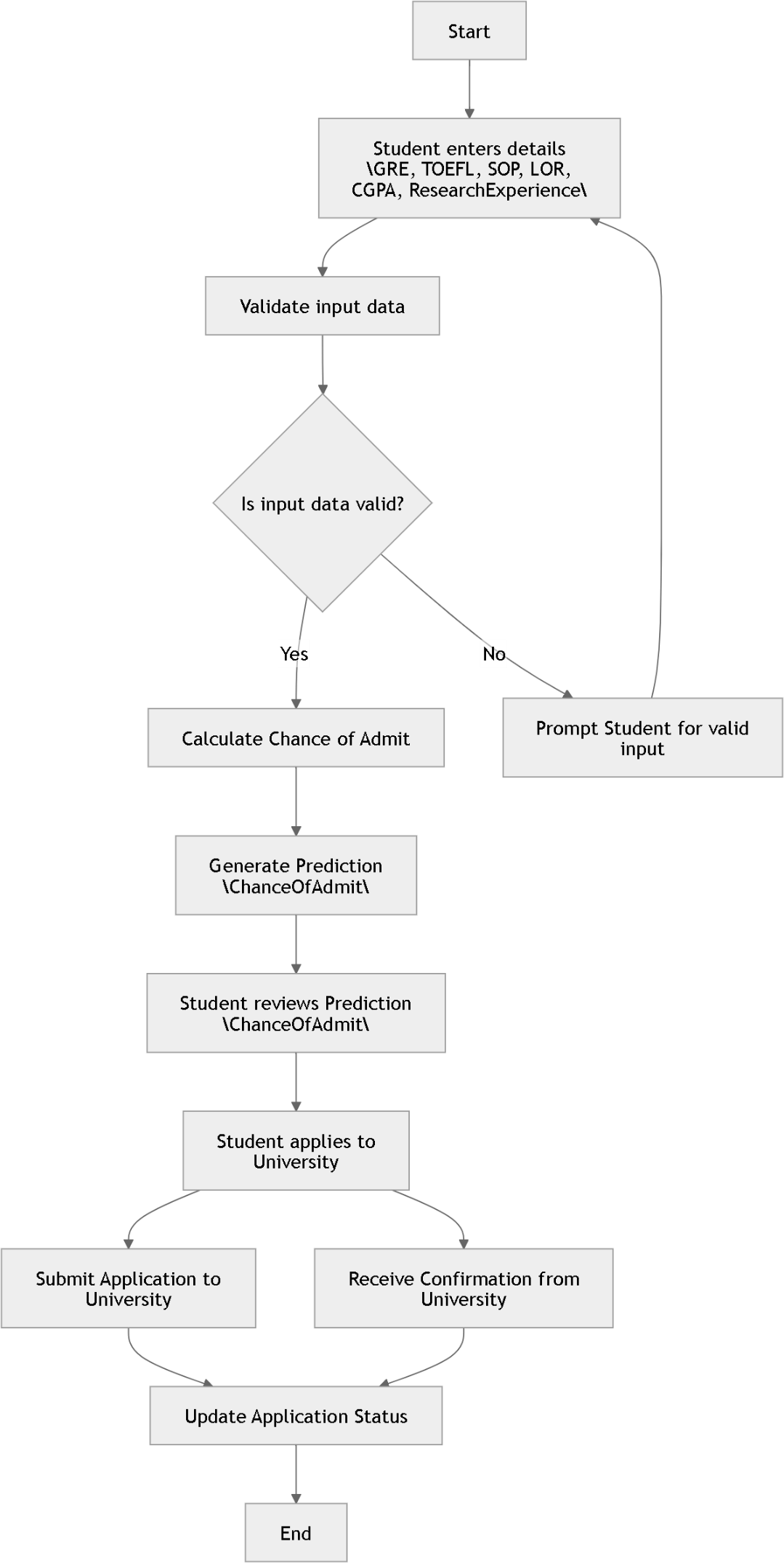
|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| University Id | INT | PRIMARY KEY, AUTO\_INCREMENT | Unique Identifier for each University |
| University Rating | INT | NOT NULL | University Rating (1-5) |

**Table: Prediction**

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| prediction\_id | INT | PRIMARY KEY, AUTO\_INCREMENT | Unique identifier for each prediction |
| student\_id | INT | FOREIGN KEY REFERENCES  ApplicantData(applicant\_id) | Link to the student data table |
| chance\_of\_admit | DECIMAL (3, 2) | NOT NULL | Predicted chance of admission (0.00-1.00) |
| GRE Score | INT | FOREIGN KEY | GRE Score, Ranging From 260 To 340. |
| TOEFL Score | INT | FOREIGN KEY | TOEFL Score, ranging from 0 to 120 |

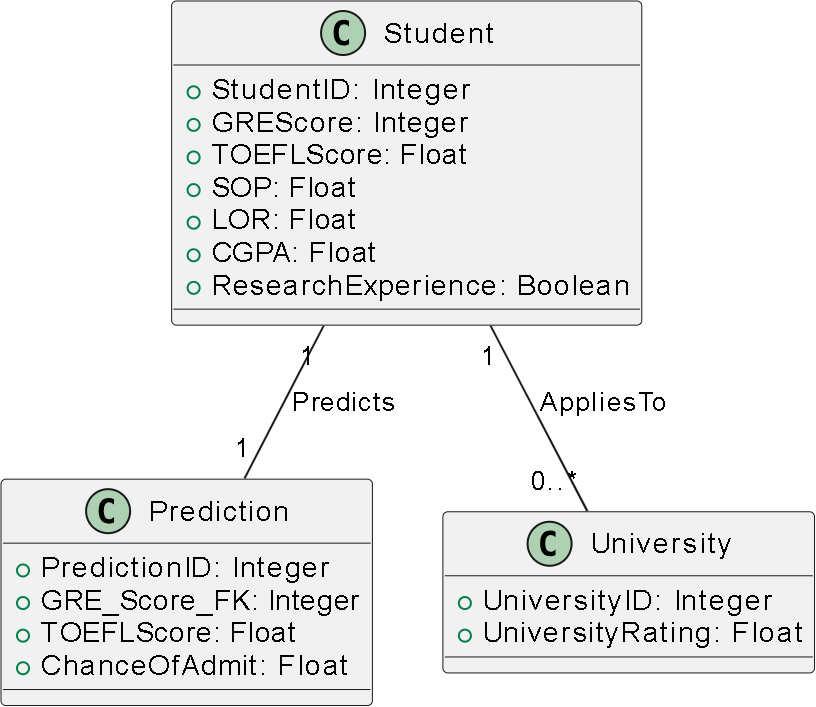
* 1. **Activity Diagram**

The activity diagram for the Admission Prediction System will showcase the flow of activities performed by both the Admin and the Students as well as the system's internal processes.



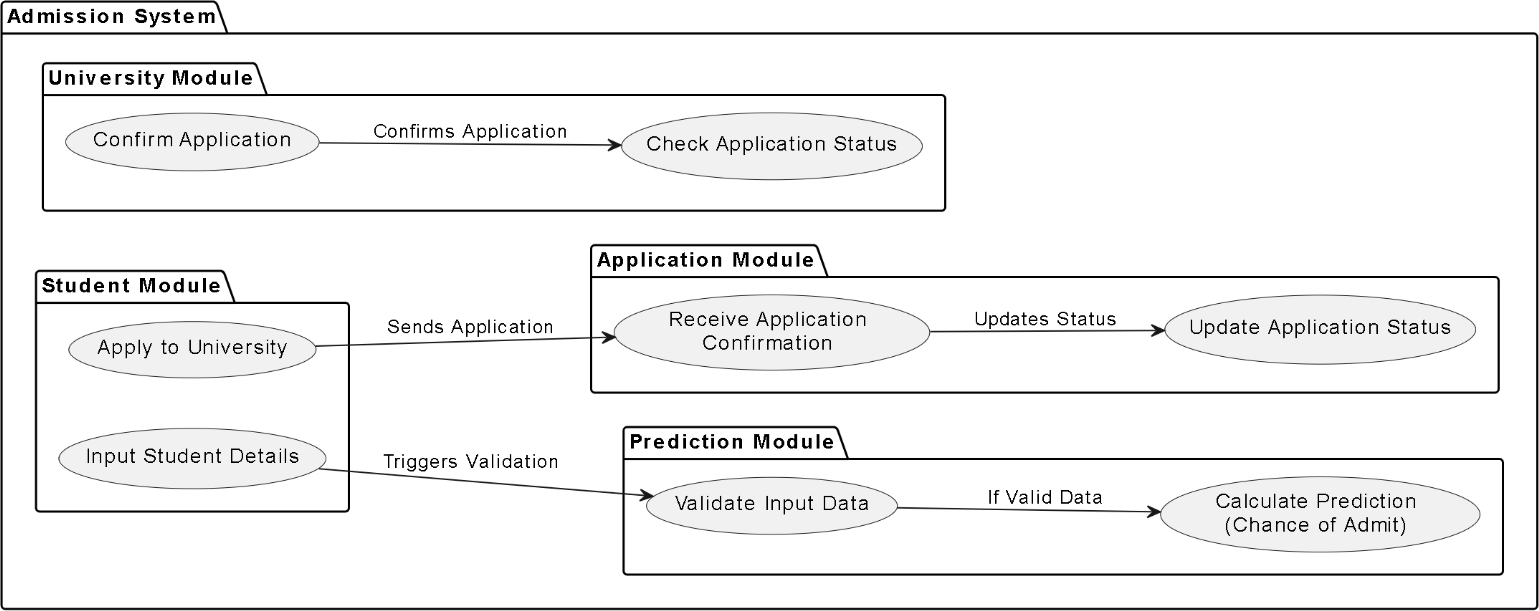
## Class Diagram

The Class Diagram for the Admission Prediction System visually represents the structure of the system. It highlights the classes, their attributes, methods, and the relationships between them.



## Module Hierarchy diagram

A Model Hierarchy Diagram shows the relationships and dependencies between different machine learning models used in the system.



## Sample Input and Output Screens

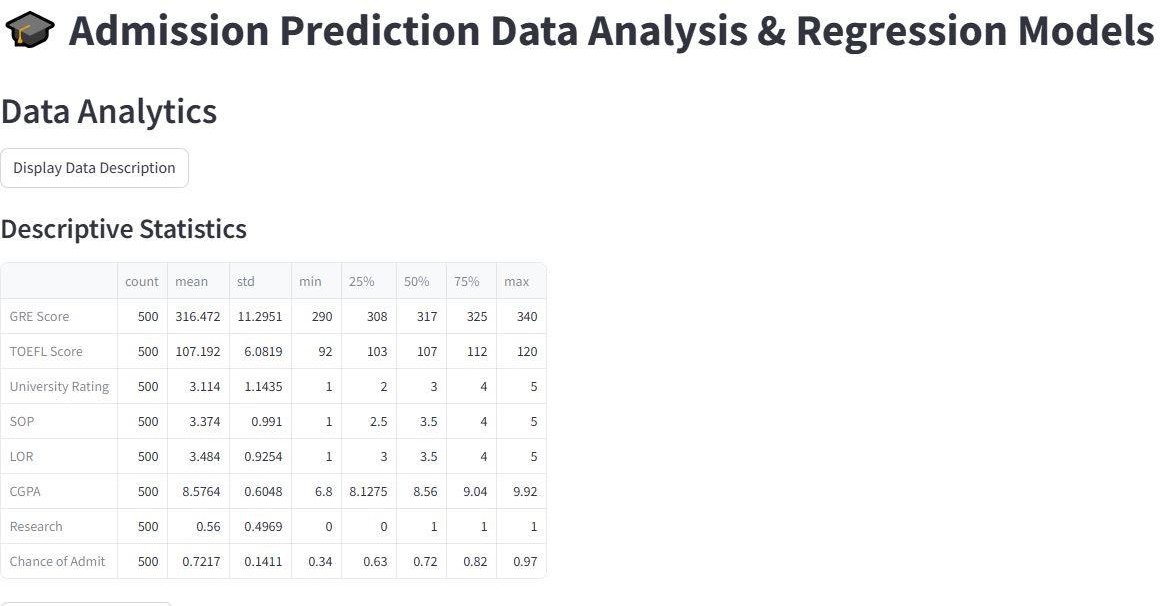
****

Fig: Showing Descriptive Statistics.

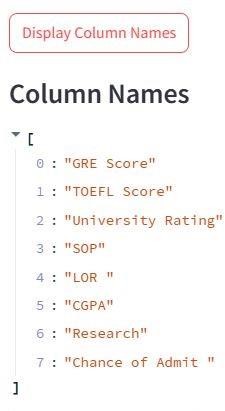


Fig: Showing Column Names.

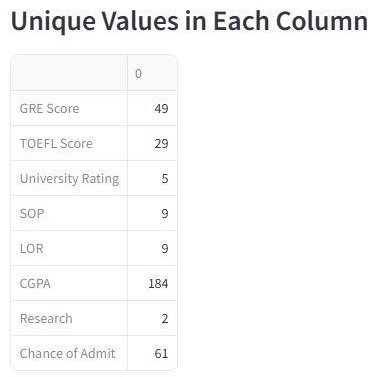


Fig: Showing Unique Values in each column.

## Heatmap

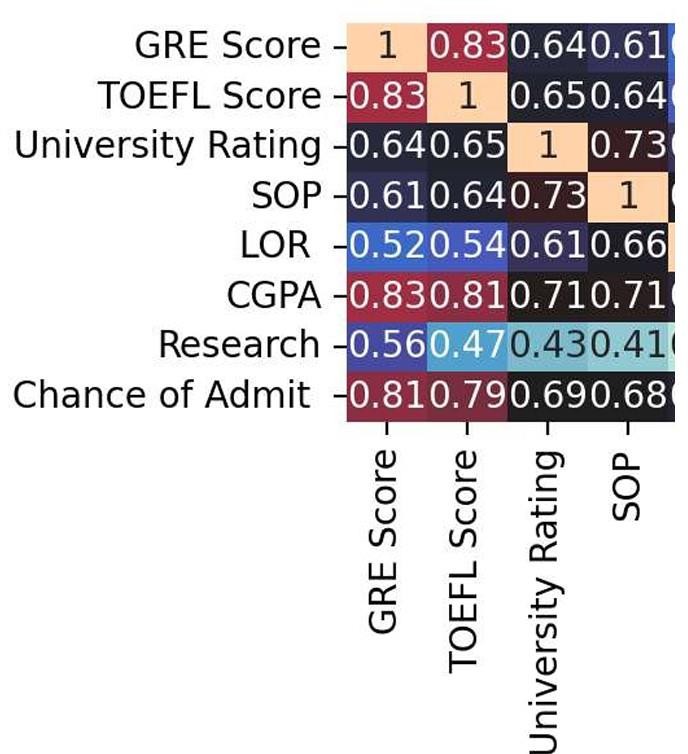
****



Fig: Showing Model Evaluation Summary

# 14. Coding

## Algorithms

* + 1. **Introduction to Project**

**Overview**: The Admission Prediction System leverages machine learning to estimate the

likelihood of a candidate’s acceptance into a university based on key attributes such as GRE and TOEFL scores, undergraduate GPA, and other factors. The system enhances decision-making for applicants and institutions by providing data-driven insights.

**Phases**: The project is divided into distinct phases, including data collection, exploratory analysis, model building, advanced machine learning techniques, deployment, and monitoring.

**Tools**: The project employs a variety of tools, including Python libraries like pandas, seaborn, scikit-learn, and streamlit, along with machine learning algorithms such as Linear Regression, Decision Tree Regressor, and Random Forest Regressor.

## Data Collection & Preparation

**Sourcing**: The data is sourced from a publicly available dataset,"admission\_predict\_system.csv," with attributes such as GRE Score, TOEFL Score, SOP, LOR, CGPA, and Chance of Admit.

**Cleaning**: Steps include handling missing values, dropping irrelevant columns (e.g., Serial No.), and ensuring consistent datatypes.

**Feature Selection**: Independent variables like GRE Score and CGPA are selected as predictors, while the dependent variable is Chance of Admit.

**Scaling**: Standardization of numerical features is performed using StandardScaler to ensure compatibility across machine learning models.

## Exploratory Data Analysis (EDA)

**Statistics:** Descriptive statistics such as mean, median, and standard deviation provide a quantitative understanding of the dataset.

**Visualizations:** Histograms and KDE plots help identify the distribution of variables. A correlation heatmap reveals relationships among features.

**Correlation:** Significant positive correlations are observed between features like CGPA and GRE Score with the target variable Chance of Admit.

**Outliers:** Visualizations like box plots are used to detect outliers, which may influence the model's predictions.

* + 1. **Model Building**

Three primary models are implemented:

**Linear Regression**: Captures linear relationships between features and the target.

**Decision Tree Regressor**: Captures non-linear relationships but is prone to overfitting. **Random Forest Regressor**: An ensemble method that balances bias and variance for improved accuracy.

**Training and Validation**: The dataset is split into training (75%) and test sets (25%) to evaluate the models. Performance metrics such as R-square, Adjusted R-square, Mean Absolute Error,

and Root Mean Square Error are computed.

## Advanced Machine Learning

**Ensembles**: Random Forest is used as an ensemble method, combining multiple decision trees to improve prediction accuracy.

**Hyperparameter Tuning**: GridSearchCV is planned for fine-tuning models, including parameters like tree depth and the number of estimators for Random Forest.

**Deep Learning**: This phase can include experimentation with deep neural networks for further improvements.

## Project Execution

**Case Studies**: Real-world scenarios simulate user interaction, such as predicting admission chances for specific applicants.

**Challenges**: Challenges include handling data bias, computational constraints during hyperparameter tuning, and ensuring predictions are interpretable.

**Success Metrics**: The project's success is measured by high R-square values and low error rates across models.

## Deployment & Monitoring

**Techniques**: The model is deployed using **Streamlit**, providing an interactive web-based interface

## Documentation

**Reporting**: Results are summarized in easy-to-read tables and charts for stakeholders.

**Storytelling**: Visuals like heatmaps and model performance comparisons effectively communicate the insights derived from the data.

**Transparency**: A comprehensive report explains the methodology, assumptions, and limitations of the model.

## Ethics & Best Practices

**Privacy**: Sensitive data, such as applicant scores, is securely handled to comply with data protection standards.

**Fairness**: The model is tested for bias to ensure fairness across demographic groups.

**Reproducibility**: Clear documentation of the code and methods ensures the project can be replicated by others.

**Explainability**: The inclusion of interpretable models ensures stakeholders understand the rationale behind predictions.

## Code snippets

**Imported Libraries** import pandas as pd import seaborn as sns

import matplotlib.pyplot as plt import warnings

import streamlit as st

from sklearn.linear\_model import LinearRegression from sklearn.tree import DecisionTreeRegressor

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import r2\_score, mean\_squared\_error, mean\_absolute\_error from sklearn.model\_selection import train\_test\_split, GridSearchCV

from sklearn.preprocessing import PolynomialFeatures, StandardScaler import numpy as np

## Function to load and process data

def load\_data():

data = pd.read\_csv("admission\_predict\_system.csv") data.drop('Serial No.', axis=1, inplace=True)

return data

## Data Description

if st.button("Display Data Description", key='desc', help="Show descriptive statistics."): st.subheader("Descriptive Statistics")

st.dataframe(data.describe().T)

## Unique Values

if st.button("Display Unique Values", key='unique', help="Show unique values in each column."):

st.subheader("Unique Values in Each Column") st.dataframe(data.nunique())

## Column Names

if st.button("Display Column Names", key='columns', help="Show column names."): st.subheader("Column Names")

st.write(data.columns.tolist())

## Data Distribution for Each Column

if st.button("Display Data Distribution", key='distribution', help="Show data distribution for each column."):

st.subheader("Data Distribution for Each Column") fig, axes = plt.subplots(5, 2, figsize=(12, 25)) columnnumber = 1

for column in data.columns:

if columnnumber <= 10:

sns.histplot(data[column], kde=True, ax=axes[(columnnumber-1)//2, (columnnumber-1)%2]) axes[(columnnumber-1)//2, (columnnumber-1)%2].set\_xlabel(column, fontsize=20) columnnumber += 1

st.pyplot(fig)

## Correlation Heatmap

if st.button("Display Correlation Heatmap", key='heatmap', help="Show correlation heatmap of the data."):

st.subheader("Correlation Heatmap") fig, ax = plt.subplots(figsize=(12, 10))

sns.heatmap(data.corr(), annot=True, cmap='icefire', ax=ax) st.pyplot(fig)

**Independent and Dependent Features** X = data.drop('Chance of Admit ', axis=1) y = data['Chance of Admit ']

## Splitting the dataset into training set and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42)

## Standardization of data

scaler = StandardScaler()

## Scaling of train and test data

X\_train = scaler.fit\_transform(X\_train) X\_test = scaler.transform(X\_test)

## Define and evaluate models in a reusable function

def evaluate\_model(model, model\_name): model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test) r2 = r2\_score(y\_test, y\_pred)

adj\_r2 = 1 - (1 - r2) \* (len(y\_test) - 1) / (len(y\_test) - X\_test.shape[1] - 1) mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred) rmse = np.sqrt(mse)

return {

'name': model\_name, 'r2': r2,

'adj\_r2': adj\_r2, 'mae': mae,

'mse': mse,

'rmse': rmse}

## Evaluate models without hyperparameter tuning

models = [

(LinearRegression(), "Linear Regression"), (DecisionTreeRegressor(random\_state=0), "Decision Tree Regressor"), (RandomForestRegressor(), "Random Forest Regressor"]

results = []

for model, name in models: results.append(evaluate\_model(model, name))

## Create DataFrame for all models' metrics

models\_df = pd.DataFrame({

'Model Name': [result['name'] for result in results],

'R Square Value': [round(result['r2'], 4) for result in results],

'Adjusted R square value': [round(result['adj\_r2'], 4) for result in results], 'Mean Absolute Error': [round(result['mae'], 2) for result in results], 'Mean Square Error': [round(result['mse'], 2) for result in results],

'Root Mean Square Error': [round(result['rmse'], 2) for result in results]

})

## Displaying the Model Evaluation Summary

if st.button("Display Model Evaluation Summary", key='eval\_summary', help="Show model evaluation metrics."):

st.subheader("Model Evaluation Metrics") st.dataframe(models\_df)

## Streamlit App

st.set\_page\_config(page\_title="Admission Prediction", page\_icon="\*◆ ", layout="wide")

## Custom CSS for styling

st.markdown( """

<style>

.main {

background-color: #f0f2f5;}

.header {

text-align: center; color: #2c3e50; font-size: 2.5em; font-weight: bold; }

.subheader { color: #2980b9; font-size: 1.5em;

font-weight: bold; }

.stButton>button { background-color: #3498db; color: white;

border-radius: 5px; height: 3em;

width: 10em; font-size: 1em; }

.stButton>button:hover { background-color: #2980b9; }

</style>

""", unsafe\_allow\_html=True)

st.title("◆ \* Admission Prediction Data Analysis & Regression Models")

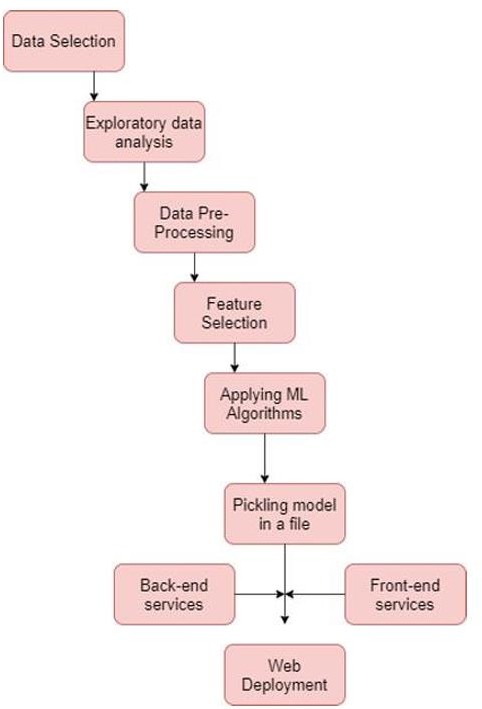
## Displaying the results

if st.button("Display Linear Regression Results with Polynomial Features", key='poly\_results', help="Show results after polynomial feature tuning."):

st.subheader("Linear Regression with Polynomial Features") st.write(f"Degree of Polynomial: {degree}")

st.write(f"R Square: {round(poly\_r2, 4)}") st.write(f"Adjusted R Square: {round(poly\_adj\_r2, 4)}") st.write(f"Mean Absolute Error: {round(poly\_mae, 2)}") st.write(f"Mean Square Error: {round(poly\_mse, 2)}") st.write(f"Root Mean Square Error: {round(poly\_rmse, 2)}")

* 1. **Methodology**

****

# Testing

## Test Strategy

A comprehensive test strategy ensures that the university admissions prediction system operates as intended, delivering accurate, reliable, and user-friendly outputs. The testing process involves multiple stages, addressing both functional and non-functional requirements, and leveraging a combination of manual and automated testing approaches.

## Testing Objectives

* + - Validate the accuracy and robustness of predictive models.
    - Ensure data integrity and correctness in preprocessing and feature engineering steps.
    - Verify the system's ability to handle edge cases and diverse user inputs.
    - Test the performance, scalability, and usability of the system.

## Unit Test Plan

Focus: Validate individual components of the system, such as preprocessing scripts, model training pipelines, and utility functions.

Tools: Python testing frameworks like unit test or pytest.

Ensuring that missing values are handled correctly in the preprocessing step and that the model outputs valid probability scores.

## Acceptance Test Plan

**Focus**: Validate that the system meets the expectations of end-users.

**Process**: Conduct sessions with applicants, admissions officers, and consultants to collect feedback on usability and accuracy. Ensuring that predictions align with expert evaluations .

## Integration Testing

Focus: Test interactions between system components, such as the data pipeline, model inference, and user interface.

Tools: Postman (for APIs), Selenium (for UI components).

Verify that the trained model integrates seamlessly with the web application to provide predictions in real-time.

## Regression Testing

Focus: Ensure that updates or changes to the system do not introduce new defects. Approach: Maintain a suite of test cases to re-run after each system update.

After hyperparameter tuning, check that the adjusted model does not negatively affect existing prediction accuracy.

## Functional Testing

Focus: Validate system features against functional requirements. Techniques: Black-box testing.

Ensuring that inputting applicant data (e.g., GRE, TOEFL, CGPA) generates a correct prediction of the chance of admission.

## Performance Testing

Focus: Assess the system's responsiveness and stability under various conditions. Techniques: Load testing and stress testing.

Verify that the system can handle simultaneous predictions for 100+ users without significant delays.

* 1. **Test Case**

**Test Cases for University Admissions Prediction System**

1. **Preprocessing Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Input** | **Expected Output** | **Pass/Fail Criteria** |
| TC\_01\_Preprocess | Verify missing value handling. | Dataset with missing GRE scores | Missing values replaced with mean/median or flagged. | Missing values handled correctly |
| TC\_02\_Preprocess | Check data standardization | Dataset with unscaled CGPA values. | CGPA values scaled to 0-1 range | Output matches scaling  function |
| TC\_03\_Preprocess | Test datatype conversion | String CGPA in dataset. | CGPA  converted to  float. | Datatypes match |

1. **Model Training Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Input** | **Expected Output** | **Pass/Fail Criteria** |
| TC\_01\_ModelTrain | Verify model training process. | Training dataset (cleaned). | Model trained successfully without errors | Training completes without  exceptions |
| TC\_02\_ModelTrain | Validate feature importance ranking | Dataset with all attributes | Model provides ranking of features | Feature ranking aligns with domain knowledge |
| TC\_03\_ModelTrain | Test performance metrics during training | Training dataset | R-squared > 0.8, MSE  within acceptable  limits | Metrics exceed threshold |

1. **Prediction Functionality Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Input** | **Expected Output** | **Pass/Fail Criteria** |
| TC\_01\_Predict | Verify prediction output format | GRE: 320,  TOEFL: 110,  CGPA: 8.5 | Predicted chance of admit (0-1) | Output is a valid probability score |
| TC\_02\_Predict | Test prediction with edge values | GRE: 0,  TOEFL: 0,  CGPA: 0 | Low chance of admit | Prediction aligns with input logic |
| TC\_03\_Predict | Test prediction for high-performing candidate | GRE: 340,  TOEFL: 120,  CGPA: 10 | High chance of admit (~1) | Prediction aligns with input logic |

1. **Edge Case Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Input** | **Expected Output** | **Pass/Fail Criteria** |
| TC\_01\_Edge | Test with missing or null fields | GRE: 320,  TOEFL: null, CGPA: 9.0. | Prediction still generated (handling missing fields) | System does not crash, and output is valid |
| TC\_02\_Edge | Test with extreme high  values | GRE: 400,  TOEFL: 200,  CGPA: 15.0 | Error or capped prediction  output | Output within logical bounds |

1. **Usability Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Input** | **Expected Output** | **Pass/Fail Criteria** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC\_01\_Usability | Validate ease of navigation | User navigates through the system | Smooth navigation without confusion | Feedback from users is positive |
| TC\_02\_Usability | Test clarity of prediction results | Prediction request submitted | Prediction displayed clearly with  explanation | Users understand the result |

* 1. **Defect report**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Defec** | **Defect** | **Module/F** | **Seve** | **Prior** | **Steps to** | **Expect** | **Actual** | **Status** | **Assign** |
| **t ID** | **Descripti** | **eature** | **rity** | **ity** | **Reproduce** | **ed** | **Behavio** |  | **ed To** |
|  | **on** |  |  |  |  | **Behavi** | **r** |  |  |
|  |  |  |  |  |  | **or** |  |  |  |
| D001 | Incorrect | Data | High | High | 1. Input | Missin | Missing | Open | Niks |
|  | handling | Preproces |  |  | dataset | g GRE | GRE |  |  |
|  | of missing | sing |  |  | with | scores | scores |  |  |
|  | GRE |  |  |  | missing | replac | remain |  |  |
|  | scores |  |  |  | GRE scores. | ed | null, |  |  |
|  |  |  |  |  | 2. Run | with | causing |  |  |
|  |  |  |  |  | preprocessi | mean/ | training |  |  |
|  |  |  |  |  | ng script | media | failure |  |  |
|  |  |  |  |  |  | n |  |  |  |
|  |  |  |  |  |  | value |  |  |  |
| D002 | Incorrect | Data | High | High | 1. Input | CGPA | CGPA | Open | Niks |
|  | scaling of | Preproces |  |  | dataset | scaled | scaled |  |  |
|  | CGPA | sing |  |  | with CGPA | to 0-1 | incorrec |  |  |
|  | values |  |  |  | ranging | range | tly, |  |  |
|  |  |  |  |  | from 6 to |  | resultin |  |  |
|  |  |  |  |  | 10. |  | g in |  |  |
|  |  |  |  |  | 2. Run |  | values > |  |  |
|  |  |  |  |  | preprocessi |  | 1 |  |  |
|  |  |  |  |  | ng |  |  |  |  |
| D003 | Model | Model | High | High | 1. Train | R- | R- | Open | Rits |
|  | performa | Training |  |  | model on | square | squared |  |  |
|  | nce below |  |  |  | provided | d > | = 0.65, |  |  |
|  | acceptabl |  |  |  | dataset. | 0.8, | MSE |  |  |
|  | e |  |  |  | 2. Evaluate | MSE | higher |  |  |
|  | threshold |  |  |  | metrics (R- | within | than |  |  |
|  |  |  |  |  | squared, | accept | accepta |  |  |
|  |  |  |  |  | MSE) | able | ble |  |  |
|  |  |  |  |  |  | limits |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D004 | Prediction | Prediction | Critic | High | 1. Input | Syste | Predicti | Open | Rits |
|  | s not | Model | al |  | GRE: 0, | m | on fails |  |  |
|  | generate |  |  |  | TOEFL: 0, | genera | with |  |  |
|  | d for edge |  |  |  | CGPA: 0. | tes | error |  |  |
|  | cases |  |  |  | 2. Submit | valid | message |  |  |
|  |  |  |  |  | request for | predic |  |  |  |
|  |  |  |  |  | prediction | tion |  |  |  |
|  |  |  |  |  |  | even |  |  |  |
|  |  |  |  |  |  | for |  |  |  |
|  |  |  |  |  |  | edge |  |  |  |
|  |  |  |  |  |  | inputs |  |  |  |

# Limitations of Proposed System

## Dependence on Historical Data

The system relies heavily on the quality and representativeness of the training dataset. If the dataset contains biases or does not reflect current admission trends, predictions may be inaccurate. Overrepresentation of candidates from specific regions or educational systems may skew the model's predictions.

## Limited Scope of Features

While the system considers important factors like GRE, TOEFL, CGPA, and others, it does not account for subjective or qualitative factors such as: Extracurricular achievements, personal statements, or recommendation letters. Institutional preferences, such as a focus on diversity or specific disciplines.

## Potential for Overfitting

If the model is too complex or tuned excessively to the training data, it may perform well on historical data but poorly on new or unseen cases, limiting its generalizability.

## Lack of Contextual Interpretation

The system provides probabilities for admission but lacks the ability to explain the broader context or rationale behind predictions. This can make it challenging for users to: Understand why they received a particular prediction.

Identify actionable steps to improve their chances of admission beyond numerical improvements.

## Sensitivity to Input Data Quality

Errors or inconsistencies in user-provided input data (e.g., incorrect GRE or TOEFL scores) can significantly impact prediction accuracy. Additionally:

Missing or incomplete data may lead to unreliable predictions.

Users may intentionally or unintentionally input extreme or fabricated values.

# Proposed Enhancements

The prediction system for university admissions using GRE and TOEFL scores can be expanded and enhanced in numerous ways to increase its utility, accuracy, and user-friendliness. The following points outline potential future developments and enhancements:

## Advanced Machine Learning Techniques:

Deep Learning Models: Implement deep learning techniques such as neural networks to capture complex relationships in the data.

Ensemble Methods: Use ensemble methods like Random Forests, Gradient Boosting Machines (GBM), or XGBoost to enhance prediction accuracy.

**Personalized Recommendations**: University and Program Recommendations: Provide personalized recommendations for universities and programs based on the applicant’s profile and historical admission data. Improvement Suggestions: Offer actionable advice to applicants on how to improve the chances of admission, such as retaking exams or enhancing specific skills.

Integration with Other Systems: Educational Platforms: Integrate with educational platforms like Coursera, edX, and LinkedIn Learning to consider completed courses and certifications. Application Portals: Link with university application portals to streamline the application process and provide seamless submission of predictions and recommendations.

**Scalability and Performance Enhancements:** Cloud Scalability: Utilize cloud infrastructure to

handle increased user load and large-scale data processes. Real-Time Predictions: Implement real- time prediction capabilities to provide immediate feedback to applicants.

**Security and Compliance:** Data Privacy: Enhance data privacy measures to comply with regulations like GDPR and CCPA. Secure Authentication: Implement advanced authentication mechanisms like biometric authentication or multi-factor authentication.

**Research and Development Continuous Learning:** Develop mechanisms for the model to continuously learn and adapt from new data. Collaborative Research: Collaborate with educational institutions and research organizations to continuously improve the prediction model and validate its accuracy.

# Conclusion

The proposed university admissions prediction system represents a significant step forward in leveraging machine learning to enhance the efficiency, accuracy, and fairness of the admissions process. By utilizing key applicant attributes such as GRE and TOEFL scores, CGPA, and other factors, the system provides data-driven predictions that can support applicants, admissions officers, and other stakeholders in making informed decisions.

Through careful implementation, including data preprocessing, exploratory data analysis, feature selection, and rigorous model evaluation, the system demonstrates the potential to streamline admissions workflows and offer valuable insights into factors influencing admission outcomes.

However, the system's limitations—such as dependence on historical data, exclusion of qualitative factors, and sensitivity to input quality—highlight the need for ongoing refinement and the integration of complementary tools to address these gaps.

In conclusion, while the system is not a substitute for comprehensive human evaluation, it servesas a powerful supplement that can improve transparency and efficiency. With further enhancements and ethical safeguards, this predictive model can become a valuable tool in modernizing university admissions, benefiting both institutions and applicants alike.

# Bibliography

## BIBLIOGRAPGHY ANEXURES

1. Rutuja Konde, Rutuja Somvanshi, Pratiksha Khaire, Prachi Zende, Kamlesh Patil, “Airfare Price

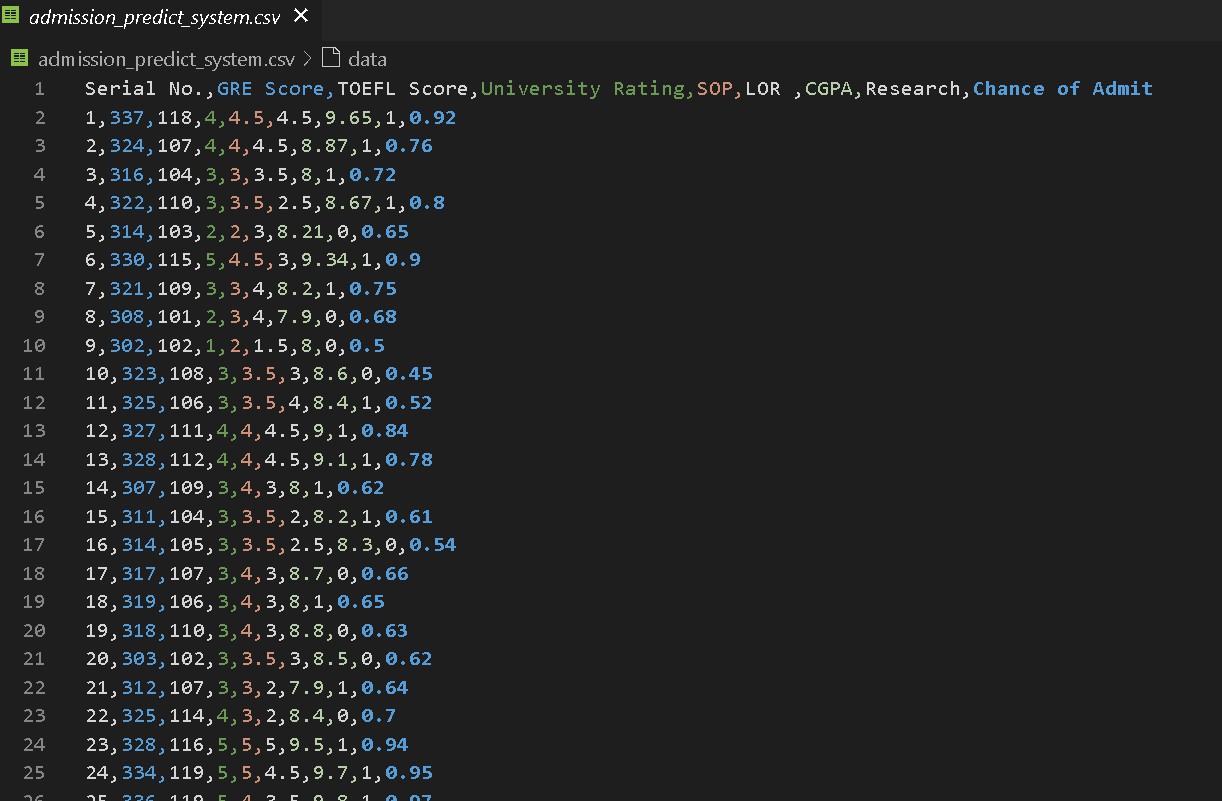
Prediction System”, International Research Journal of Engineering and Technology (IRJET), Volume: 09

Issue: 05, May 2022.

1. Mrs. Sowjanya, Saud Ikram, Rayyan Khan, Shaik Haseeb Jawad, Saadul arifeen, “Prediction of Airfare Prices Using Machine Learning,” International Journal of Mechanical Engineering, Vol.7 No.6, ISSN: 0974-5823, June, 2022.
2. Pavithra Maria K, Anitha K L, “Flight Price Prediction for Users by Machine Learning Techniques”, International Advanced Research Journal in Science, Engineering and Technology, Vol.8, Issue 3, DOI: 10.17148/IARJSET.2021.8321, March 2021.
3. Prof. Ms. Archana Dirgule, Shubham Agarwal, Ram Agrawal, Neha Singh, Kiran Adsul, “Flight Fare Prediction using Random Forest Algorithm,” International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), Volume 2, Issue 3, ISSN (Online) 2581-9429, May 2022.
4. Neel Bhosale, Pranav Gole, Hrutuja Handore, Priti Lakde, Gajanan Arsalwad, “Flight Fare Prediction Using Machine Learning,” International Journal for Research in Applied Science & Engineering Technology (IJRASET), ISSN: 2321-9653, Volume 10, Issue V, May 20

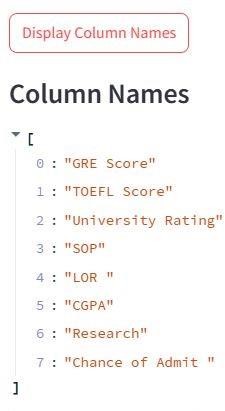
# Appendix

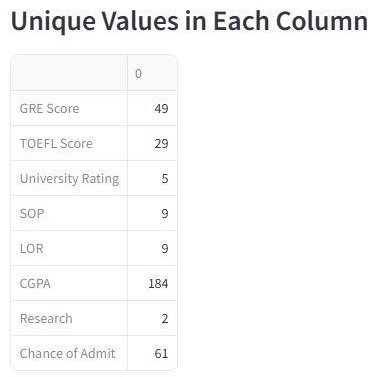
**Data Sheet:**

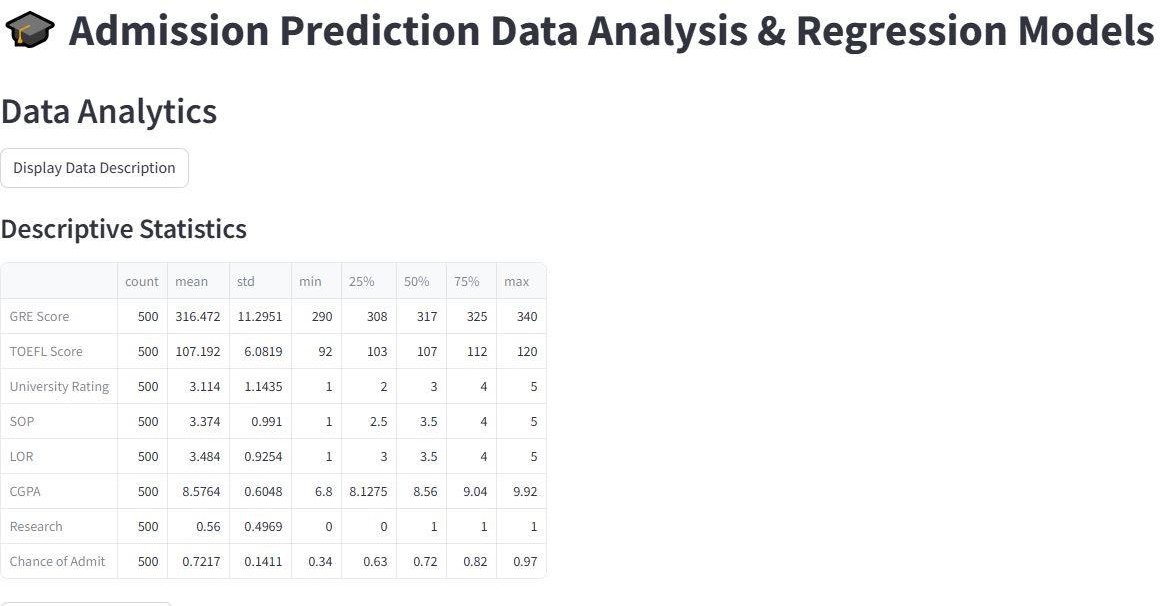
****

**User Manual**

**1. Screens**

****





****