UNIVERSITY ADMIT ELIGIBILITY PREDICTOR

PROJECT DOCUMENTATION

Team ID : PNT2022TMID15504

Team Members

1. BOOSHIKA A
2. JAYAPRIYA A
3. JANSHI MARY S
4. KEERTHI REDDY R
5. MANASA S

Abstract:

In the current world scenario, it is not enough for a student to just have an Undergraduate degree. Most employers now look for higher qualifications in their new recruits. As a result, the demands for a good higher education are at an all time high. A lot of students from India prefer to continue their higher education with foreign universities, especially in the United States. In order to get admitted to these foreign universities, a set of academic requirements are needed. However, because of the sheer number of universities of different levels, students are often stuck in a dilemma till the very last minute as to whether or not their applications will be accepted or not as no concrete documentation is available which lists the requirements.

We are here to provide a solution to that problem. Not only do we provide a single platform that documents all the requirements as well as the different tiers of universities, but our website also incorporates an AI Model that was built after considering many leading Machine Learning Algorithms, to provide the most accurate prediction of how much of a chance of admissions does a student’s current grades and other academic transcripts allow them in the tier of universities of their choice.

We have developed an AI based application that asks for the users to input their academic transcripts data and calculates their chances of admission into the University Tier that they selected. It also provides an analysis of the data and shows how chances of admissions can depend on various factors.

TABLE OF CONTENTS

CHAPTER - 1

INTRODUCTION

1.1 Problem Statement

Students are often worried about their chances of admission to University. The aim of this project is to help students in shortlisting universities with their profiles. The predicted output gives them a fair idea about their admission chances in a particular university. This analysis should also help students who are currently preparing or will be preparing to get a better idea.

Significance

We provide a single platform that documents all the requirements as well as the different tiers of universities, but our website also incorporates an AI Model that was built after considering many leading Machine Learning Algorithms, to provide the most accurate prediction of how much of a chance of admissions does a student's current grades and other academic transcripts allow them in the tier of universities of their choice.Students can register with their personal as well as marks details for prediction the admission in colleges and the administrator can allot the seats for the students.Administrator can add the college details and the batch details. Using this software, the entrance seat allotment became easier and can be implemented using the system .The main advantage of the project is the computerization of the entrance seat allotment process. Administrator has the power for the allotment. He can add the allotted seats into a file and the details are saved into the system. The total time for the entrance allotment became lesser and the allotment process became faster.

Project scope

The scope of this project is a web application that allows users to enter their academic data and get predictions of their chances of admissions in the university tier of their choosing.It also provides an analysis based on the data set used that shows how the different parameters affect chances of admissions. A Database will also be implemented for the system so that students can save their data and review and edit it as they progress with the most recent predictions being saved with their profile.

Need

This system is needed so as to answer the queries of students in a complete and concise manner as well as to provide them an as accurate as possible analysis of their chances of admissions to their dream universities.

Benefits

The people who will benefit the most from using this system are students. Especially students looking to pursue their higher education from foreign universities, particularly in the United States.

1.2. Related Works

Binu et al. proposed a cloud-based data analysis and prediction system for predicting university admission. There were two modules in the proposed framework, i.e. A Hadoop MapReduce data storage module and an Artificial Neural Network to predict the chances. The data collected had attributes such as status, rank, board, quota, etc. The system did not use academic qualifications in the forecasting process. The neural network had two input nodes, one hidden layer with two nodes, and one output layer with two nodes.

Ghai developed an American Graduate Admission Prediction model that allows students to choose an apt university by predicting whether or not they will be admitted to the university. Gupta et al. developed a machine learning decision support system for the prediction of graduate admissions in the USA by taking account of certain parameters in standardized tests.

Acharya et al. proposed a comparative approach by developing four machine learning regression models: linear regression, support vector machine, decision tree, and random forest for predictive analytics of graduate admission chances.

Then compute error functions for the developed models and compare their performances to select the best performing model out of these developed models. The linear regression is the best performing model with an R2 score of 0.72.

1.3. Existing System

Previous research done in this area used Naive Bayes algorithm which will evaluate the success probability of student application into a respective university but the main drawback is they did not consider all the factors which will contribute in the student admission process like TOEFL/IELTS , SOP, LOR and under graduate score.

Bayesian Networks Algorithms have been used to create a decision support network for evaluating the application submitted by foreign students of the university. This model was developed to forecast the progress of prospective students by comparing the score of students currently studying at university.

The model thus predicted whether the aspiring student should be admitted to university on the basis of various scores of students. Since the comparisons are made only with the students who got admission into the universities but not with students who got their admission rejected, this method will not be that accurate.

Today in college's student details are entered manually .The student details in separate records are tedious tasks .Referring to all these records updating is needed .There is a chance for more manual errors.

1. When the student comes to college.

2. First of all,he/she takes admission form from reception.

3. Fill it and submit it to the office.

4. Filled form is first checked with documents like merit list an details came from universitY and verified by an official person ,if there is any mistake then it is corrected

5. At the time of submission of it the fee is deposited by the candidate.

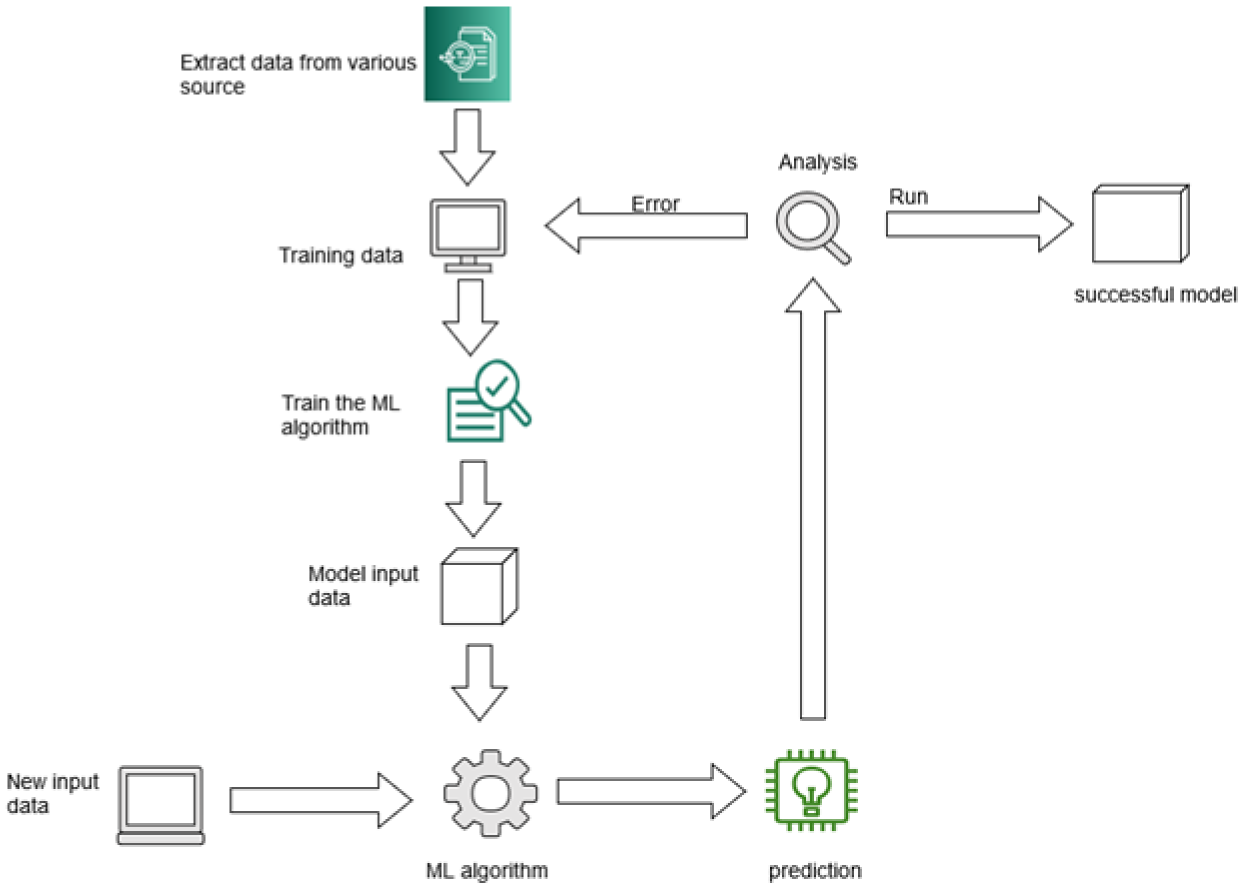
6. At the time of submission of admission form admission number is assigned to the candidate by the institute.

7. Candidate gets the receipt of fees deposition.

When students come from rural places , they find it hard to go along with the formal procedures. So, this application helps them a lot and eases out their fear. Whatever may be their scores , this application helps to find the best colleges . Hence, our proposed computer aided system will help the students to get the list of all colleges in which they could get the admission at the click of a button.

After intermediate, students desiring to pursue engineering face a lot of problems in choosing a good college and branch of their choice. Admission into engineering colleges across states in India happens generally through Common Entrance Tests (CET). The examination authority of every state carries out the admission, through a centralized admission process. This admission process happens through many rounds, depending on availability of seats.

1.4. Proposed System



Initially, the authors collect data from the respective university. After collecting the data, they pre-process the data, extract the features. Then they apply supervised machine learning methods that train, validate the data, and extract knowledge from it.

The main objective of this project is to help the students to save their time and money that they have to spend at the education consultancy firms. And also it will help them to limit their number of applications to a small number by providing them with the suggestion of the universities where they have the best chance of securing admission thus saving more money on the application fees.

CHAPTER - 2

PROJECT FUNCTIONALITY

2.1 Anaconda Installation

Anaconda Navigator is a desktop graphical user interface that comes with the Anaconda distribution and allows us to run programs and manage anaconda packages, environments, and channels without having to use command-line commands. Packages can be found on Anaconda.org or in a local Anaconda Repository using Navigator. It’s compatible with Windows, Mac OS X, and Linux. Many scientific packages rely on certain versions of other programs to run.

What is Anaconda Navigator?

- Data scientists frequently utilize various versions of many software and isolate these versions using distinct environments.
- Anaconda is a package and environment manager that can be run from the command line. This assists data scientists in ensuring that each version of each package has all of the dependencies it needs and functions properly.
- Navigator is a point-and-click interface for working with packages and environments that eliminates the need to type anaconda instructions into a terminal window. We may use it to search packages, install them in an environment, execute them, and update them all from within the navigator.
- Jupyter Notebooks can be used in the same way. Jupyter Notebooks are a popular system that allows us to integrate our code, descriptive text, output, graphics, and interactive interfaces into a single notebook file that we can edit, view, and use in a web browser.

How to install an anaconda navigator?

Below steps shows how to install anaconda navigator on the windows operating system are as follows.

1) Download anaconda navigator's latest version. At the time of downloading we need to make sure we download the python 3.7 version.

2) After downloading the installer file. Open the same with admin privileges. It will open the installer window. Click on the next button to start the anaconda navigator installation process.

3) After opening the installer accept the license agreement.

4) After accepting the license agreement, select the installation type. We have selected the installation type as just me.

5) After selecting the installation type, select the installation location.

6) After selecting the installation location, the next step is to select the advanced installation options.

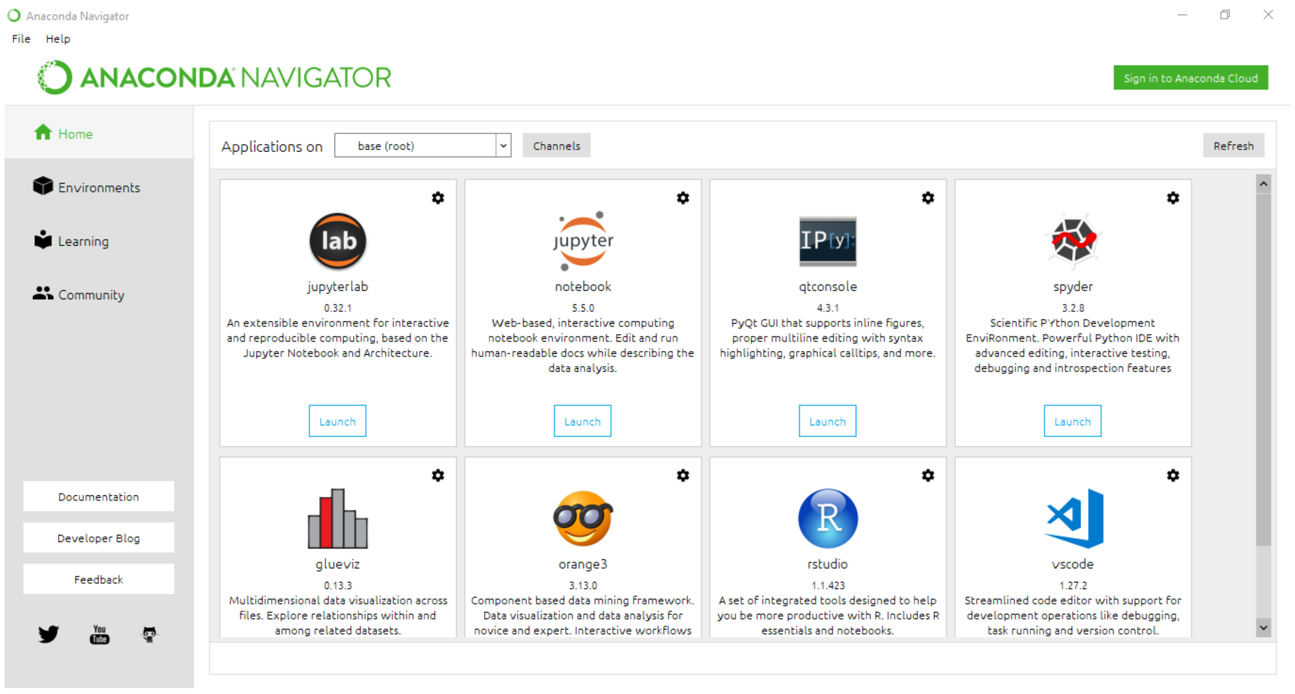
7) After selecting the advanced installation option click on install and check the progress of the installation process.

8) Click on next to download and install the PyCharm. PyCharm is recommended to be installed at the time of installing anaconda navigator.

9) Finish the installation –

10) Open the anaconda navigator to check the GUI.

ANACONDA®



jupyterlab
0.32.1
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

notebook
5.5.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

spyder
3.2.8
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

glueviz
0.13.3
Multidimensional data visualization across files. Explore relationships within and among related datasets.

orange3
3.13.0
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows

rstudio
1.1.423
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

vscode
1.27.2
Streamlined code editor with support for development operations like debugging, task running and version control.

2.2 Install Python Packages

Python libraries that are used in Machine Learning are:

- Numpy
- Scipy
- Scikit-learn
- Theano
- TensorFlow
- Keras
- PyTorch
- Pandas
- Matplotlib

Numpy

NumPy is a very popular python library for large multi-dimensional array and matrix processing, with the help of a large collection of high-level mathematical functions. It is very useful for fundamental scientific computations in Machine Learning. It is particularly useful for linear algebra, Fourier transform, and random number capabilities. High-end libraries like TensorFlow use NumPy internally for manipulation of Tensors.

SciPy

SciPy is a very popular library among Machine Learning enthusiasts as it contains different modules for optimization, linear algebra, integration and statistics. There is a difference between the SciPy library and the SciPy stack. The SciPy is one of the core packages that make up the SciPy stack. SciPy is also very useful for image manipulation.

Scipit-learn

Scikit-learn is one of the most popular ML libraries for classical ML algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms. Scikit-learn can also be used for data-mining and data-analysis, which makes it a great tool who is starting out with ML.

TensorFlow

TensorFlow is a very popular open-source library for high performance numerical computation developed by the Google Brain team in Google. As the name suggests, Tensorflow is a framework that involves defining and running computations involving tensors. It can train and run deep neural networks that can be used to develop several AI applications. TensorFlow is widely used in the field of deep learning research and application.

2.3 Python Web Frameworks

Flask for Web API Applications

The tools that allow us to write programs in Python to build a web-based application are called web frameworks. There are a lot. Django is probably the most famous one. However, the learning curve of different web frameworks can vary dramatically. Some web frameworks assume you use a model-view design, and you need to understand the rationale behind it to make sense of how you should use it.

As a machine learning practitioner, you probably want to do something quick, not too complex, and yet powerful enough to meet many use cases. Flask is probably a good choice in this class.

Flask is a lightweight web framework. You can run it as a command and use it as a Python module. Save the above into server.py or any filename you like, then run it on a terminal.

2.4 Jupyter Notebook

Notebook documents (or "notebooks", all lower case) are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python) and rich text elements (paragraph, equations, figures, links, etc…). Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc..) as well as executable documents which can be run to perform data analysis.

Jupyter Notebook App

The *Jupyter Notebook App* is a server-client application that allows editing and running notebook documents via a web browser. The *Jupyter Notebook App* can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet.

In addition to displaying/editing/running notebook documents, the *Jupyter Notebook App* has a "Dashboard" (Notebook Dashboard), a "control panel" showing local files and allowing to open notebook documents or shutting down their kernels.

kernel

A notebook *kernel* is a “computational engine” that executes the code contained in a Notebook document. The *ipython kernel*, referenced in this guide, executes python code. Kernels for many other languages exist (official kernels).

When you open a Notebook document, the associated *kernel* is automatically launched. When the notebook is *executed* (either cell-by-cell or with menu *Cell -> Run All*), the *kernel* performs the computation and produces the results. Depending on the type of computations, the *kernel* may consume significant CPU and RAM. Note that the RAM is not released until the *kernel* is shut-down.

2.5 IBM Cloud

Cloud Pak for Data as a Service is a cloud service platform for all your data governance, data engineering, data analysis, and AI lifecycle tasks. Cloud Pak for Data as a Service implements a data fabric solution so that you can provide instant and secure access to trusted data to your organization, automate processes and compliance, and deliver trustworthy AI in your applications.

Build a model asset

1. Create a project.
2. Associate the Watson Machine Learning service with the project.

3. Add data to the project. If necessary, prepare your data.
4. Choose a tool to build a model. You can choose from code editors, graphical builders, or automatic tools.

Deploy the model

1. Create a deployment space and add the model to it.
2. Deploy and score the model, and review prediction scores and insights.
3. Monitor deployment jobs in a dashboard.

Build trust in your models

1. Evaluate your deployment for bias or drift.
2. Update your data and retrain the model until you reach your quality goals.
3. Update deployments with better-performing models.
4. Continue to evaluate, retrain, and update the deployed model.

CHAPTER - 3

REQUIREMENT SPECIFICATIONS

3.1 Functional Requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP

FR-3	Instant Prediction	It is provided by filling the details and submitting the form. Form will be given Fill the form Submit the form. Get Instant Prediction
FR-4	Search Result	Displays the college specific details when the feature is clicked. Enter the college name in the given field. Get the details about the college
FR-5	Student Data Management	Previous year students' dataset is maintained and updated. Students can access these data in read-only mode. Students cannot update the database.
FR-6	University Details	In-detail information like placements and higher studies will be displayed

3.2 Non-Functional Requirements

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	● No training is required to use the website. ● The form, home, about, FAQ and analysis pages load up within 10 seconds. ● The results from the predictor should not take more than 30 seconds.
NFR-2	**Security**	● The system shall provide password protected access to the website to all users – students and admins both
NFR-3	**Reliability**	● The system shall be completely operational all hours of the day unless

		system failure or upgradation work is to be performed ● Down time after a failure shall not exceed 24 hours
NFR-4	**Performance**	● The system can support any number of users at a time. ● The mean time to view a web page over a 56Kbps modem connection shall not exceed 5 seconds
NFR-5	**Availability**	● The product is always readily available. ● The system will be able to incorporate more features without major reengineering. ● The system web site shall be viewable from Internet Explorer 4.0 or later, Netscape Navigator/Communicator 3.0 or later and the America Online web browser version 3.0 or later.
NFR-6	**Scalability**	● It can deliver the same throughput for varying levels of load on the internal applications, hardware, and database. Lesser response time

3.3 Hardware and Software Requirements

HARDWARE REQUIREMENTS

Operating System :

Windows, Android, MacOs or a popular Linux distribution like Ubuntu

Processor:

2.7 GHz Intel Core i5 and above

Memory:

8GB RAM

SOFTWARE REQUIREMENTS

Web Application :

HTML, CSS, JS using VS Code

Model Building:

Jupyter Notebook, Anaconda Navigator

Training and Deployment:

IBM Cloud, IBM Watson Studio

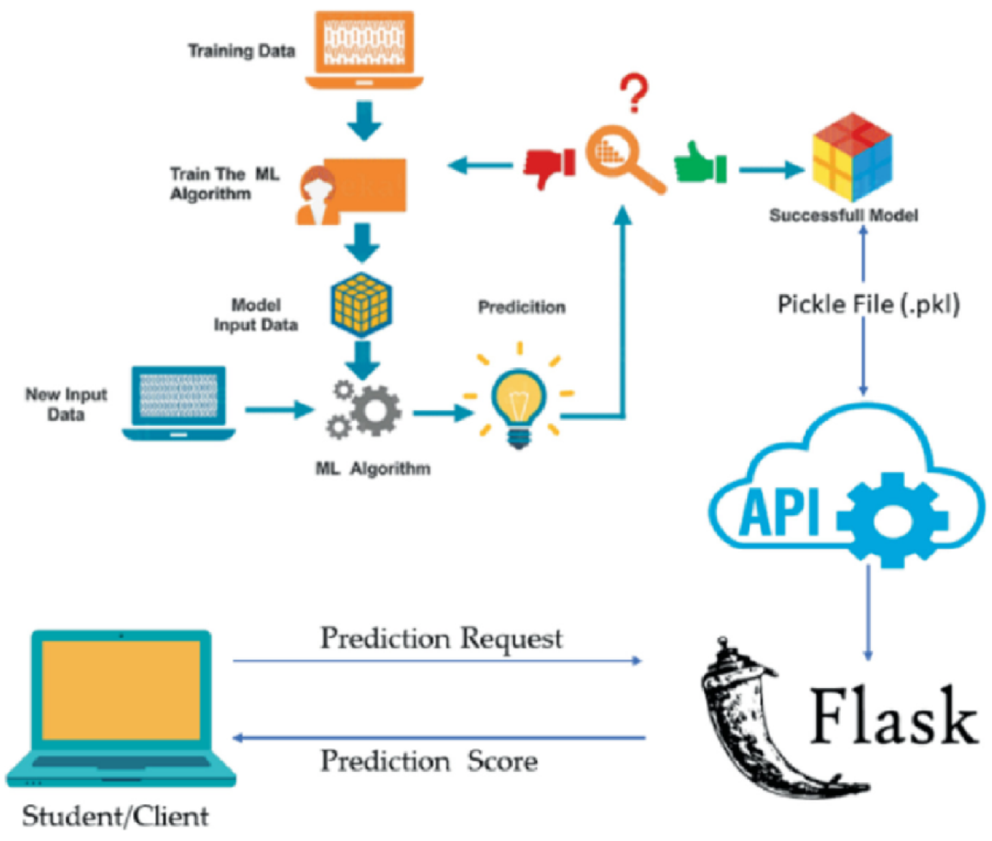
Dataset:

Kaggle

CHAPTER - 4

DIAGRAMS

4.1 Architecture Diagram



CHAPTER - 5

PROPOSED WORK

5.1 Explanation

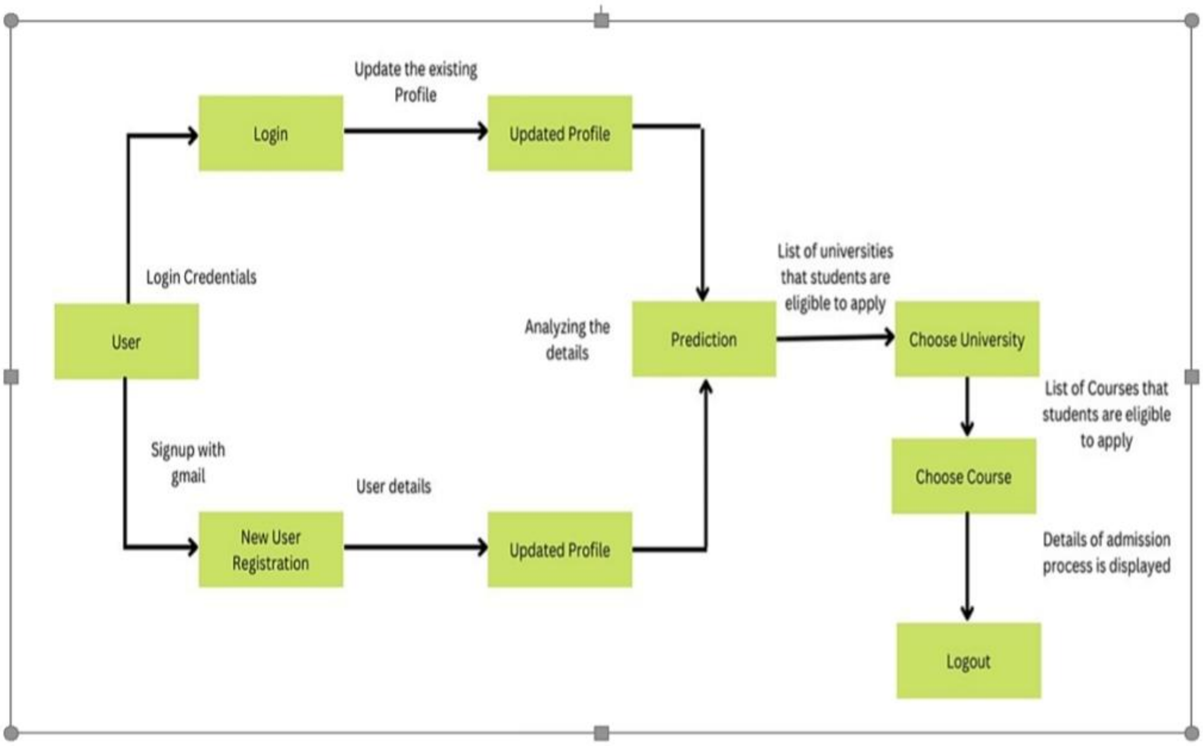
We will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI. This section has the following tasks

- Building HTML Pages
- Building server-side script

Next, we will integrate the flask and IBM cloud for deployment. Then finally flask is integrated with the scoring endpoint.

DATA FLOW DIAGRAM

A Data Flow Diagram(DFD) is a traditional visual representation information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.


Update the existing Profile
Login
Updated Profile
Login Credentials
List of universities that students are eligible to apply
User
Analyzing the details
Prediction
Choose University
List of Courses that students are eligible to apply
Signup with gmail
Choose Course
User details
New User Registration
Updated Profile
Details of admission process is displayed
Logout

5.2 Model Building

Introduction to Dataset

When applying for Master's degrees in foreign countries, there are several variables to consider. You must have a decent GRE score, a sop (statement of purpose), or a letter of reference, among other things. If you are not from an English-speaking nation, you also need to submit a TOEFL score.

The dataset includes the following attributes:

1. GRE Scores (out of 340)
2. TOEFL Scores (out of 120)
3. University Rating (out of 5)
4. Statement of Purpose and Letter of Recommendation Strength (out of 5)
5. Undergraduate GPA (out of 10)
6. Research Experience (either 0 or 1)
7. Chance of Admit (ranging from 0 to 1)

Step 1: Importing Necessary Modules/Libraries

First and foremost, import the necessary Python libraries. In our case, we'll be working with pandas, NumPy, matplotlib, seaborn, and scikit-learn.

To import them, use the following code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

Step 2: Loading Dataset Into the Program

Import csv file

The csv module enables us to read each of the row in the file using a comma as a delimiter. We first open the file in read only mode and then assign the delimiter. Finally use a for loop to read each row from the csv file.

Step 3: Data Pre-processing and Data Splitting

Data Preprocessing

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.

Why is Data preprocessing important?

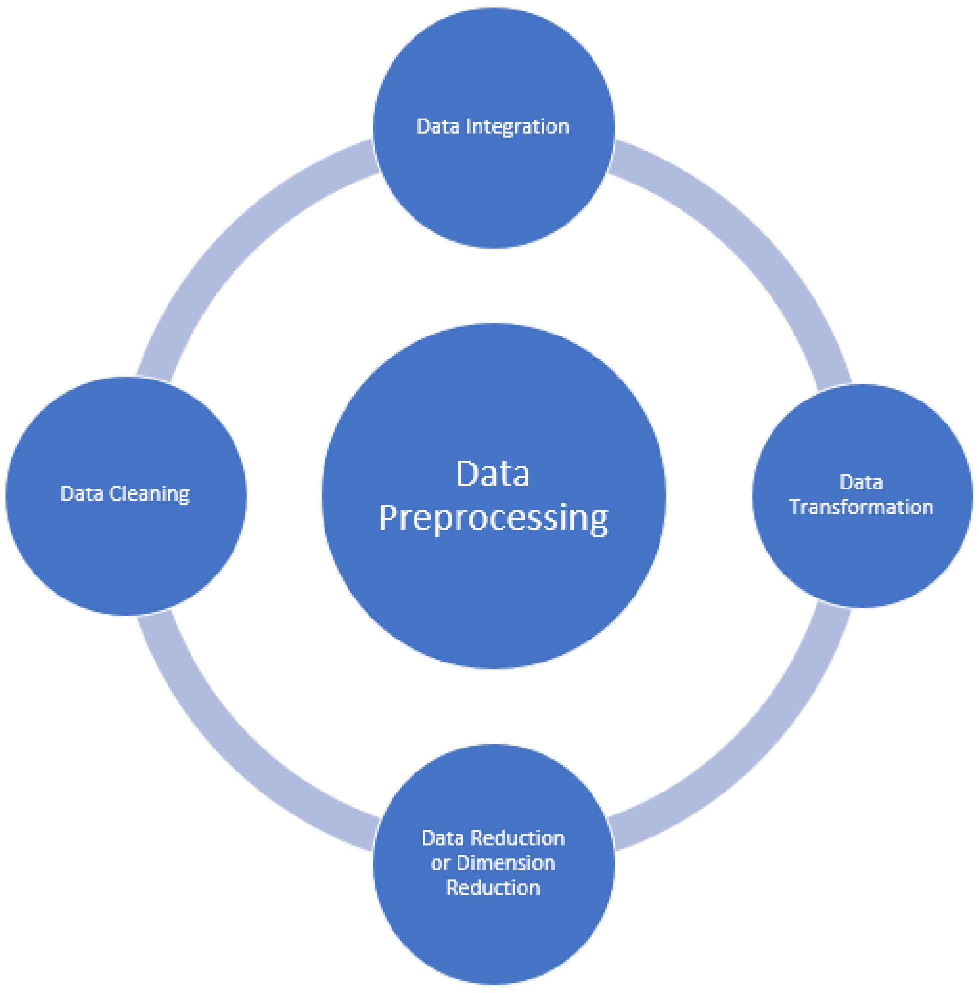
Preprocessing of data is mainly to check the data quality. The quality can be checked by the following

- Accuracy: To check whether the data entered is correct or not.

- Completeness: To check whether the data is available or not recorded.
- Consistency: To check whether the same data is kept in all the places that do or do not match.
- Timeliness: The data should be updated correctly.
- Believability: The data should be trustable.
- Interpretability: The understandability of the data.

Major Tasks in Data Preprocessing:

1. Data cleaning
2. Data integration
3. Data reduction
4. Data transformation



Data cleaning:

Data cleaning is the process to remove incorrect data, incomplete data and inaccurate data from the datasets, and it also replaces the missing values.

Data Preprocessing

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.

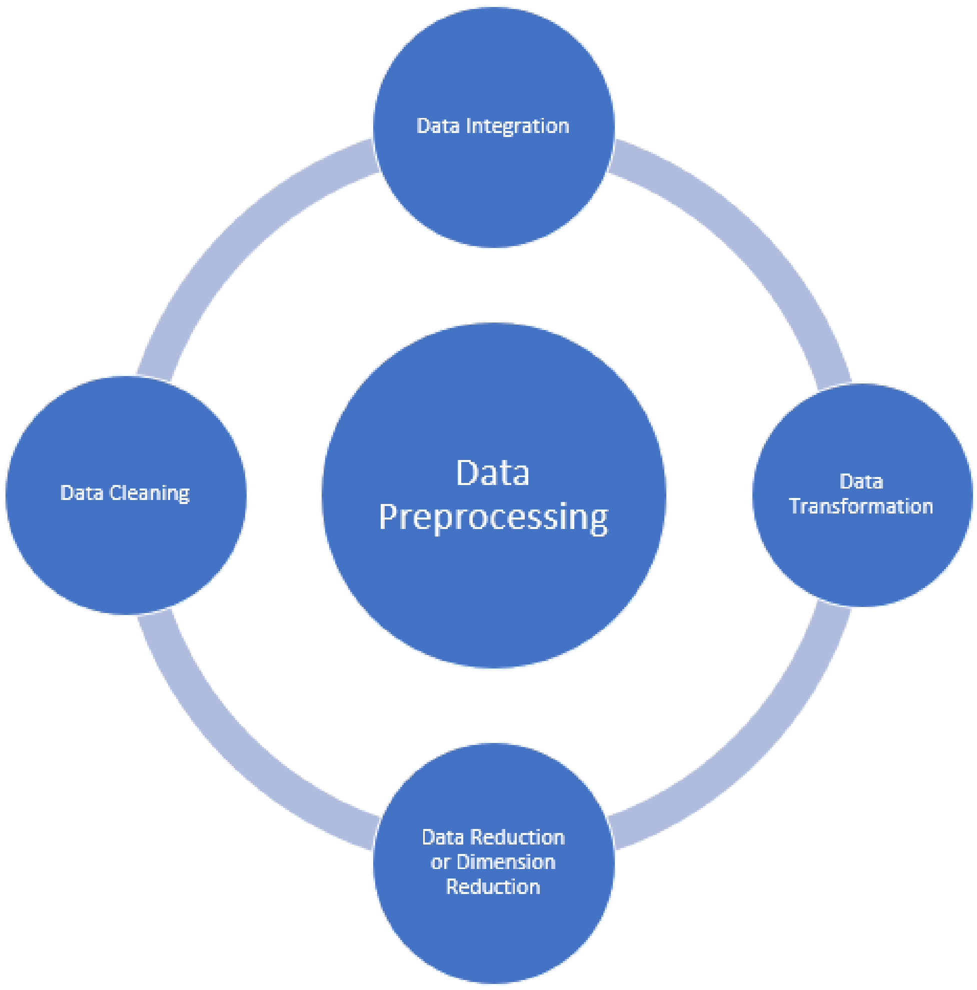
Why is Data preprocessing important?

Preprocessing of data is mainly to check the data quality. The quality can be checked by the following

- Accuracy: To check whether the data entered is correct or not.
- Completeness: To check whether the data is available or not recorded.
- Consistency: To check whether the same data is kept in all the places that do or do not match.
- Timeliness: The data should be updated correctly.
- Believability: The data should be trustable.
- Interpretability: The understandability of the data.

Major Tasks in Data Preprocessing:

1. Data cleaning
2. Data integration
3. Data reduction
4. Data transformation



Data cleaning:

Data cleaning is the process to remove incorrect data, incomplete data and inaccurate data from the datasets, and it also replaces the missing values. There are some techniques in data cleaning

Handling missing values:

- Standard values like “Not Available” or “NA” can be used to replace the missing values.
- Missing values can also be filled manually but it is not recommended when that dataset is big.

- The attribute’s mean value can be used to replace the missing value when the data is normally distributed
  wherein in the case of non-normal distribution the median value of the attribute can be used.

- While using regression or decision tree algorithms the missing value can be replaced by the most probable value.

Step 3: Building the Model

Logistic regression Methodology

Logistic regression is a linear classifier, so you’ll use a linear function $f(\mathbf{x}) = b_0 + b_1x_1 + \cdots + b_rx_r$, also called the logit. The variables b_0, b_1, …, b_r are the estimators of the regression coefficients, which are also called the predicted weights or just coefficients.

The logistic regression function $p(\mathbf{x})$ is the sigmoid function of $f(\mathbf{x})$: $p(\mathbf{x}) = 1 / (1 + \exp(-f(\mathbf{x})))$. As such, it’s often close to either 0 or 1. The function $p(\mathbf{x})$ is often interpreted as the predicted probability that the output for a given $\mathbf{x}$ is equal to 1. Therefore, $1 - p(x)$ is the probability that the output is 0.

Logistic regression determines the best predicted weights b_0, b_1, …, b_r such that the function $p(\mathbf{x})$ is as close as possible to all actual responses y_i, $i = 1, …, n$, where n is the number of observations. The process of calculating the best weights using available observations is called model training or fitting.

To get the best weights, you usually maximize the log-likelihood function (LLF) for all observations $i = 1, …, n$. This method is called the maximum likelihood estimation and is represented by the equation $\text{LLF} = \Sigma_i(y_i \log(p(\mathbf{x}_i)) + (1 - y_i) \log(1 - p(\mathbf{x}_i)))$.

When $y_i = 0$, the LLF for the corresponding observation is equal to $\log(1 - p(\mathbf{x}_i))$. If $p(\mathbf{x}_i)$ is close to $y_i = 0$, then $\log(1 - p(\mathbf{x}_i))$ is close to 0. This is the result you want. If $p(\mathbf{x}_i)$ is far from 0, then $\log(1 - p(\mathbf{x}_i))$ drops significantly. You don't want that result because your goal is to obtain the maximum LLF. Similarly, when $y_i = 1$, the LLF for that observation is $y_i \log(p(\mathbf{x}_i))$. If $p(\mathbf{x}_i)$ is close to $y_i = 1$, then $\log(p(\mathbf{x}_i))$ is close to 0. If $p(\mathbf{x}_i)$ is far from 1, then $\log(p(\mathbf{x}_i))$ is a large negative number.

There are several mathematical approaches that will calculate the best weights that correspond to the maximum LLF, but that's beyond the scope of this tutorial. For now, you can leave these details to the logistic regression Python libraries you'll learn to use here!

Once you determine the best weights that define the function $p(\mathbf{x})$, you can get the predicted outputs $p(\mathbf{x}_i)$ for any given input $\mathbf{x}_i$. For each observation $i = 1, …, n$, the predicted output is 1 if $p(\mathbf{x}_i) > 0.5$ and 0 otherwise. The threshold doesn't have to be 0.5, but it usually is. You might define a lower or higher value if that's more convenient for your situation.

There's one more important relationship between $p(\mathbf{x})$ and $f(\mathbf{x})$, which is that $\log(p(\mathbf{x}) / (1 - p(\mathbf{x}))) = f(\mathbf{x})$. This equality explains why $f(\mathbf{x})$ is the logit. It implies that $p(\mathbf{x}) = 0.5$ when $f(\mathbf{x}) = 0$ and that the predicted output is 1 if $f(\mathbf{x}) > 0$ and 0 otherwise.

Classification Performance

Binary classification has four possible types of results:

1. True negatives: correctly predicted negatives (zeros)
2. True positives: correctly predicted positives (ones)
3. False negatives: incorrectly predicted negatives (zeros)
4. False positives: incorrectly predicted positives (ones)

You usually evaluate the performance of your classifier by comparing the actual and predicted outputs and counting the correct and incorrect predictions.

The most straightforward indicator of classification accuracy is the ratio of the number of correct predictions to the total number of predictions (or observations). Other indicators of binary classifiers include the following:

- The positive predictive value is the ratio of the number of true positives to the sum of the numbers of true and false positives.
- The negative predictive value is the ratio of the number of true negatives to the sum of the numbers of true and false negatives.
- The sensitivity (also known as recall or true positive rate) is the ratio of the number of true positives to the number of actual positives.
- The specificity (or true negative rate) is the ratio of the number of true negatives to the number of actual negatives.

Gradient Boosting Regressor Methodology:

"Boosting" in machine learning is a way of combining multiple simple models into a single composite model. This is also why boosting is known as an additive model, since simple models (also known as weak learners) are added one at a time, while keeping existing trees in the model unchanged. As we combine more and more simple models, the complete final model becomes a stronger predictor. The term "gradient" in "gradient boosting" comes from the fact that the algorithm uses gradient descent to minimize the loss.

When gradient boost is used to predict a continuous value – like age, weight, or cost – we're using gradient boost for regression. This is not the same as using linear regression. This is slightly different from the configuration used for classification, so we'll stick to regression in this article.

Decision trees are used as the weak learners in gradient boosting. Decision Tree solves the problem of machine learning by transforming the data into tree representation. Each internal node of the tree representation denotes an attribute and each leaf node denotes a class label. The loss function is generally the squared error (particularly for regression problems). The loss function needs to be differentiable.

Also like linear regression we have concepts of residuals in Gradient Boosting Regression as well. Gradient boosting Regression calculates the difference between the current prediction and the known correct target value.

This difference is called residual. After that Gradient boosting Regression trains a weak model that maps features to that residual. This residual predicted by a weak model is added to the existing model input and thus this process nudges the model towards the correct target. Repeating this step again and again improves the overall model prediction.

Also it should be noted that Gradient boosting regression is used to predict continuous values like house price , while Gradient Boosting Classification is used for predicting classes like whether a patient has a particular disease or not.

The high level steps that we follow to implement Gradient Boosting Regression is as below:

- Select a weak learner
- Use an additive model
- Define a loss function
- Minimize the loss function

Advantages of Gradient Boosting

Better accuracy: Gradient Boosting Regression generally provides better accuracy. When we compare the accuracy of GBR with other regression techniques like Linear Regression, GBR is mostly the winner all the time. This is why GBR is being used in most of the online hackathons and competitions.

Deploy a Machine Learning Model from a Jupyter Notebook using IBM Watson Studio

Install the Python SDK

You will need to install the IBM Watson Machine Learning python SDK. You can do that by running pip install ibm-watson-machine-learning in the command line. You can also run this in its own cell in your notebook like this:

!pip install ibm-watson-machine-learning.

Set Your Client

You can start your code by importing the Watson Machine Learning package and supplying your API key and region. If you don't know what value you should supply for location you should check the city your instance is in. You take that city and find the region by consulting the locations page. Since my instance is in Dallas, I supplied "us-south" for the location variable below.

```
from ibm_watson_machine_learning import APIClient

import json

import numpy as np
```

These can be supplied as credentials.

```
wml_credentials = { "apikey": api_key, "url": 'https://' + location + '.ml.cloud.ibm.com'}client = APIClient(wml_credentials)
```

Set Your Deployment Space

We need to get the ID of your deployment space. You can do that by running the code below

```
client.spaces.list()
```

From the output of the above you will get an ID for your deployment space. Set your APIClient to use your deployment space with the code below.

```
client.set.default_space("<ID>")
```

Publish Your Model

Before we deploy our model, we need to publish it. First we have to decide what version of python we want to use and set some metadata. If you need to figure out what version of python you are using, run `import sys` then `sys.version`.

If you are also deploying a model based on sklearn, you will need to determine the version of that. You can run `sklearn.__version__` to figure that out.

```
metadata={client.repository.ModelMetaNames.NAME:'Admission Logr Model',

client.repository.ModelMetaNames.TYPE: 'scikit-learn_0.22',

    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

You will need to get the ID of your published model. You can get that by running the following.

```
models_details = client.repository.list_models()
```

Deploy Your Model

To deploy your model use your published model's ID from the cell above and run the code below.

```
metadata = {

    client.deployments.ConfigurationMetaNames.NAME: "Deployment of Iris KNN Model",

   client.deployments.ConfigurationMetaNames.ONLINE: {}

}


created_deployment = client.deployments.create("<your-published-model-id>", meta_props=metadata)
```

Test Your Deployment

Your model is now deployed and we can now send data to it and get a response. First we need to get the ID of the deployed model by running the code below.

```
client.deployments.list()
```

We can then use that ID and send observations in a list of lists.

```
scoring_payload = {"input_data": [{"values": [[5.1, 3.5, 1.4, 0.2]]}]}
```

```
predictions = client.deployment.score("<your-deployment-id>", scoring_payload)
```

In the predictions variable we now have stored a prediction of the class and probability of each class.

5.4 Future Enhancement

The main objective of this project is to help the students to save their time and money that they have to spend at the education consultancy firms. and also it will help them to limit their number of applications to a small number by providing them with the suggestion of the universities where they have the best chance of securing admission thus saving more money on the application fees.

From the proposed work, we are able to identify only the chance to get a seat and we are not able to identify which university we are obtaining. So, in future we can develop a representation, which gives us a list of universities in which we can obtain admission. The user does not need to travel a long distance for the admission and his/her time is also saved as a result of this automated system.

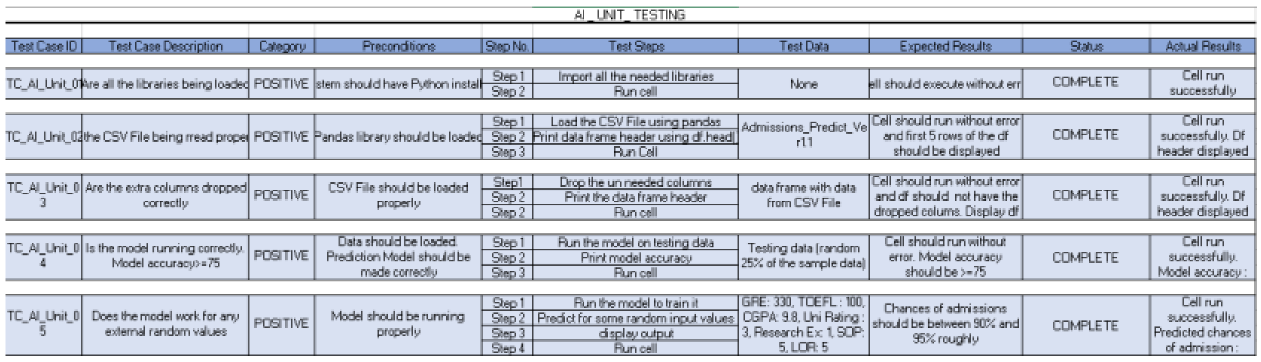
CHAPTER - 6

TESTING

The following section contains the report of the testing phase of the software
Project Type - Web Application
Language - Python, javascript, css, html
Total Number of test cases - 51
Number of test cases executed - 51
Number of test cases Passed- 50
Number of test cases Failed -1
Positive Test Cases- 39
Negative Test Cases -12
The code was tested both while developing as well as after development the following are some classes of tests that were executed.

Unit Testing

Testing of an individual software component or module is termed as Unit Testing. It is typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. The Code was developed in 2 separate parts.
1. AI Model developed using Jupyter Notebook.
2. Web Front end was developed using VS Code .

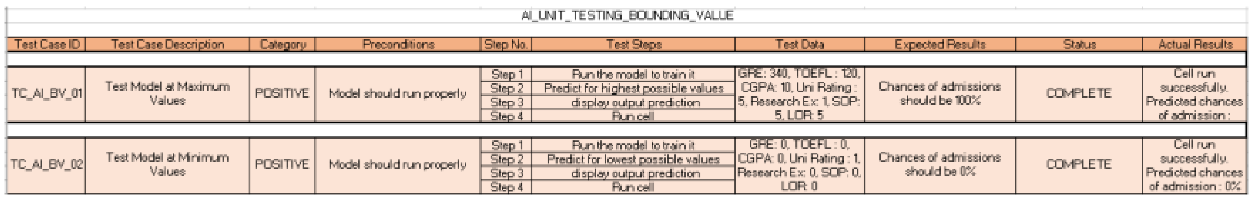


AI _ UNIT _ TESTING

Test Case ID	Test Case Description	Category	Preconditions	Step No.	Test Steps	Test Data	Expected Results	Status	Actual Results
TC_AI_Unit_01	Are all the libraries being loaded	POSITIVE	stem should have Python install	Step 1	Import all the needed libraries	None	ell should execute without err	COMPLETE	Cell run successfully
				Step 2	Run cell				
TC_AI_Unit_02	the CSV File being rread proper	POSITIVE	Pandas library should be loaded	Step 1	Load the CSV File using pandas	Admissions_Predict_Ver11	Cell should run without error and first 5 rows of the df should be displayed	COMPLETE	Cell run successfully. Df header displayed
				Step 2	Print data frame header using df.head()				
				Step 3	Run Cell				
TC_AI_Unit_03	Are the extra columns dropped correctly	POSITIVE	CSV File should be loaded properly	Step1	Drop the un needed columns	data frame with data from CSV File	Cell should run without error and df should not have the dropped colums. Display df	COMPLETE	Cell run successfully. Df header displayed
				Step 2	Print the data frame header				
				Step 2	Run cell				
TC_AI_Unit_04	Is the model running correctly. Model accuracy>=75	POSITIVE	Data should be loaded. Prediction Model should be made correctly	Step 1	Run the model on testing data	Testing data (random 25% of the sample data)	Cell should run without error. Model accuracy should be >=75	COMPLETE	Cell run successfully. Model accuracy :
				Step 2	Print model accuracy				
				Step 3	Run cell				
TC_AI_Unit_05	Does the model work for any external random values	POSITIVE	Model should be running properly	Step 1	Run the model to train it	GRE: 330, TOEFL : 100, CGPA: 9.8, Uni Rating : 3, Research Ex: 1, SOP: 5, LOR: 5	Chances of admissions should be between 90% and 95% roughly	COMPLETE	Cell run successfully. Predicted chances of admission :
				Step 2	Predict for some random input values				
				Step 3	display output				
				Step 4	Run cell				

Boundary Value Testing

This type of testing checks the behaviour of the application at the boundary level. Boundary Value Testing is performed for checking if defects exist at boundary values. Boundary Value Testing is used for testing a different range of numbers. There is an upper and lower boundary for each range and testing is performed on these boundary values.

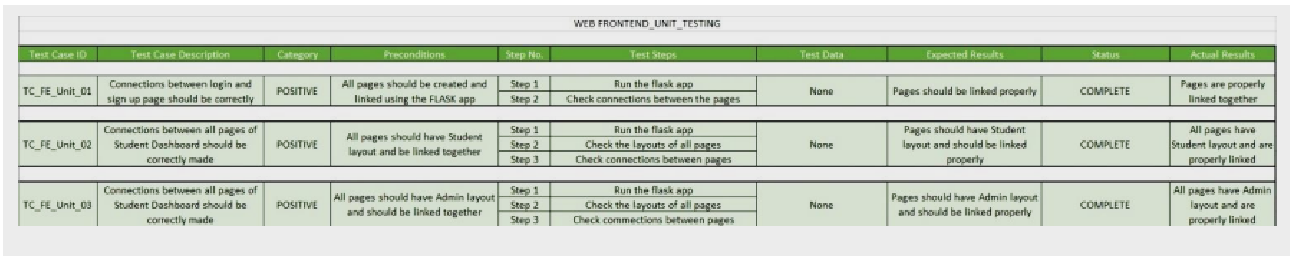


AI_UNIT_TESTING_BOUNDING_VALUE

Test Case ID	Test Case Description	Category	Preconditions	Step No.	Test Steps	Test Data	Expected Results	Status	Actual Results
TC_AI_BV_01	Test Model at Maximum Values	POSITIVE	Model should run properly	Step 1	Run the model to train it	GRE: 340, TOEFL : 120, CGPA: 10, Uni Rating : 5, Research Ex: 1, SOP: 5, LOR: 5	Chances of admissions should be 100%	COMPLETE	Cell run successfully. Predicted chances of admission :
				Step 2	Predict for highest possible values				
				Step 3	display output prediction				
				Step 4	Run cell				
TC_AI_BV_02	Test Model at Minimum Values	POSITIVE	Model should run properly	Step 1	Run the model to train it	GRE: 0, TOEFL : 0, CGPA: 0, Uni Rating : 1, Research Ex: 0, SOP: 0, LOR: 0	Chances of admissions should be 0%	COMPLETE	Cell run successfully. Predicted chances of admission : 0%
				Step 2	Predict for lowest possible values				
				Step 3	display output prediction				
				Step 4	Run cell				

Web Front End

The objective of this GUI Testing is to validate the GUI as per the business requirement. The expected GUI of the application is mentioned in the Detailed Design Document and GUI mockup screens.



WEB FRONTEND_UNIT_TESTING

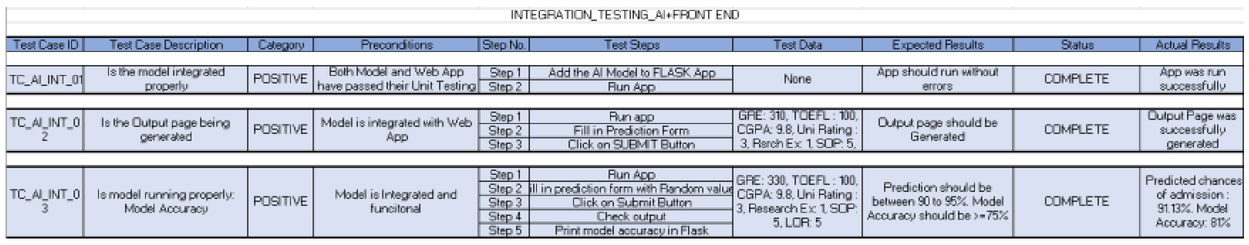
Test Case ID	Test Case Description	Category	Preconditions	Step No.	Test Steps	Test Data	Expected Results	Status	Actual Results
TC_FE_Unit_01	Connections between login and sign up page should be correctly	POSITIVE	All pages should be created and linked using the FLASK app	Step 1	Run the flask app	None	Pages should be linked properly	COMPLETE	Pages are properly linked together
				Step 2	Check connections between the pages				
TC_FE_Unit_02	Connections between all pages of Student Dashboard should be correctly made	POSITIVE	All pages should have Student layout and be linked together	Step 1	Run the flask app	None	Pages should have Student layout and should be linked properly	COMPLETE	All pages have Student layout and are properly linked
				Step 2	Check the layouts of all pages				
				Step 3	Check connections between pages				
TC_FE_Unit_03	Connections between all pages of Student Dashboard should be correctly made	POSITIVE	All pages should have Admin layout and should be linked together	Step 1	Run the flask app	None	Pages should have Admin layout and should be linked properly	COMPLETE	All pages have Admin layout and are properly linked
				Step 2	Check the layouts of all pages				
				Step 3	Check commections between pages				

Incremental Integration Testing

Testing of all integrated modules to verify the combined functionality after integration is termed as Integration Testing. Modules are typically code modules, individual applications, client and server applications on a network, etc. After all the parts were created, they were then integrated as follows

1. AI Component was integrated with the front end
2. Backend was integrated with the entire mode

AI component + Front End

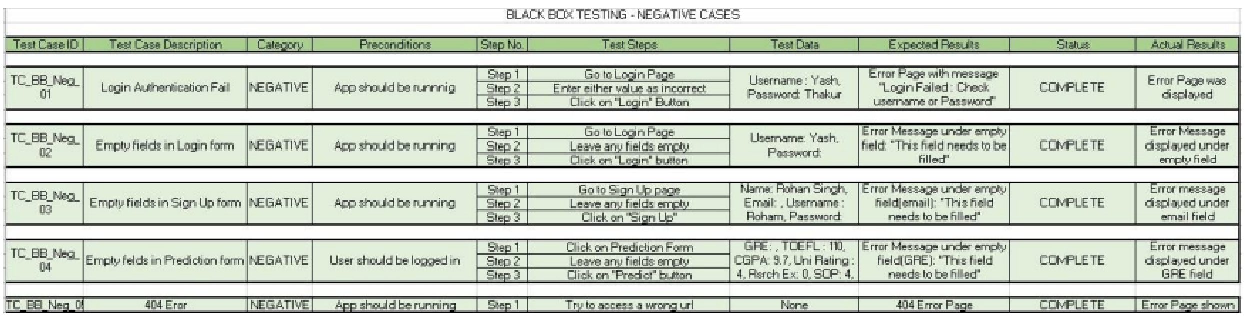


INTEGRATION_TESTING_AI+FRONT END

Test Case ID	Test Case Description	Category	Preconditions	Step No.	Test Steps	Test Data	Expected Results	Status	Actual Results
TC_AI_INT_01	Is the model integrated properly	POSITIVE	Both Model and Web App have passed their Unit Testing	Step 1	Add the AI Model to FLASK App	None	App should run without errors	COMPLETE	App was run successfully
				Step 2	Run App				
TC_AI_INT_0 2	Is the Output page being generated	POSITIVE	Model is integrated with Web App	Step 1	Run app	GRE: 310, TOEFL : 100, CGPA: 9.8, Uni Rating : 3, Rsrch Ex: 1, SOP: 5,	Output page should be Generated	COMPLETE	Output Page was successfully generated
				Step 2	Fill in Prediction Form				
				Step 3	Click on SUBMIT Button				
TC_AI_INT_0 3	Is model running properly: Model Accuracy	POSITIVE	Model is Integrated and funcitonal	Step 1	Run App	GRE: 330, TOEFL : 100, CGPA: 9.8, Uni Rating : 3, Research Ex: 1, SOP: 5, LOR: 5	Prediction should be between 90 to 95%. Model Accuracy should be >=75%	COMPLETE	Predicted chances of admission : 91.13%. Model Accuracy: 81%
				Step 2	ill in prediction form with Random value				
				Step 3	Click on Submit Button				
				Step 4	Check output				
				Step 5	Print model accuracy in Flask				

Negative Testing

Testers have the mindset of "attitude to break" and using Negative Testing they validate that if a system or application breaks. A Negative Testing technique is performed using incorrect data, invalid data or input. It validates that if the system throws an error of invalid input and behaves as expected.

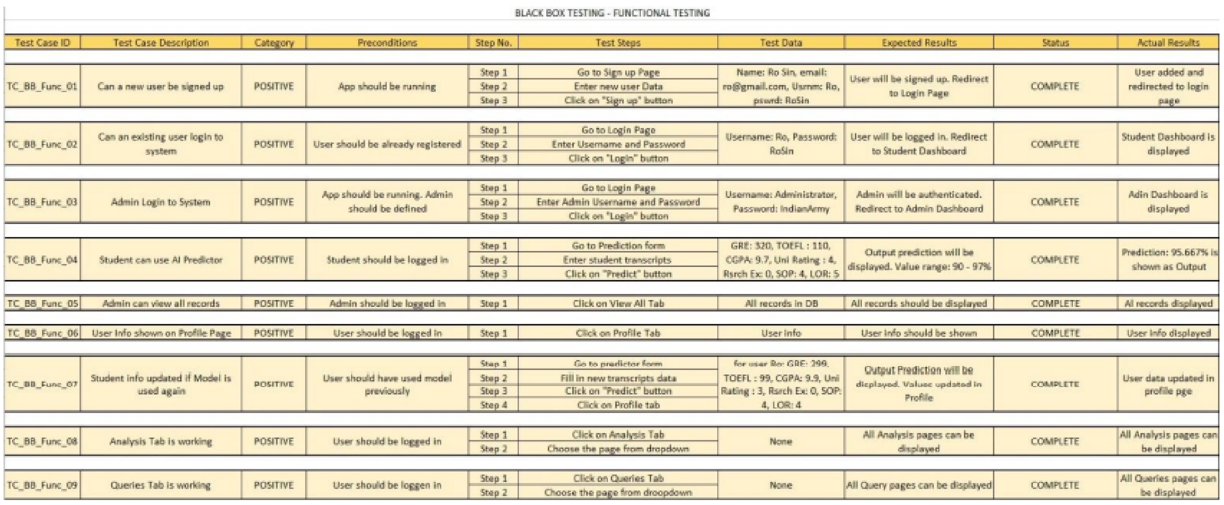


BLACK BOX TESTING - NEGATIVE CASES

Test Case ID	Test Case Description	Category	Preconditions	Step No.	Test Steps	Test Data	Expected Results	Status	Actual Results
TC_BB_Neg_01	Login Authentication Fail	NEGATIVE	App should be runnnig	Step 1	Go to Login Page	Username : Yash, Password: Thakur	Error Page with message "Login Failed : Check username or Password"	COMPLETE	Error Page was displayed
				Step 2	Enter either value as incorrect				
				Step 3	Click on "Login" Button				
TC_BB_Neg_02	Empty fields in Login form	NEGATIVE	App should be running	Step 1	Go to Login Page	Username: Yash, Password:	Error Message under empty field: "This field needs to be filled"	COMPLETE	Error Message displayed under empty field
				Step 2	Leave any fields empty				
				Step 3	Click on "Login" button				
TC_BB_Neg_03	Empty fields in Sign Up form	NEGATIVE	App should be running	Step 1	Go to Sign Up page	Name: Rohan Singh, Email: , Username : Roham, Password:	Error Message under empty field(email): "This field needs to be filled"	COMPLETE	Error message displayed under email field
				Step 2	Leave any fields empty				
				Step 3	Click on "Sign Up"				
TC_BB_Neg_04	Empty felds in Prediction form	NEGATIVE	User should be logged in	Step 1	Click on Prediction Form	GRE: , TOEFL : 110, CGPA: 9.7, Uni Rating : 4, Rsrch Ex: 0, SOP: 4,	Error Message under empty field(GRE): "This field needs to be filled"	COMPLETE	Error message displayed under GRE field
				Step 2	Leave any fields empty				
				Step 3	Click on "Predict" button				
TC_BB_Neg_0!	404 Eror	NEGATIVE	App should be running	Step 1	Try to access a wrong url	None	404 Error Page	COMPLETE	Error Page shown

Functionality Testing

This type of testing ignores the internal parts and focuses only on the output to check if it is as per the requirement or not. It is a Black-box type testing geared to the functional requirements of an application .



BLACK BOX TESTING - FUNCTIONAL TESTING

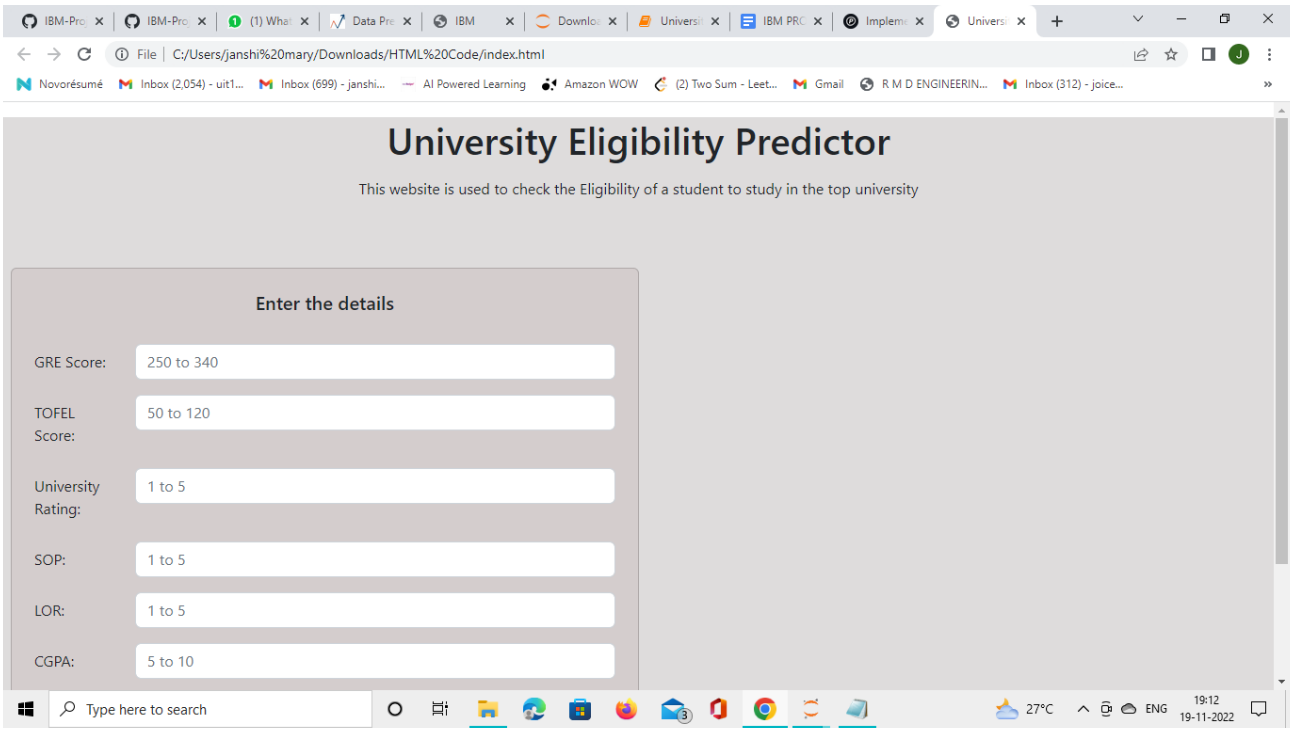
Test Case ID	Test Case Description	Category	Preconditions	Step No.	Test Steps	Test Data	Expected Results	Status	Actual Results
TC_BB_Func_01	Can a new user be signed up	POSITIVE	App should be running	Step 1	Go to Sign up Page	Name: Ro Sin, email: ro@gmail.com, Usrnm: Ro, pswrd: RoSin	User will be signed up. Redirect to Login Page	COMPLETE	User added and redirected to login page
				Step 2	Enter new user Data				
				Step 3	Click on "Sign up" button				
TC_BB_Func_02	Can an existing user login to system	POSITIVE	User should be already registered	Step 1	Go to Login Page	Username: Ro, Password: RoSin	User will be logged in. Redirect to Student Dashboard	COMPLETE	Student Dashboard is displayed
				Step 2	Enter Username and Password				
				Step 3	Click on "Login" button				
TC_BB_Func_03	Admin Login to System	POSITIVE	App should be running. Admin should be defined	Step 1	Go to Login Page	Username: Administrator, Password: IndianArmy	Admin will be authenticated. Redirect to Admin Dashboard	COMPLETE	Adin Dashboard is displayed
				Step 2	Enter Admin Username and Password				
				Step 3	Click on "Login" button				
TC_BB_Func_04	Student can use AI Predictor	POSITIVE	Student should be logged in	Step 1	Go to Prediction form	GRE: 320, TOEFL : 110, CGPA: 9.7, Uni Rating : 4, Rsrch Ex: 0, SOP: 4, LOR: 5	Output prediction will be displayed. Value range: 90 - 97%	COMPLETE	Prediction: 95.667% is shown as Output
				Step 2	Enter student transcripts				
				Step 3	Click on "Predict" button				
TC_BB_Func_05	Admin can view all records	POSITIVE	Admin should be logged in	Step 1	Click on View All Tab	All records in DB	All records should be displayed	COMPLETE	Al records displayed
TC_BB_Func_06	User Info shown on Profile Page	POSITIVE	User should be logged in	Step 1	Click on Profile Tab	User info	User info should be shown	COMPLETE	User info displayed
TC_BB_Func_07	Student info updated if Model is used again	POSITIVE	User should have used model previously	Step 1	Go to predictor form	for user Ro: GRE: 299, TOEFL : 99, CGPA: 9.9, Uni Rating : 3, Rsrch Ex: 0, SOP: 4, LOR: 4	Output Prediction will be displayed. Values updated in Profile	COMPLETE	User data updated in profile pge
				Step 2	Fill in new transcripts data				
				Step 3	Click on "Predict" button				
				Step 4	Click on Profile tab				
TC_BB_Func_08	Analysis Tab is working	POSITIVE	User should be logged in	Step 1	Click on Analysis Tab	None	All Analysis pages can be displayed	COMPLETE	All Analysis pages can be displayed
				Step 2	Choose the page from dropdown				
TC_BB_Func_09	Queries Tab is working	POSITIVE	User should be loggen in	Step 1	Click on Queries Tab	None	All Query pages can be displayed	COMPLETE	All Queries pages can be displayed
				Step 2	Choose the page from droopdown				

Non-Functional Testing

Non-Functional Testing involves testing of non-functional requirements such as Load Testing, Stress Testing, Security, Volume, Recovery Testing, etc. The objective of NFT testing is to ensure whether the response time of software or application is quick enough as per the business requirement.

CHAPTER - 7

RESULT AND OUTPUT

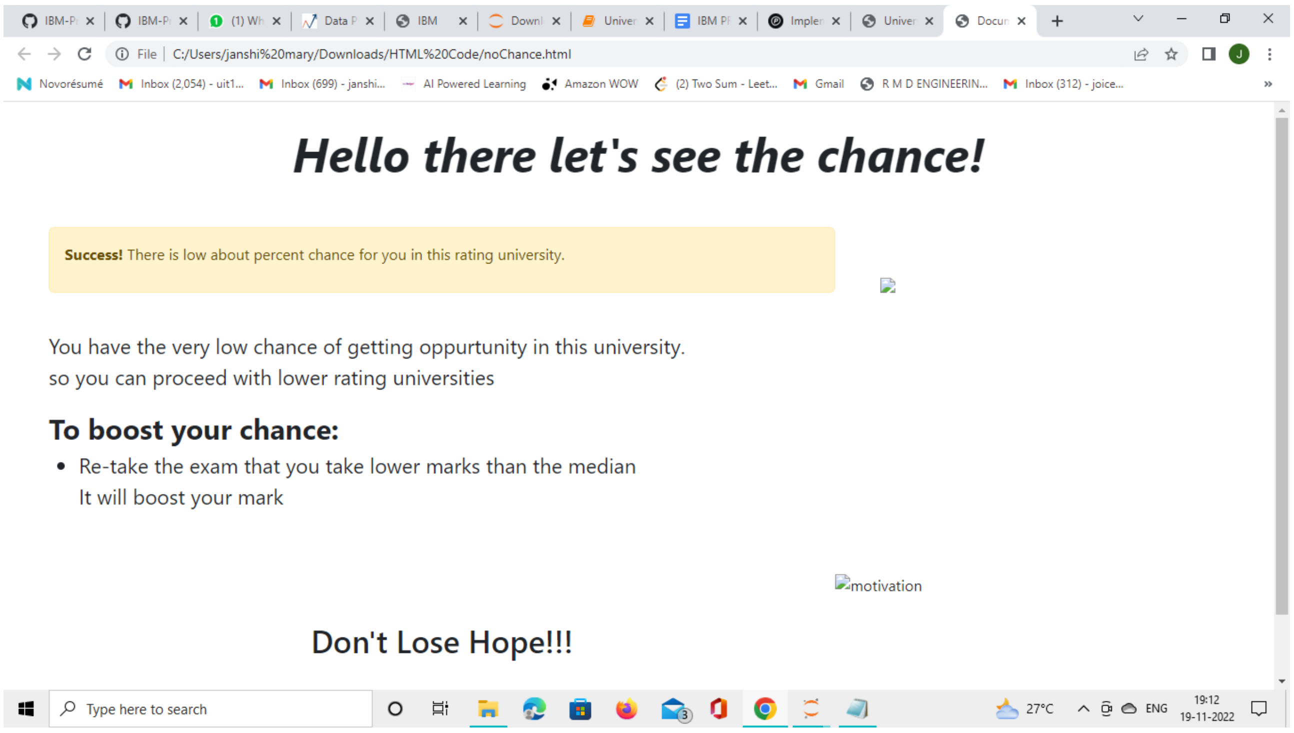


University Eligibility Predictor

This website is used to check the Eligibility of a student to study in the top university

Enter the details

GRE Score:	250 to 340
TOFEL Score:	50 to 120
University Rating:	1 to 5
SOP:	1 to 5
LOR:	1 to 5
CGPA:	5 to 10



Hello there let's see the chance!

Success! There is low about percent chance for you in this rating university.

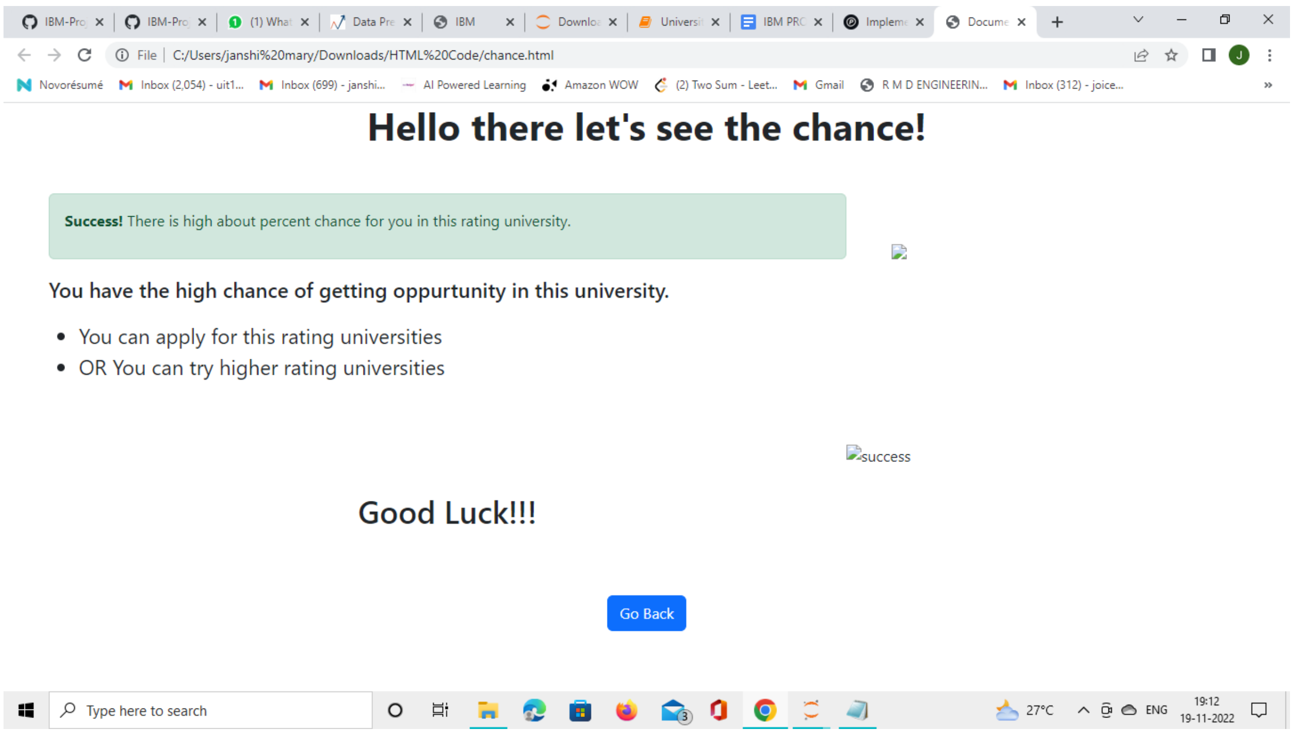
You have the very low chance of getting oppurtunity in this university.
so you can proceed with lower rating universities

To boost your chance:

- Re-take the exam that you take lower marks than the median
  It will boost your mark

motivation

Don't Lose Hope!!!



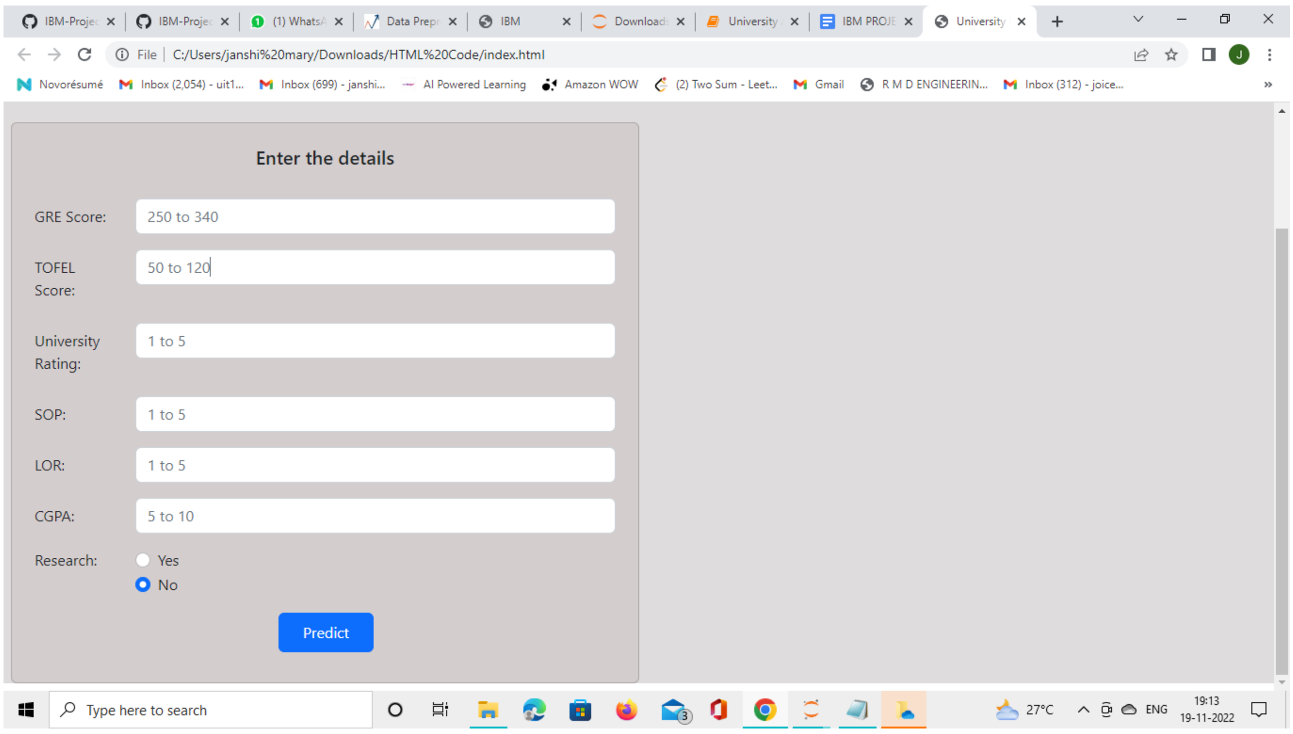
Hello there let's see the chance!

Success! There is high about percent chance for you in this rating university.

You have the high chance of getting oppurtunity in this university.

- You can apply for this rating universities
- OR You can try higher rating universities

Good Luck!!!



Enter the details

GRE Score:	250 to 340
TOFEL Score:	50 to 120
University Rating:	1 to 5
SOP:	1 to 5
LOR:	1 to 5
CGPA:	5 to 10
Research:	Yes [✓ No]

CHAPTER - 8

CONCLUSION

The main goal of this work is to create a Machine Learning model which could be used by students who want to pursue their education in the US. Many machine learning algorithms were utilized for this research. Linear Regression model compared to other ones. Students can use the model to assess their chances of getting admission into a particular university
with an average accuracy of 79 percent. A GUI was developed to make the program, from a non-technical perspective, usable and user-friendly. Using node-red the user interface was developed. The ultimate goal of research will be accomplished successfully, as the system allows students to save the lot of time and money that they would spend on educational mentors and application fees for colleges where they have less chances of getting admissions. The main limitation of this research is that we developed models based solely on data from Indian Students studying Masters in Computer Science in the United States, we considered only a few universities with different rankings. More information relating to new colleges and courses can be added to the curriculum in the future. The system may also be modified to a web-based application by making node-red modifications. To solve the problem, it is possible to test other classification algorithms if they have higher accuracy scores than the current algorithm, the framework can be easily modified to support the new algorithm by changing the server code in the Node Red. Finally students can have an open source machine Learning model which will help the students to know their chance of admission into a particular university with high accuracy.

CHAPTER - 9

REFERENCES

Abdul Fatah S; M, A. H. (2012). Hybrid Recommender System for Predicting College Admission, pp. 107–113. Bibodi, J., Vadodaria, A., Rawat, A. and Patel, J. (n.d.).

Admission Prediction System Using Machine Learning. Eberle, W., Simpson, E., Talbert, D., Roberts, L. and Pope, A. (n.d.).

Using Machine Learning and Predictive Modeling to Assess Admission Policies and Standards. Jamison, J. (2017).

Applying Machine Learning to Predict Davidson College ’ s Admissions Yield, pp. 765–766. Mane, R. V. (2016).

Predicting Student Admission decisions byAssociation Rule Mining with Pattern Growth Approach, pp. 202–207.

Data Cleaning and Analytics, Machine Learning https://archieve.ics.uci.edu/ml/index.php.

DataVisualization,MachineLearning
https://www.analyticsvidhya.com/blog/2017/09/common-machin
e-learning-algorithms/