|  |
| --- |
| **UNIVERSITY ADMIT ELIGIBILTY PREDICTOR** |

https://lh4.googleusercontent.com/J-Ga2uA362Jb_YAXEiRwTgW7LNVnX55djksJ3FpWUOuJBxZtqZ03jYGCj555XjxUr-NY_kye1wIz5p6dQ6oQbJLTJRDJ6UwwtPkEQz0lXBgeUFXLo66Au_rKTUGtAnXqnUkk_txLG8_q_sC65CLPD66ozlco8hV9KjODgpDkuqXJsV3TYqqDQlOFeSkMsw

|  |
| --- |
| **NALAIYA THIRAN PROJECT BASED LEARNING**  **On** |

|  |
| --- |
| **PROFESSIONAL READINESS FOR INNOVATION,**  **EMPLOYABILITY AND ENTREPRENEURSHIP** |

**A PROJECT REPORT**

**TEAM ID: PNT2022TMID10342**

|  |  |
| --- | --- |
| APURVA SV | 720819104009 |
| KESAVAN B | 720819104056 |
| SUDHAKAR M | 720819104110 |
| THILLAI AISHWARYA S | 720819104117 |

|  |
| --- |
| **BACHELOR OF ENGINEERING**  **IN**  **COMPUTER SCIENCE ENGINEERING**  **HINDUSTHAN INSTITUTE OF TECHOLOGY**  Approved by AICTE, New Delhi, Accredited with ‘A’ Grade by NAAC  **(Affiliated to Anna University, Chennai)** COIMBATORE – 641 032November 2022 |

#### **TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE.NO** |
| 1 | INTRODUCTION | 4 |
|  | 1.1 Project Overview | 4 |
|  | 1.2 Purpose | 6 |
| 2 | LITERATURE SURVEY | 7 |
|  | 2.1 Existing Problem | 9 |
|  | 2.2 References | 10 |
|  | 2.3 Problem Statement Definition | 11 |
| 3 | IDEATION & PRPOSED SOLUTION | 13 |
|  | 3.1 Empathy Map Canvas | 13 |
|  | 3.2 Ideation & Brainstorming | 14 |
|  | 3.3 Proposed Solution | 16 |
|  | 3.4 Problem Solution Fit | 17 |
| 4 | REQUIREMENT ANALYSIS | 18 |
|  | 4.1 Functional Requirement | 18 |
|  | 4.2 Non- Functional Requirement | 19 |
| 5 | PROJECT DESIGN | 20 |
|  | 5.1 Data Flow Diagrams | 20 |
|  | 5.2 Solution & Technical Architecture | 21 |
|  | 5.3 User Stories | 22 |
| 6 | PROJECT PLANNING & SCHEDULING | 23 |
|  | 6.1 Sprint Planning & Estimation | 23 |
|  | 6.2 Sprint Delivery Schedule | 23 |
| 7 | CODING & SOLUTIONING | 24 |
|  | 7.1 Feature 1 | 24 |
|  | 7.2 Feature 2 | 30 |
|  | 7.3 Database Schema | 32 |
| 8 | TESTING | 33 |
|  | 8.1 Testcases | 33 |
|  | 8.2 User Acceptance Testing | 35 |
| 9 | RESULTS | 37 |
|  | 9.1 Performance Metrics |  |
| 10 | ADVANTAGES & DISADVANTAGES | 38 |
| 11 | CONCLUSION | 39 |
| 12 | FUTURE SCOPE | 40 |
| 13 | APPENDIX | 40 |
|  | Source Code |  |
|  | Github & Project Demo link |  |

**1.INTRODUCTION**

A machine leaning based system built on a linear regression model using the data set available on kaggle for predicting chances of admissions for Indian students hoping to pursue their graduate degrees. The scope of this project is a web academic data and get predictions of their chances of admission in the university tier of their choosing. It also provides an analysis based on the data set used that shows how different parameters affect the chances of admissions.

University Admit Eligibility Predictor provides a solution to the problem of getting into an university. It is a single platform that documents all the requirements as well as the different tiers of universities. The website 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 admission does a student’s current grades and other academic transcripts allow them in the tier of universities of their choice.

* 1. **Project Overview**

University Admit Eligibility Predictor is 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. This document describes the scope, objectives and goals of the system.

OBJECTIVES:

● To build a model which gives accurate results

● To build a user-friendly web-based application for the users

● To create a simple user interface getting all the required input from the users

● To integrate the model with the web application using flask

● To deploy the model on IBM cloud

DATASET:

The dataset is taken from Kaggle. The dataset contains several parameters which are considered important during the application for Masters Programs.

● GRE Scores (out of 340)

● TOEFL Scores (out of 120

● University Rating (out of 5)

● Statement of Purpose and Letter of Recommendation Strength (out of 5)

● Undergraduate GPA (out of 10)

● Research Experience (either 0 or 1)

● Chance of Admit (ranging from 0 to 1)

APPROACH:

The dataset is trained using different models and the model which gives high accuracy is considered. Random forest model gives the highest accuracy of 94%.

* 1. **Purpose**

A web-based application in which students can register and enter their grades to determine whether or not they will be admitted to a university with a specific ranking can help the students proceed further. The analysis can help students who are currently preparing for exams or who have received their results gain a better understanding. It helps the students to know the weightage of each category of marks and to concentrate more in it.

**2. LITERATURE SURVEY**

1. “Graduate Admission Chance Prediction Using Deep Neural

Network” by M. Omaer Faruq Goni, A. Matin, T. Hasan, M. Abu Ismail Siddique, O. Jyoti and F. M. Sifnatul Hasnain, at *(*2020 IEEE).

Every year many students apply for graduate admission to different universities. To select an applicant, each university has different selection criteria such as GRE score, CGPA, research background, statement of purpose, letter of recommendation, university rating etc. There are some web applications as well as some consultancy services for suggesting the appropriate university based on students’ portfolio. These help to give an idea which universities should be applied for admission. But they have limitations because humans are incapable of considering all the conditions and universities. Moreover, web applications have accuracy problems. In this study, we have proposed a deep neural network (DNN) to predict the

chance of getting admitted to a university according to the student’s portfolio. All the selection criteria are considered here to predict the chance of admission. The DNN model has been compared with existing methods in terms of different performance metrics including mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), R- squared score. It has shown the most promising result that includes R- squared score of 0.8538 and MSE of 0.0031. The proposed method has also outperformed all the existing methods in each benchmark.

2. “College Admission Prediction using Ensemble Machine Learning Models” by

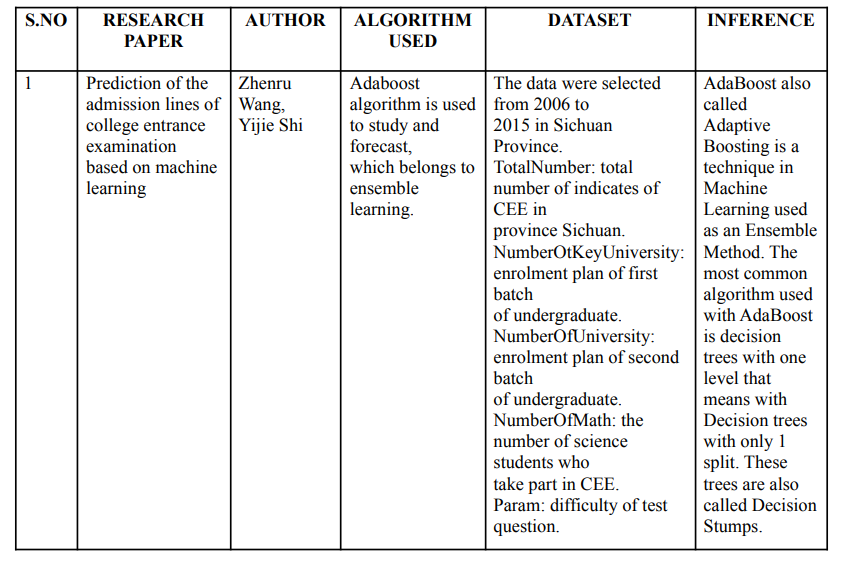
Vandit Manish Jain, Rihaan Satia at (2021 IRJET).

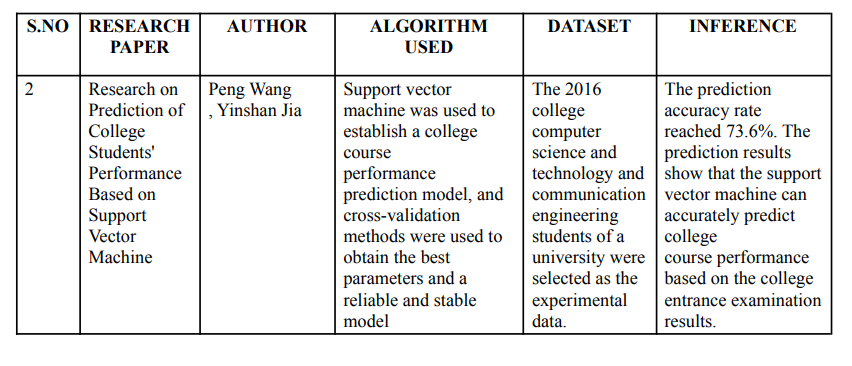
The aim to build a model that can help students to pick the right universities based on their profiles. We can judge across a wide variety of domains that include MS (international), M.Tech (India) and MBA (India and International). For the accurate predictions we plan on training a machine learning model in order to provide results. The dataset contains information on the student profile and the university details with a field detailing if the admission was positive or not. Various algorithms have been used i.e., Ensemble Machine Learning and the predictions have been compared using key performance indicators (KPIs). The model performing the best is then used to evaluate the dependent variable i.e. The chances of admit to a university. The chances of admit variable is a variable ranging from 0 to 1 which equates to the predicted probability of successful acceptance to a university. We also aim to create a portal which filters and then provides a list of universities that fall into the profile's acceptance range.

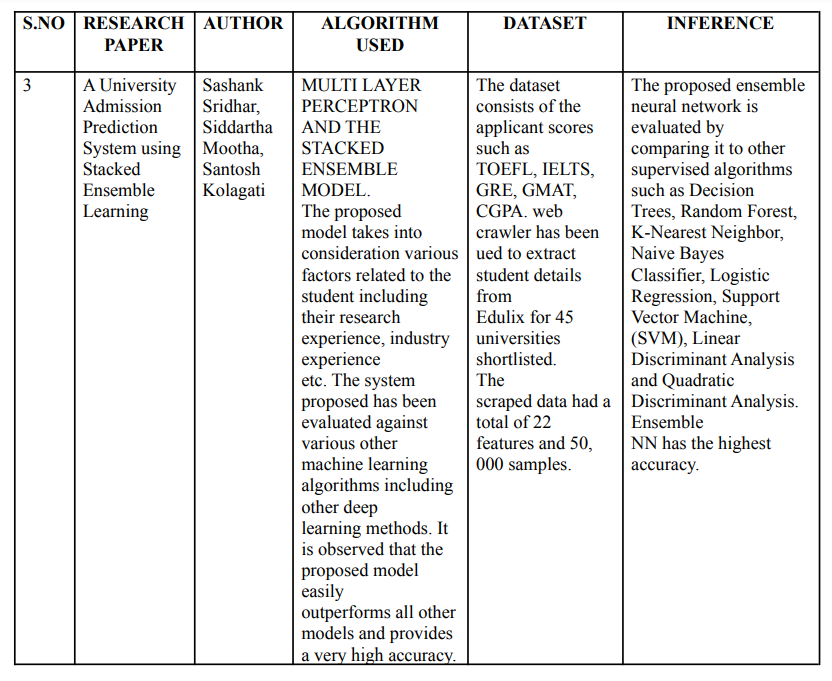
3. “Using Data Mining Techniques to Predict Student Performance to Support Decision Making in University Admission Systems” by Hanan Abdullah Mengash at (2020 IEEE).

An admissions system based on valid and reliable admissions criteria is very important to select candidates likely to perform well academically at institutions of higher education. This study focuses on ways to support universities in admissions decision making using data mining techniques to predict applicants’ academic performance at university. A data set of 2,039 students enrolled in a Computer Science and Information College of a Saudi public university from 2016 to 2019 was used to validate the proposed methodology. The results demonstrate that applicants’ early university performance can be predicted before admission based on certain pre-admission criteria (high school grade average, Scholastic Achievement Admission Test score, and General Aptitude Test score). The results also show that Scholastic Achievement Admission Test score is the preadmission criterion that most accurately predicts future student performance. Therefore, this score should be assigned more weight in admissions systems. We also found that the Artificial Neural Network technique has an accuracy rate above 79%, making it superior to other classification techniques considered (Decision Trees, Support Vector Machines, and Naïve Bayes).

**2.1 Existing Problem**

****

****

****

**2.2 References**

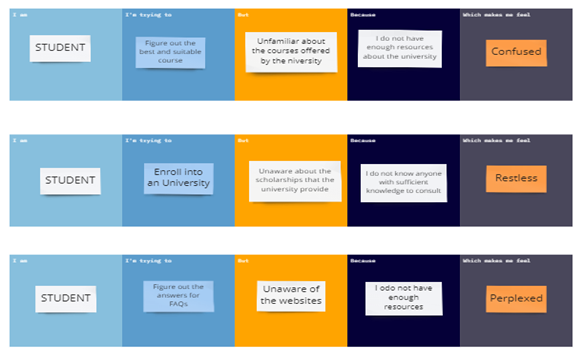
* <https://ieeexplore.ieee.org/document/7924718>
* <https://ieeexplore.ieee.org/document/9213205>
* <https://www.researchgate.net/publication/341740217_ENGINEERING_TECHNOLOGY_ADMISSION_ANALYSIS_AND_PREDICTION>
* <https://www.jncet.org/Manuscripts/Volume-8/Issue-4/Vol-8-issue-4-M-32.pdf>

**2.3 Problem Statement**

Customer Problem Statement:

Problem Statement allows us to understand the problem from customer’s point of view. It helps us to focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you’ll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.



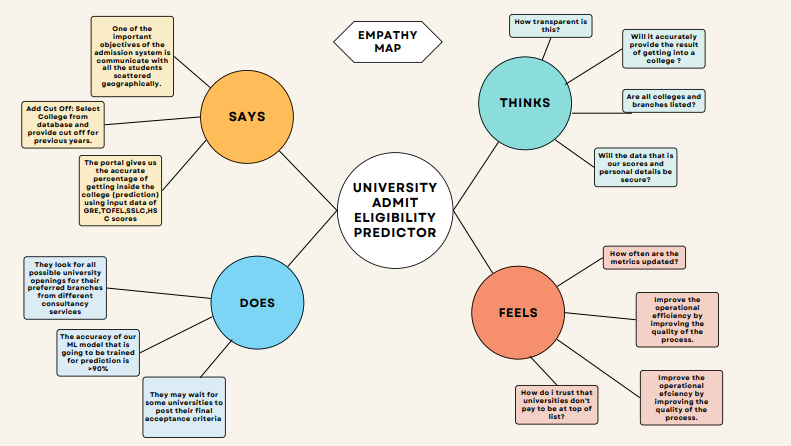


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Problem Statement (PS)** | **I am (Customer)** | **I’m trying to** | **But** | **Because** | **Which makes me feel** |
| PS-1 | Student | Predict the chances of getting into an university | Unaware of the eligibility criteria of the university | I do not have anyone with sufficient knowledge to consult | Anxious |
| PS-2 | Student | Find out the best universities based on current year’s admission criteria | Unaware about the process | I do not have anyone with sufficient knowledge to consult | Depressed |
| PS-3 | Student | Figure out the suitable course for me | Unfamiliar about the courses offered by the university | I do not have enough resources about the university | Confused |
| PS-4 | Student | Enrol into a university | Unaware about the scholarships that the universities provide | I do not have anyone with sufficient knowledge to consult | Restless |
| PS-5 | Student | Figure out the answers for FAQs | Unaware of the website | I do not have enough resources about the university | Perplexed |

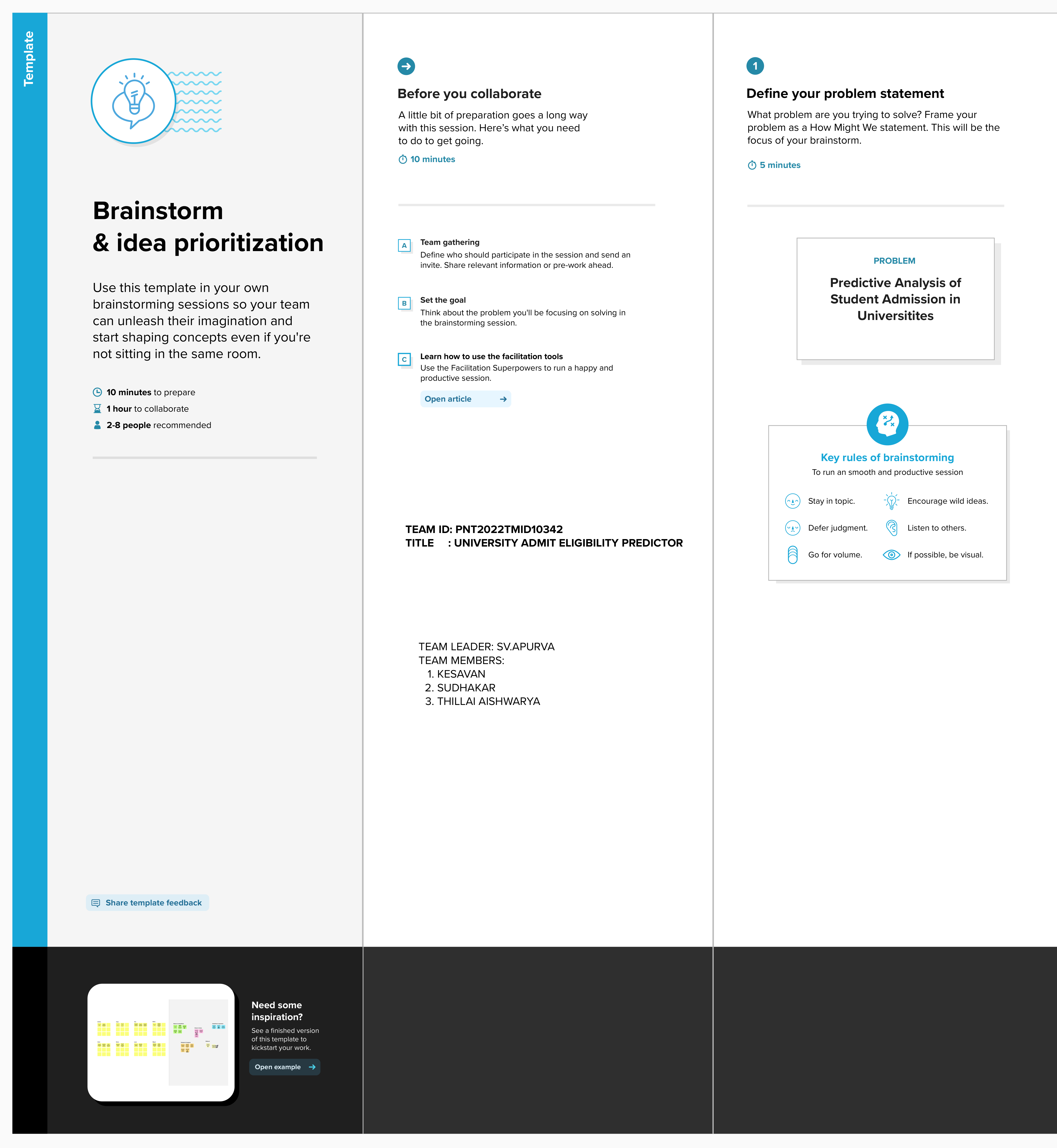
**3.IDEATION & PROPOSED SOLUTION**

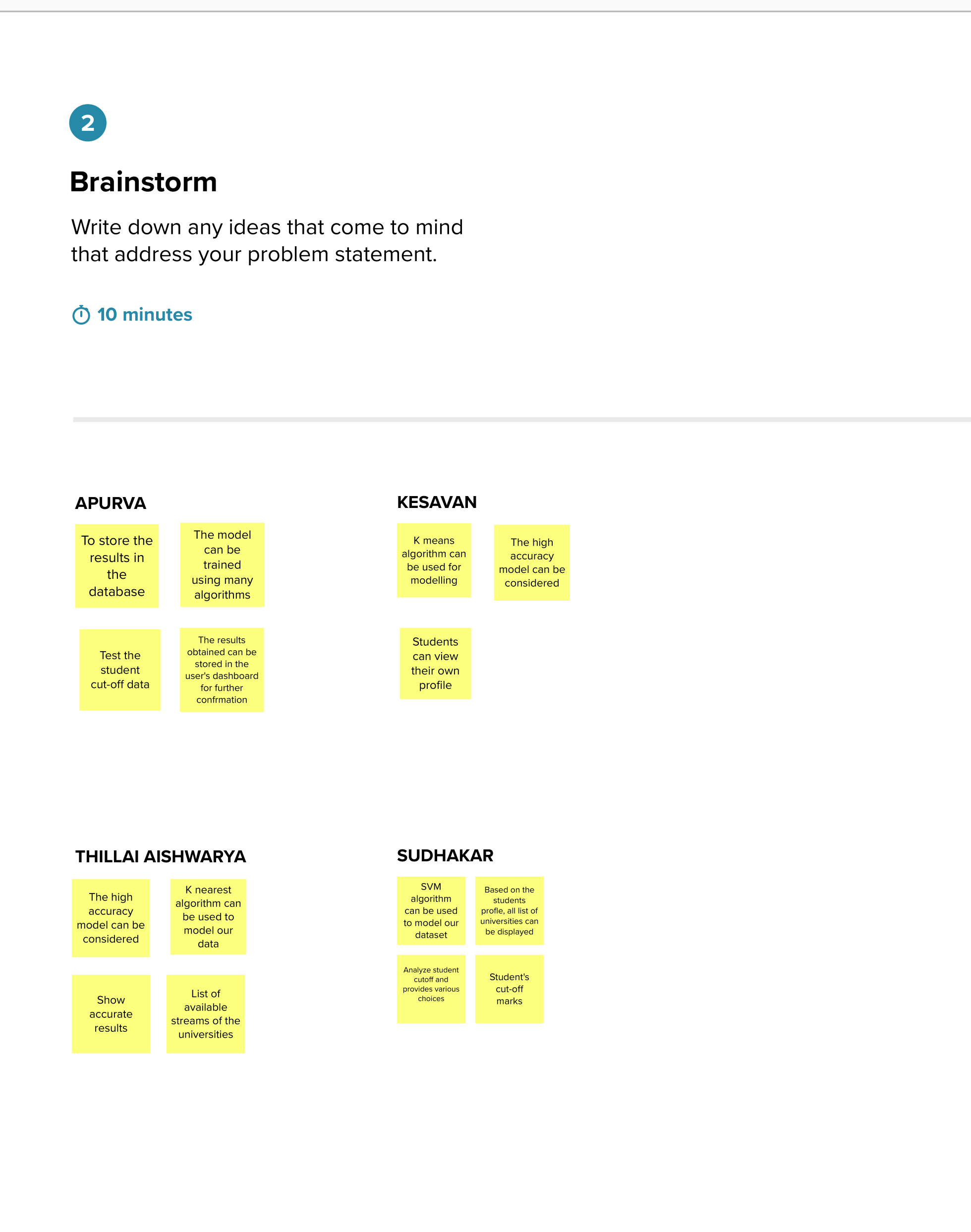
**3.1Empathy Map Canvas**

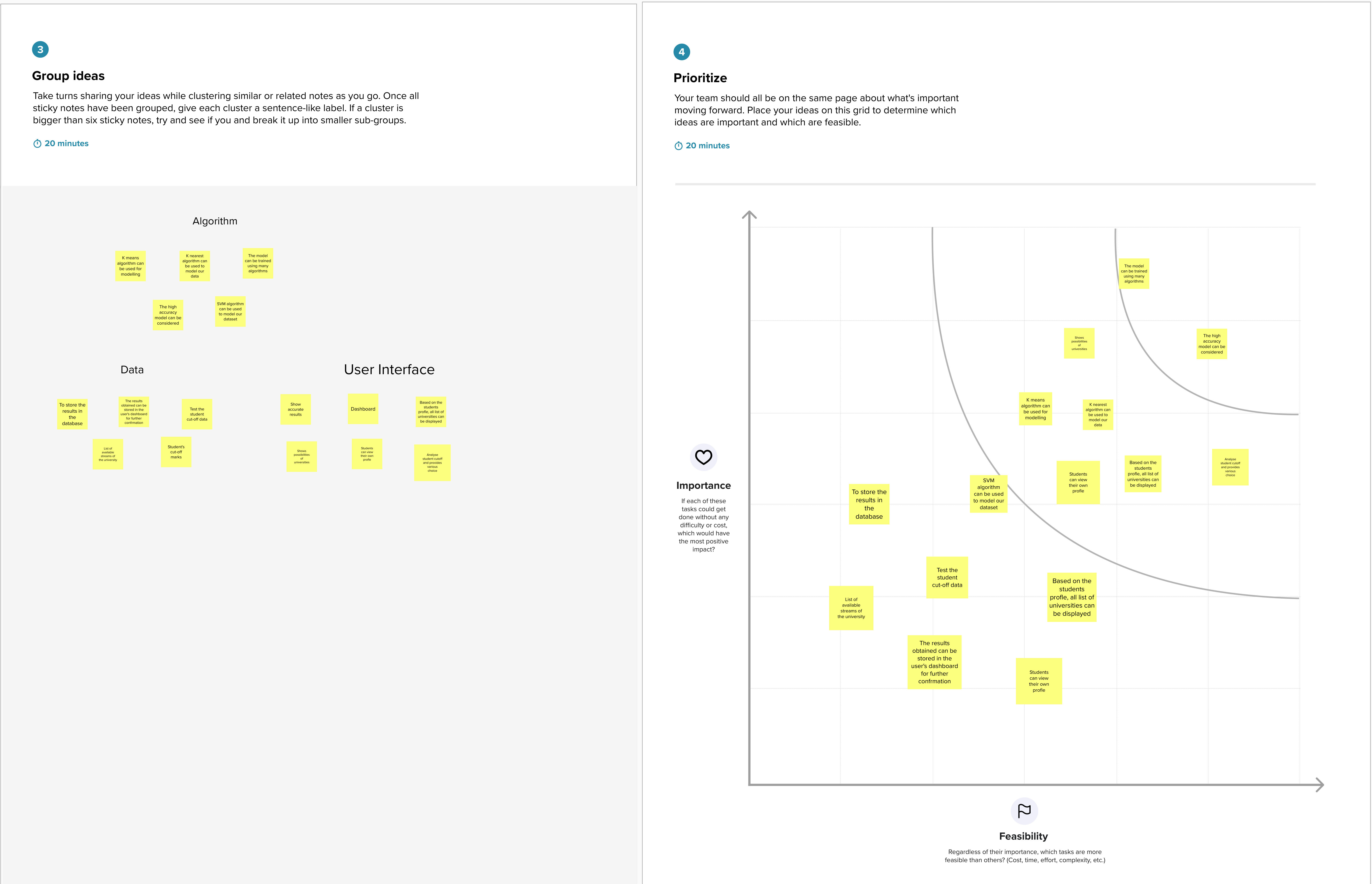
An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment.



**3.2 Ideation & Brainstorming**



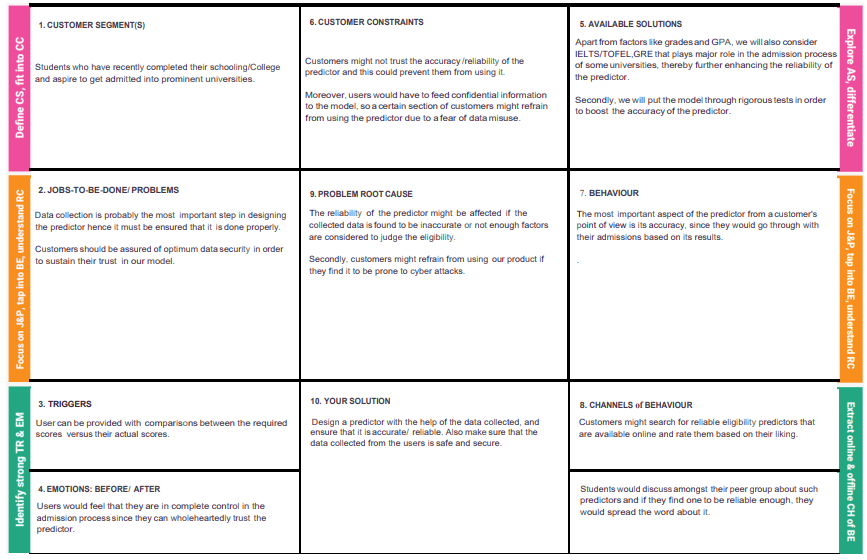




**3.3** **Proposed Solution:**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Parameter** | **Description** |
|  | Problem Statement (Problem to be solved) | Students do not have much idea about the procedures, requirement and details of the universities they want to join, so they seek help from various educational consultancies to help them secure admission in the universities based on their profile, for which the students are supposed pay a hefty amount as consultancy fee. |
|  | Idea / Solution description | Providing an accurate prediction for the student’s admission into the university of their choice based on various parameters like IELTS, GRE, Academic Performance, etc. |
|  | Novelty / Uniqueness | It seems there are no web applications for predicting the eligibility criteria of a student for getting in to their dream university and also provide personalized insights on specific areas where they can improve. |
|  | Social Impact / Customer Satisfaction | It helps student in making the right decision for choosing the universities. It cuts the cost of consultancy services by creating a direct connection between students and universities. |
|  | Business Model (Revenue Model) | Universities are under immense pressure to admit more students and ensure student success. To overcome this pressure, they can make use of predictive models which help them to ease the intake process of students and improve efficiency. |
|  | Scalability of the Solution | Further to reduce the immense pressure faced by the students to get admitted in a university, the model can also be evolved to consider university specific examinations and to maintain the latest eligibility criteria. |

**3.4Problem Solution Fit**



**4. REQUIREMENT ANALYSIS**

**4.1 Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Form  Registration through Gmail |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | User Requirements | All the needed files are been asked to feed in the website. By having the file, it will do all the pre-processing and shows all the required information to the student(user). The information includes the list of all the possible universities and streams. |
| FR-4 | User Details | Has to feed some documents  1. Score Sheets  2. Letter of Recommendation (LOR  ) 3. Statement of Purpose (SOP)  4. Curriculum Vitae (CV) |

**4.2 Non-functional Requirements:**

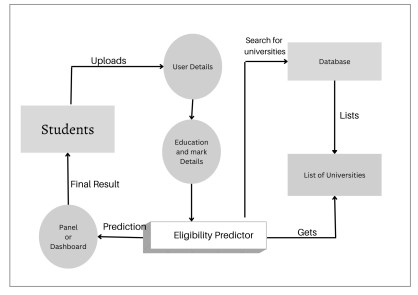
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | • Our website is very user friendly so even the layman can able to access our website.  • There is no need for any pre requisite technical skill in order to access our website.  • Each and every content of the page will be in synchronous way. Thus, it will not take much time to refresh or reload. |
| NFR-2 | **Security** | • The user who is having the valid credentials can able to access our site.  • Data they are feeding into our website will not be accessed by any one of them. |
| NFR-3 | **Reliability** | • Our website is more reliable. Since nobody can able to see the data fed by the user.  • The user can get the result with higher percent of accuracy. |
| NFR-4 | **Performance** | • User can able to handle the process in our website even by having internet connection with normal  speed. There is no need of high -speed internet connection.  • Traffics can be handled effectively. |
| NFR-5 | **Availability** | There is no need of high -speed internet connection and Traffics can be handled effectively. |
| NFR-6 | **Scalability** | • Our website will be easily scalable in the case of getting increasing number of users data from our website.  • If needed we do scale up the CPU or Processer in order to speed up the processing  capability of our website.  • There by it reduces the downtime of our website |

**5. PROJECT DESIGN**

**5.1 Data Flow Diagrams**

A Data Flow Diagram (DFD) is a traditional visual representation of the 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.

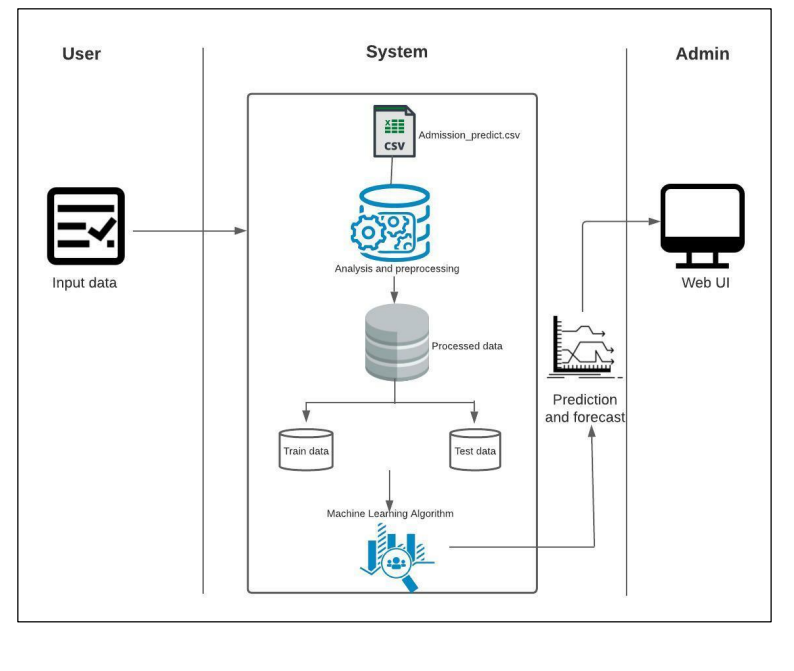


**5.2 Solution & Technical Architecture**

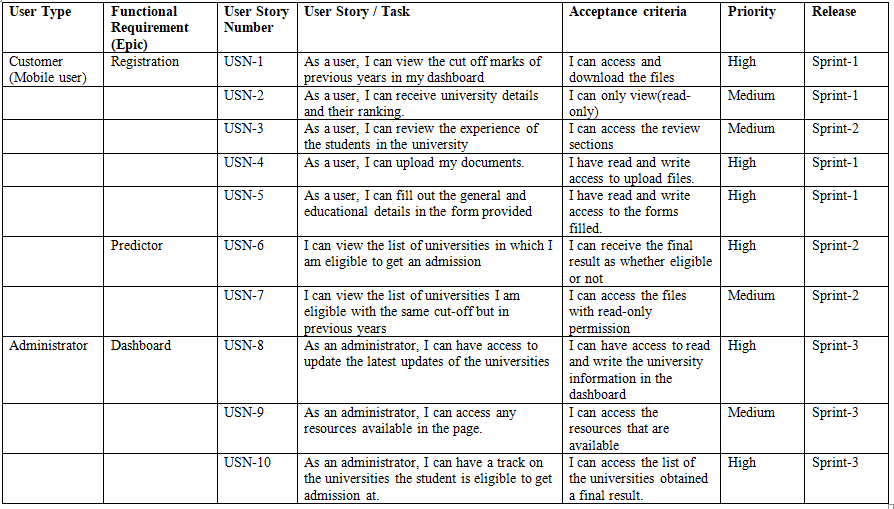
**Solution Architecture**

A Solution Architecture (SA) is an architectural description of a specific solution. Solution Architectures combine guidance from different enterprise architecture viewpoints (business, information and technical), as well as from the enterprise solution architecture (ESA). Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions.

**Technical Architecture:**

****

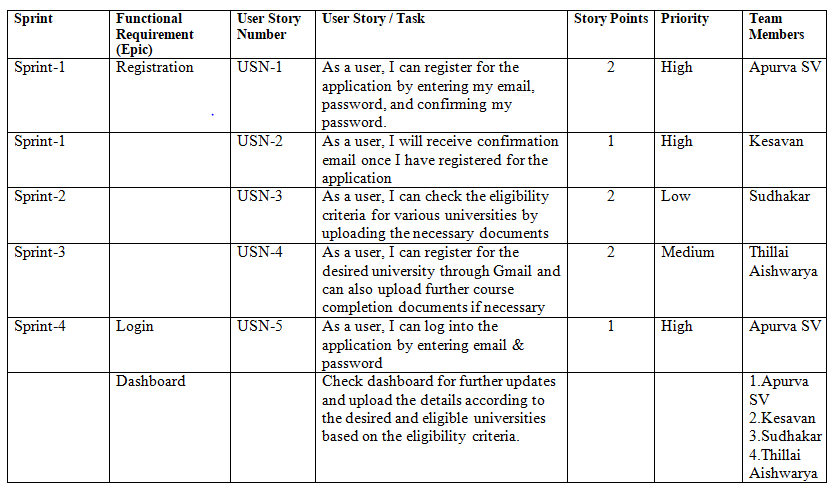
**5.3 User Stories:**



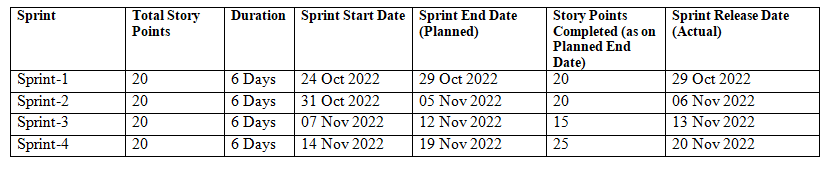
**6. PROJECT PLANNING & SCHEDULING**

**6.1** **Sprint Planning & Estimation**

Product Planning: Product Backlog, Sprint Schedule, and Estimation:



**6.2 Sprint Delivery Plan**



1. **CODING & SOLUTIONING**
   1. **Feature 1**

Fundamental Regression Model

Comparing some fundamental regression methods on Admission Prediction Data: There are lots of regression methods to predict data such as Linear Regression, Decision Tree Regression and so on. In this notebook, we will try to predict the "Chance of Admit" data by using some of fundamental regression methods, which are Linear Regression, Multiple Linear Regression, Decision Tree Regression and Random Forest Regression.

**import** numpy **as** np

**import** pandas **as** pd

**import** seaborn **as** sns

**import** matplotlib.pyplot **as** plt

**import** warnings

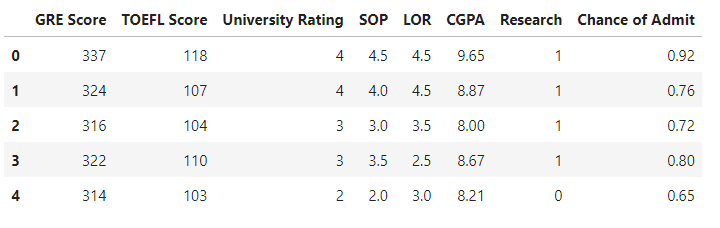
warnings**.**filterwarnings('ignore')

**%matplotlib** inline

df **=** pd**.**read\_csv("Data/Admission\_Predict\_Ver1.1.csv")

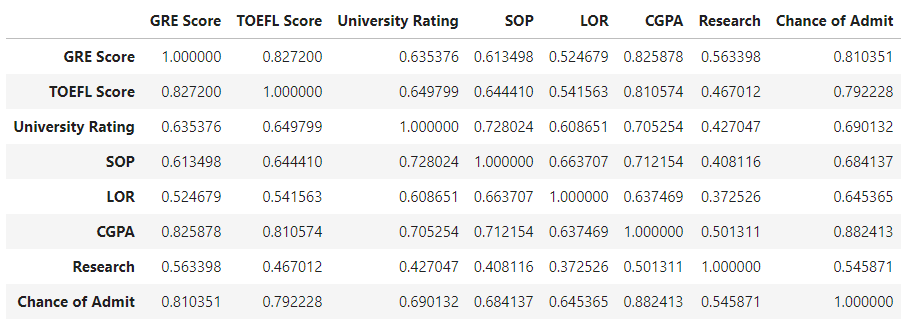
df **=** df**.**iloc[:,1:]

df**.**head()



corr\_matrix **=** df**.**corr()

corr\_matrix



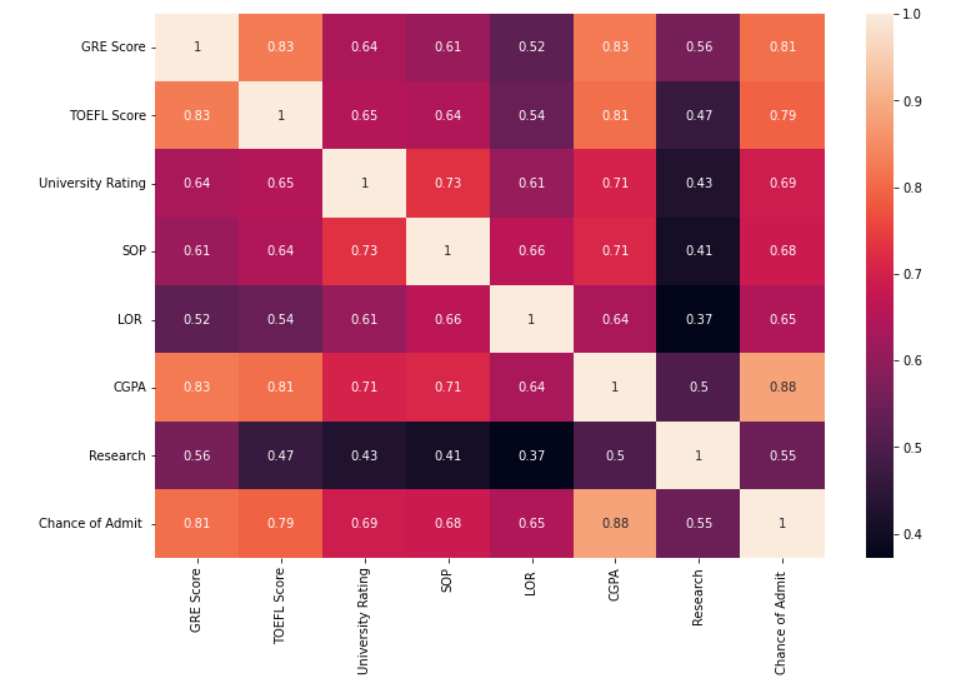
When looking at the data and correlation matrix, it seems that "Chance of Admit" values depend on lots of variables. To observe the effects of columns on "Chance of Admit" values in detail, so creating a correlation graph.

*#plotting the correlation matrix as a heatmap*

fig **=** plt**.**figure(figsize**=**(12,8))

sns**.**heatmap(corr\_matrix,annot**=True**)

plt**.**show()



As seen, there are direct proportions between "Chance of Admit" data and the other columns. In order to observe these direct proportions more clearly, we will visualize the data.

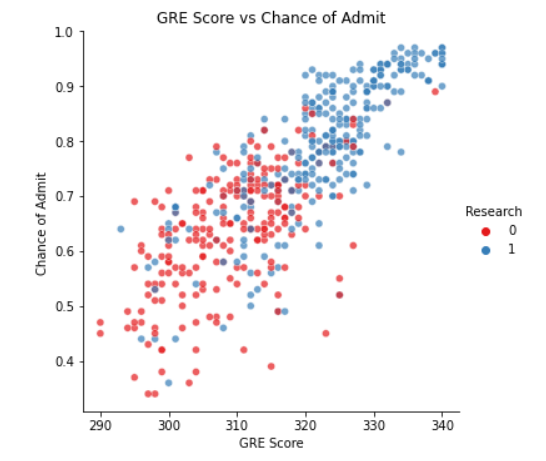
*#plotting data which have high correlation*

sns**.**relplot(data**=**df,x**=**"GRE Score",y**=**"Chance of Admit ",hue**=**"Research",

palette**=**"Set1",alpha**=**0.7)

plt**.**title("GRE Score vs Chance of Admit")

plt**.**show()

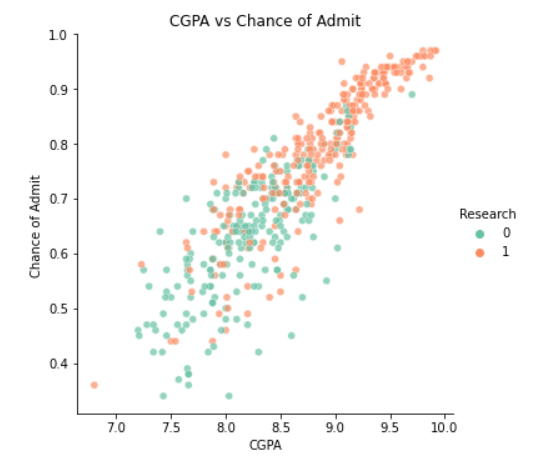


sns**.**relplot(data**=**df,x**=**"CGPA",y**=**"Chance of Admit ",hue**=**"Research",

palette**=**"Set2",alpha**=**0.7)

plt**.**title("CGPA vs Chance of Admit")

plt**.**show()

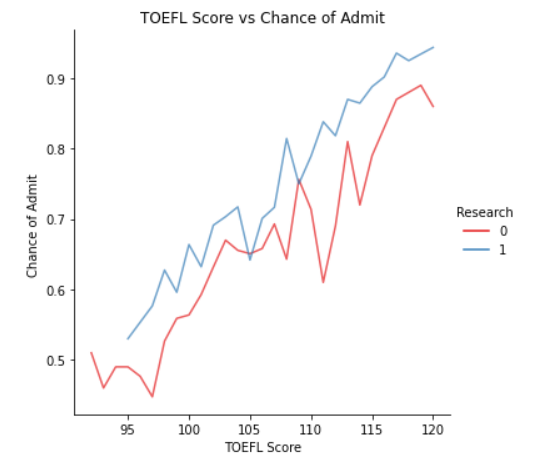


sns**.**relplot(data**=**df,x**=**"TOEFL Score",y**=**"Chance of Admit ",hue**=**"Research",

palette**=**"Set1",alpha**=**0.7,kind**=**"line",ci**=None**)

plt**.**title("TOEFL Score vs Chance of Admit")

plt**.**show()

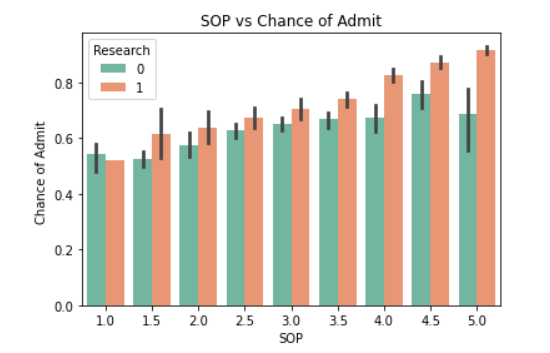


sns**.**barplot(data**=**df,x**=**"SOP",y**=**"Chance of Admit ",

palette**=**"Set2",hue**=**"Research")

plt**.**title("SOP vs Chance of Admit")

plt**.**show()

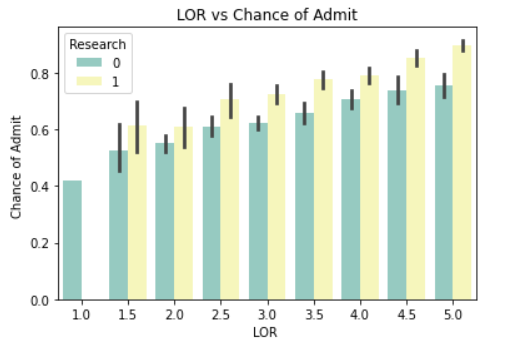


sns**.**barplot(data**=**df,x**=**"LOR ",y**=**"Chance of Admit ",

palette**=**"Set3",hue**=**"Research")

plt**.**title("LOR vs Chance of Admit")

plt**.**show()

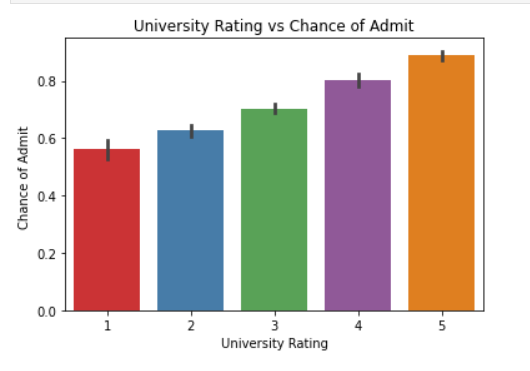
****

sns**.**barplot(data**=**df,x**=**"University Rating",y**=**"Chance of Admit ",

palette**=**"Set1")

plt**.**title("University Rating vs Chance of Admit")

plt**.**show()

****

It was said that there are direct proportions between the columns and "Chance of Admit" data. Also, when observing the graphs, there are linear relationships between them. Therefore, using Linear Regression method may be suitable on the data, but since "Chance of Admit" data depends on more than one varible, it is more appropriate to imply Multiple Linear Regression method instead of Linear Regression method.

## Importing the required libraries for regression analysis:

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.linear\_model **import** LinearRegression

**from** sklearn.tree **import** DecisionTreeRegressor

**from** sklearn.ensemble **import** RandomForestRegressor

**from** sklearn.metrics **import** r2\_score

## Spliting the dataset into training and testing data

x **=** df[["GRE Score","TOEFL Score","University Rating","SOP","LOR ","CGPA"]]

y **=** df["Chance of Admit "]**.**values**.**reshape(**-**1,1)

x\_train, x\_test, y\_train, y\_test **=** train\_test\_split(x,y,test\_size**=**0.2,random\_state**=**42)

## Multiple Linear Regression

*#implying multiple linear regression and determining its score*

multiple\_lin\_reg **=** LinearRegression()

multiple\_lin\_reg**.**fit(x\_train,y\_train)

y\_pred\_mlr **=** multiple\_lin\_reg**.**predict(x\_test)

r2\_score\_mlr **=** r2\_score(y\_test,y\_pred\_mlr)

print("Mutiple Linear Regression's Score = {:.3f}"**.**format(r2\_score\_mlr))

Mutiple Linear Regression's Score = 0.813

## Decision Tree Regression

*#implying decision tree regression and determining its score*

tree\_reg **=** DecisionTreeRegressor()

tree\_reg**.**fit(x\_train,y\_train)

y\_pred\_tree **=** tree\_reg**.**predict(x\_test)

r2\_score\_tree **=** r2\_score(y\_test,y\_pred\_tree)

print("Decision Tree Regression's Score = {:.3f}"**.**format(r2\_score\_tree))

Decision Tree Regression's Score = 0.551

## Random Forest Regression

*#implying random forest regression and determining its score*

ran\_for\_reg **=** RandomForestRegressor(n\_estimators**=**100,random\_state**=**42)

ran\_for\_reg**.**fit(x\_train,y\_train)

y\_pred\_rfr **=** ran\_for\_reg**.**predict(x\_test)

r2\_score\_rfr **=** r2\_score(y\_test,y\_pred\_rfr)

print("Random Forest Regression's Score = {:.3f}"**.**format(r2\_score\_rfr))

Random Forest Regression's Score = 0.786

* 1. **Feature 2**

import pandas as pd

from flask import Flask, request, jsonify, render\_template,redirect,url\_for

import pickle

app = Flask(\_\_name\_\_,template\_folder='Template')

model = pickle.load(open('D:/IBM\_UAEP/model.pkl', 'rb'))

@app.route('/')

def home():

return render\_template('index.html')

@app.route('/predict', methods=['GET','post'])

def predict():

GRE\_Score = int(request.form['GRE Score'])

TOEFL\_Score = int(request.form['TOEFL Score'])

University\_Rating = int(request.form['University Rating'])

SOP = float(request.form['SOP'])

LOR = float(request.form['LOR'])

CGPA = float(request.form['CGPA'])

Research = int(request.form['Research'])

final\_features = pd.DataFrame([[GRE\_Score, TOEFL\_Score, University\_Rating, SOP, LOR, CGPA,Research]])

predict=model.predict(final\_features)

output=predict[0]

if output > 0.5:

return redirect(url\_for('chance', percent=output\*100))

else:

return redirect(url\_for('no\_chance', percent=output\*100))

@app.route("/chance/<percent>")

def chance(percent):

return render\_template("chance.html", content=[percent])

@app.route("/nochance/<percent>")

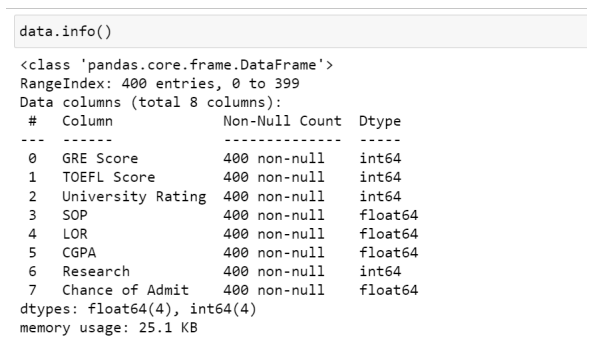
def no\_chance(percent):

return render\_template("noChance.html", content=[percent])

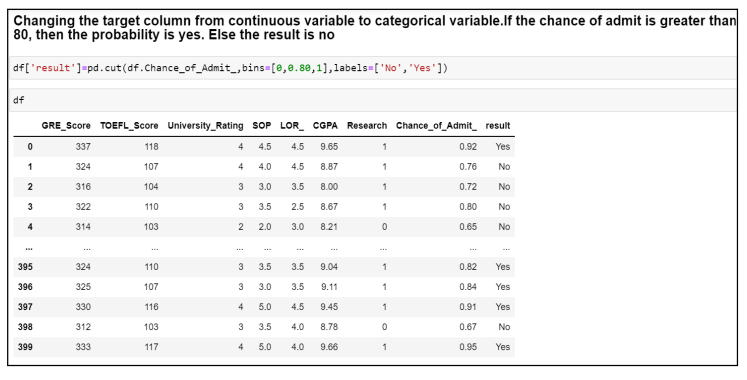
if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

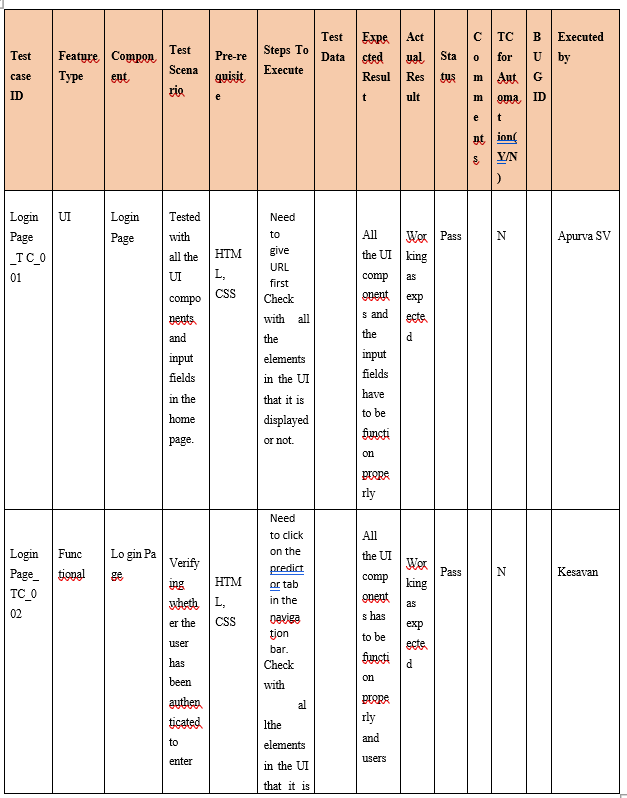
* 1. **Database Schema**

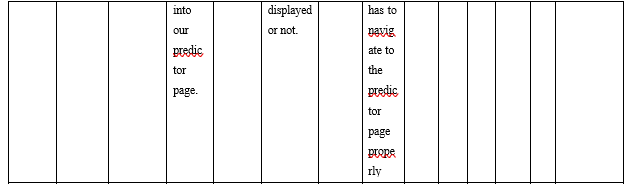
****

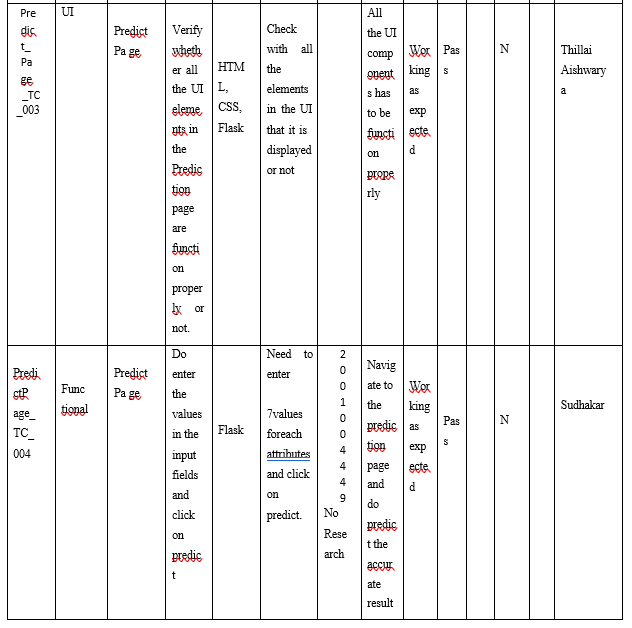
An additional column is created based on the chance of admit column values which changes the continuous value to categorical value giving Yes/No output predictions.

****

1. **TESTING**
   1. **Test Cases**

****

****



****

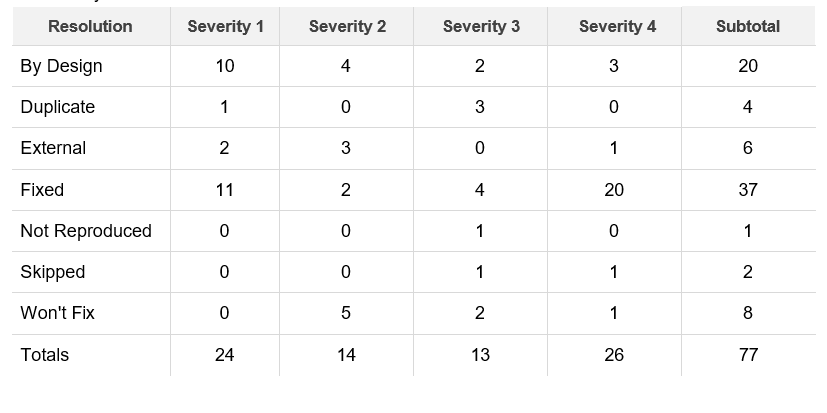
**8.2User Acceptance Testing**

1.Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the University Admit Eligibility Predictor project at the time of the release to User Acceptance Testing (UAT).

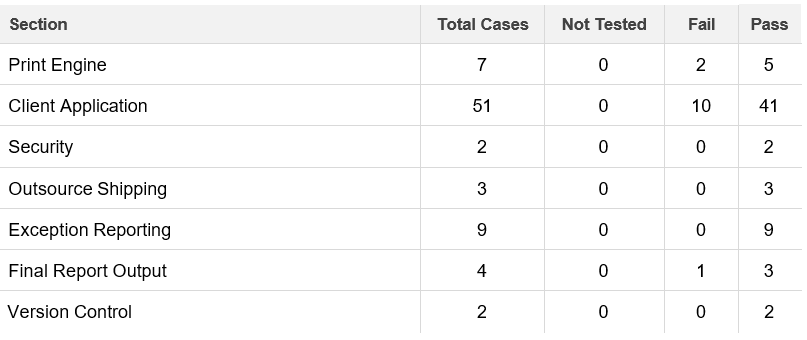
2.Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved.

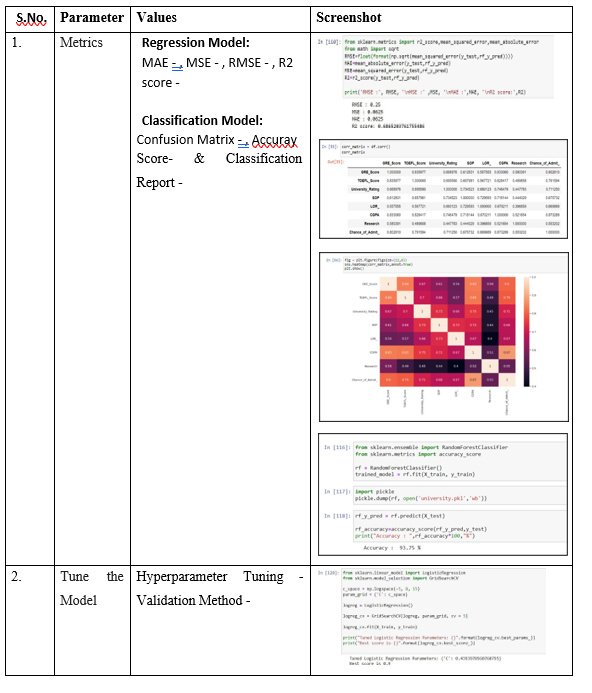


3.Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested.



1. **RESULTS**
   1. **Performance Metrics**

****

**10. ADVANTAGES AND DISADVANTAGES**

ADVANTAGES:

* It gives an overall accuracy of 94%, which is really high.
* The dataset consists of all possible attributes needed for prediction.
* Confidence booster if results are positive.
* Students can also change their scores to see how they affect the overall prediction results and focus more on that area.
* It is fast, efficient and reliable.
* Avoids Data redundancy and inconsistency.
* Very user-friendly.
* Easy accessibility of data.

DISADVANTAGES:

* The model is built in such a way that the prediction is positive only if the chance of admit percent is greater than 80%. Even if the result is 79%, the prediction would be NO.
* The complexity of the examinations is not considered. Therefore the results may vary every year for the same set of attribute.
* The dataset used for training the model is of comparatively small size. Therefore, the model cannot be relied on to take accurate real-time decisions.

**11.CONCLUSION**

The web application helps the user make wise choice of colleges for his/her option-entry. Also, the user gets an outline/rough idea of the entries they can make in the option-entry process provided by examination authority. Proposed application benefits for the student admission community that accommodates the need of students to choose the best college and helps colleges too to recognize their stand in attracting students and finer prediction implies better results for the students. The dataset is trained with different ML model. The ML models used to train our dataset are KNN, Logistic Regression, Random Forest, SVM each having accuracy of 86%, 88%, 93%, 89% respectively. Random forest algorithm is finally selected to be used in our model. The Machine Learning model is integrated using flask for our web application. At long last, understudies can have an open-source AI model which will assist the understudies with knowing their opportunity of entrance into a specific college with high exactness.

**12. FUTURE SCOPE**

A real-time project can be developed by gathering data from institutions. The data can be processed and trained using big data frameworks like spark and ML Lib can be used to train the data using different machine learning models.

**13.APPENDIX**

**Source Code**

**index.html**

<!DOCTYPE html>

<html>

<body

background="https://oge.tmu.edu.tw/admission/apply/graduation-hat-with-degree-paper

-on-a-stack-of-book/">

<head>

<style>

body {

background-repeat: no-repeat;

background-attachment: fixed;

background-size: 100% 100%;

background-image: url("admission.jpg");

height: 300;

width: 300;

}

</style>

<title>University Admit Eligibility Predictor</title>

</head>

<h2 style="color:blue"><font size="+2">UNIVERSITY ADMISSION PREDICTION

SYSTEM</font></h2>

<h3><font size="+1">Enter your details and get probability of your

admission</font></h3>

<form action="{{url\_for('predict')}}"method="POST">

<br>

<font size="+1">Enter GRE score <input id="gre" type="number" name="gre"

required="required"></font>

<br><br>

<font size="+1">Enter TOEFL score <input id="toefl" type="number"

name="toefl" required="required"></font>

<br><br>

<font size="+1">Select University number</font><br>

<input id="universityNumber" type="radio" id="1" name="universityNumber"

value="1">

<label for="1">1</label><br>

<input type="radio" id="2" name="universityNumber" value="2">

<label for="2">2</label><br>

<input type="radio" id="3" name="universityNumber" value="3">

<label for="3">3</label><br>

<input type="radio" id="4" name="universityNumber" value="4">

<label for="4">4</label><br>

<input type="radio" id="5" name="universityNumber" value="5">

<label for="5">5</label><br>

<br>

<font size="+1">Enter SOP <input id="sop" type="number" name="sop" min="1"

max="5" required="required"></font>

<br><br>

<font size="+1">Enter LOR <input id="lor" type="number" name="lor" min="1"

max="5" required="required"></font>

<br><br>

<font size="+1">Enter CGPA <input id="cgpa" type="number" name="cgpa"

required="required"></font>

<br><br>

<font size="+1">Research</font><br>

<input id="research" type="radio" id="researchDone" name="research"

value="1">

<label for="research">Research</label><br>

<input id="noresearch" type="radio" name="research" value="0">

<label for="noresearch">No Research</label><br>

<br>

<button type="submit" value="Predict"><font

size="+1">Predict</font></button>

</form>

</body>

</html>

**login.html**

<!DOCTYPE html>

<html>

<head>

<style>

body{

background-image: url("admissions.jpg");

background-repeat: no-repeat;

background-position: top 200px left 100px;

width: "300";

height: "300";

object-fit: cover;

}

</style>

<style>

div.static{

position: absolute;

top: 250px;

left:900px;

}

</style>

<h2 style="color:black"><font size="+5">

<b>UNIVERSITY ADMISSION PREDICTION

SYSTEM</b></font></h2>

<title></title>

<link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"

integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"

crossorigin="anonymous">

<!-- Optional theme -->

<link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css"

integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp"

crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript -->

<script

src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"

integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"

crossorigin="anonymous"></script>

</head>

<body>

<div class="static">

<form action="{{url\_for('homepage')}}" method="post">

<h2>Sign in</h2><br>

<font size="+1"><label for="inputEmail">Email address</label></font>

<input type="email" id="inputEmail" name="name" placeholder="Email

address" required autofocus>

<br><br><font size="+1"><label for="inputPassword">Password

</label></font>

<input type="password" id="inputPassword" name="pass"

placeholder="Password" required>

<div class="checkbox">

<label>

<input type="checkbox" value="remember-me"> Remember me

</label>

</div>

<button style="background-color:black; color:white; width:100%"

type="submit">Sign in</button>

</form>

</div>

</div>

<footer></footer>

</body>

</html>

**chance.html**

<!DOCTYPE html>

<html>

<head>

<style>

div.static{

position: absolute;

top: 100px;

left:100px;

}

</style>

</head>

<body>

<div class="static">

<h2>Predicting Chance of Admission</h2>

<p>Prediction: <b ><u>You have a chance</u> </b><b

style='font-size:40px;'>&#128077;</b> </p>

</div>

<img align="right"

src="student.jpg" alt="" width="550" height="380";>

</body>

</html>

**nochance.html**

<!DOCTYPE html>

<html>

<head>

<style>

div.static{

position: absolute;

top: 100px;

left:100px;

}

img{

position: absolute;

top: 150px;

right:200px;

}

</style>

</head>

<body>

<div class="static">

<h2>Predicting Chance of Admission</h2>

<p>Prediction: <b ><u>You don't have a chance</u> </b><b

style='font-size:40px;'>&#128078;</b> </p>

</div>

<img align="right"

src="upset.jpg" alt="" width="550" height="380" >

</body>

</html>

**app.py**

from flask import Flask, request, jsonify, render\_template, redirect, url\_for

import requests

import json

import pickle

model = pickle.load(open('university.pkl','rb'))

import pyrebase

API\_KEY = "w7wZ3NDUKJjLg9ulwEFwDCKCnOurNNLrzp3gZ-SNrbGO"

token\_response = requests.post('https://iam.cloud.ibm.com/identity/token',

data={"apikey":API\_KEY, "grant\_type": 'urn:ibm:params:oauth:grant-type:apikey'})

mltoken = token\_response.json()["access\_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

config = {

"apiKey": "AIzaSyBLcGYGA82pCAHW4xKjgDYv\_bsnEJEgo1E",

"authDomain": "universityadmitpredictor.firebaseapp.com",

"projectId": "universityadmitpredictor",

"storageBucket": "universityadmitpredictor.appspot.com",

"messagingSenderId": "938493164189",

"databaseURL":

"https://console.firebase.google.com/u/0/project/universityadmitpredictor/database/

universityadmitpredictor-default-rtdb/data/~2F",

}

firebase = pyrebase.initialize\_app(config)

auth = firebase.auth()

model = pickle.load(open('university.pkl','rb'))

app = Flask(\_\_name\_\_)

@app.route('/',methods=['GET','POST'])

def homepage():

if request.method == 'POST':

unsuccessful = 'Please check your credentials'

email = request.form['name']

password = request.form['pass']

try:

auth.sign\_in\_with\_email\_and\_password(email, password)

return render\_template('index.html')

except:

auth.create\_user\_with\_email\_and\_password(email,password)

auth.sign\_in\_with\_email\_and\_password(email, password)

return render\_template('index.html')

return render\_template('login.html')

@app.route('/predict', methods=['GET', 'POST'])

def predict():

if request.method == 'POST':

gre = request.form['gre']

toefl = request.form['toefl']

universityNumber = request.form['universityNumber']

sop = request.form['sop']

lor = request.form['lor']

cgpa = request.form['cgpa']

research = request.form['research']

y\_pred = [[gre, toefl, universityNumber, sop, lor, cgpa, research]]

payload\_scoring = {"input\_data": [

{"field": [["GRE Score", "TOEFL Score", "University Rating", "SOP",

"LOR ", "CGPA", "Research"]],

"values": y\_pred}]}

response\_scoring =

requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/67f91885-c382-4d

94-9b23-60bbc3f65a47/predictions?version=2022-11-18',json=payload\_scoring,headers={

'Authorization': 'Bearer ' + mltoken})

print("Scoring response")

print(response\_scoring.json())

predictions = response\_scoring.json()

output = predictions['predictions'][0]['values'][0][0]

print(output)

if output == 'Yes':

return render\_template('chance.html')

if output == 'No':

return render\_template('Nochance.html')

return render\_template('index.html')

@app.route('/index.html',methods=['GET', 'POST'])

def index():

return render\_template('index.html')

@app.route('/about.html')

def about():

return render\_template('about.html')

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**model.py**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import pickle

data=pd.read\_csv('C:/Users/Lenovo/OneDrive/Documents/Nalaiya Thiran/University

Admit Eligibility Predictor/dataset/Admission\_predict.csv')

data.drop(["Serial No."],axis=1,inplace=True)

df = pd.DataFrame(data)

df.columns = df.columns.str.replace(' ', '\_')

df['result']=pd.cut(df.Chance\_of\_Admit\_,bins=[0,0.80,1],labels=['No','Yes'])

independent = data.iloc[:,0:7].values

dependent = data.iloc[:,8:].values

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(independent, dependent,

random\_state=0, train\_size = .2)

from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()

trained\_model = rf.fit(X\_train, y\_train)

pickle.dump(rf, open('university.pkl','wb'))

**Link**

Git hub Link: <https://github.com/IBM-EPBL/IBM-Project-35825-1660289004>