UNIVERSITY ADMIT ELIGIBILITY PREDICTOR

A PROJECT REPORT

Submitted by

Mohammed Fazith.J
Shreeja K
Bogar Ranganathan
Sowmya R
Dhanush Krishnan.A

of

INFORMATION TECHNOLOGY

MADRAS INSTITUTE OF TECHNOLOGY

CHAPTER 1

INTRODUCTION

1.1 Project Overview:

The quality of one's education determines their destiny, hence it plays a significant part in one's life. Following graduation, they frequently have a lot of questions about going back to school and selecting the finest university. Most students favour institutions of higher learning that are well-known worldwide. Therefore, a greater proportion of Indian students prefer to pursue higher education in the United States. Even though India is home to some well regarded universities, graduate students sometimes struggle to gain admission to these institutions. Additionally, because there aren't many job prospects available, finding a position is also a challenge. Students spend time and money seeking advice because they are unsure of which university is the best. There are various blogs and websites that inform students about their possibilities of admission in addition to consulting firms and consultants, but such resources are not particularly accurate, so you shouldn't rely entirely on them.

1.2 Purpose:

In this paper, a machine learning model is established that considers parameters including undergraduate GPA, TOEFL score, university ranking, proposal statement and recommendation letter strength, and study experience in addition to the GRE and TOEFL scores. It forecasts the possibility of admission after receiving all the information. On obscure test occasions, the developed model has significant factual findings for the (like) assessment of the chance of confirmation and, as a result, provides an unbiased picture of measurement.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Problem:

1. Prediction Probability of Getting an Admission into a University using Machine Learning

   The suggested model employs the random forest and linear regression methods, although the cat boost approach provides the maximum accuracy. Making a short list of schools to apply to is a challenging task for prospective graduate students. Because applications are so flexible, students frequently wonder if their profile fits the requirements of a particular university. Additionally, because it is so expensive to apply to a university, it is crucial that students narrow down their list of potential universities based on their profile.

2. A University Admission Prediction System using Stacked Ensemble Learning

   Students can utilize a university admission prediction system to figure out their odds of admittance at a particular institution. Early versions of these prediction systems had several flaws, such as failing to consider crucial factors like GRE (Graduate Record Exam) results or research experience. Furthermore, older models' stated accuracy is likewise insufficiently low. A stacked ensemble model that forecasts a student's chances of admission to a specific university has been proposed in this study.

3. University Admissions Predictor Using Logistic Regression

   The suggested model considers several student-related aspects, such as their background in research and other fields of employment. It might be confusing for students seeking for university admission to know if they stand a decent chance of being accepted or not. To keep this in mind, we used logistic regression techniques, which have drawn attention in the field of software engineering due to their capacity for making predictions.

2.2 References:

[1] Sivasangari, A., et al. "Prediction Probability of Getting an Admission into a University using Machine Learning." 2021 5th International Conference on Computing Methodologies and Communication (ICCMC). IEEE, 2021

[2] Sridhar, Sashank, Siddartha Mootha, and Santosh Kolagati. "A University Admission Prediction System using Stacked Ensemble Learning." 2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA). IEEE, 2020

[3] Fathiya, Haseeba, and Lipsa Sadath. "University Admissions Predictor Using Logistic Regression." 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE). IEEE, 2021.

2.3 Problem statement definition:

Every year thousands of college graduates apply for the master and PhD programs in various universities from all around the world. Applying to foreign universities is not an easy task, it involves many steps and procedures to follow. Choosing the right universities or colleges is definitely an another hurdle students because of the sheer number of universities of different levels. Many 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. Also the cost for applying to universities is time consuming and expensive . This leads students of poor economic backgrounds to frustration and anxiety as they only lose surplus amount of money just for applying to those universities. This is because overall university application cost is not affordable for students with low economic backgrounds

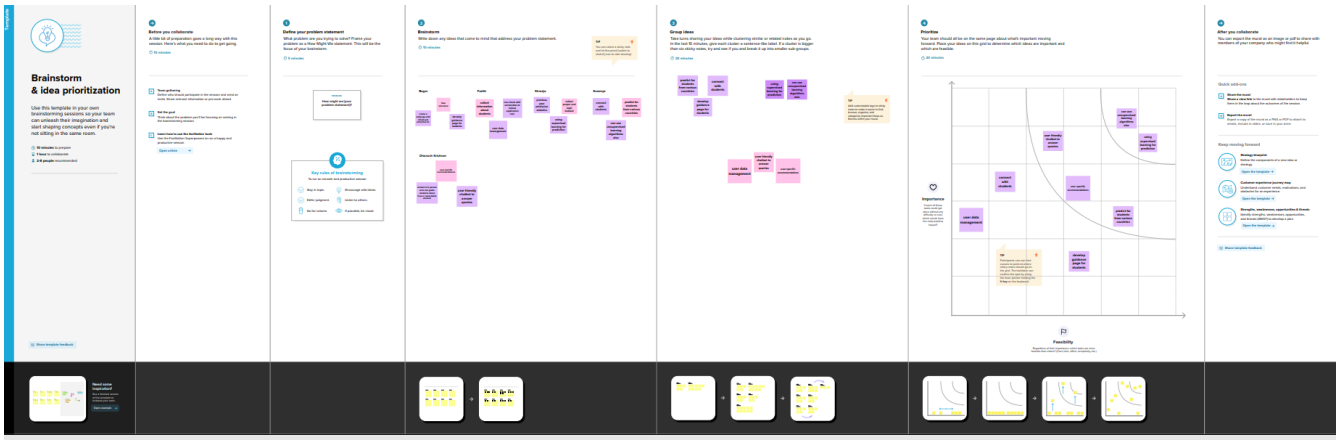
Every year thousands of college graduates apply for the master and PhD programs in various universities from all around the world. Applying to foreign universities is not an easy task, it involves many steps and procedures to follow. Choosing the right universities or colleges is definitely an another hurdle students because of the sheer number of universities of different levels. Many 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. Also the cost for applying to universities is time consuming and expensive . This leads students of poor economic backgrounds to frustration and anxiety as they only lose surplus amount of money just for applying to those universities. This is because overall university application cost is not affordable for students with low economic backgrounds

CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:

3.2 Ideation and Brainstorming:



3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Admission to universities and colleges involves a complicated decision-making process that goes beyond just comparing test results and admission standards. It might be challenging for aspiring graduate students to decide which universities to apply to. Students frequently question whether they are qualified enough for a particular university. This issue has been dealt with in this research by designing a recommender system based on different classification techniques.
2.	Idea / Solution description	Students will log in and fill out their information on the website. Depending on their grades, universities will suggest courses to them. The customer has a variety of universities and courses to pick from. The predictor will compare the data and present a list of programmes and institutions to which the candidate is admittedly qualified.
3.	Novelty / Uniqueness	Instead of the students having to repeatedly choose between several universities and courses, the system suggests options for them. This system links students with the top universities suitable for the individuals based on the university rankings. There will be no need for the student to search for any other websites because the information about the suggested universities will be presented. The changes in qualifications criteria for universities will be updated periodically to be up to date and to provide accurate prediction.
4.	Social Impact / Customer Satisfaction	It will help students to be aware of various universities which are dispersed around the world. Additionally, it cuts down on the time needed to process student applications, admit students, and check their grades. It serves as a central data handling system. Reduce the price of the admissions procedure. By raising the

		process quality, it also increases operational effectiveness.
5.	Business Model (Revenue Model)	By forming partnerships with institutions, the site suggests universities to students. Additionally, the system might be helpful for any student who wants to participate in an undergraduate programme.
6.	Scalability of the Solution	For students, the initiative will forecast their chances of admission in universities. Students can determine if they are qualified for their preferred course at their preferred university.

3.4 Problem Solution Fit:

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Users Data Collection	Collecting the TOEFL, GRE scores from the user.
FR-2	User Registration	Registration through Form Registration through Gmail
FR-3	Predicting the Data	Analysing the given data with the previous year cut-off of the universities and then system provides the list of universities based on the student cut-off.
FR-4	Users Preference	Users can select the universities based on their convenient and preference from the predicted list.
FR5	Output	The Universities are listed based on the Student marks where the universities will be listed in the rankwise , So the predicted output gives them a fair idea about their admission chances in a particular university.

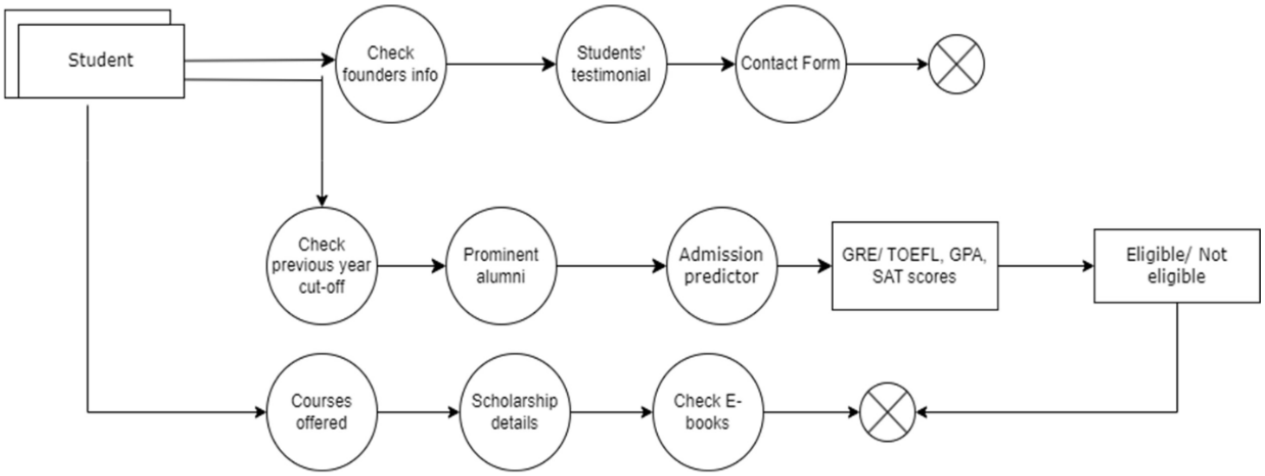
4.2 Non – Functional Requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The predictor platform should provide the capacity to perform the right options for the users based on their profiles.
NFR-2	**Security**	The student profile and data should be maintained in a secured manner.
NFR-3	**Reliability**	• The user can find universities based on their preferred locations and results. • The predictor system should be consistent and the system will give accurate and reliable results.
NFR-4	**Performance**	The system can supply any number of users at a time and provides the list of universities, the predictor platform gives the good performance criteria.
NFR-5	**Availability**	The system predictor will available to users to accessed anytime and anywhere whenever they required.
NFR-6	**Scalability**	The system must be scalable to support many users at a time.

CHAPTER 5

PROJECT DESIGN

5.1 Data Flow Diagram:



5.2 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Landing page	USN-1	As a user, I can view the details about the university	I can access the university landing page	Medium	Sprint-1
		USN-2	As a user, I can view the latest news about the university	I can access the latest news	Medium	Sprint-1
		USN-3	As a user, I can fill the contact form for queries	I can fill and submit the contact form	Low	Sprint-2
		USN-4	As a user, I can see the social media profiles of the university	I can reach out to them via social media	Medium	Sprint-1
		USN-5	As a user, I can see testimonials of students who graduated from the university	I can access the testimonials	Medium	Sprint-1
	Admissions	USN-6	As a user, I can see the previous year cut-off marks	I can download the previous year cut-off details	High	Sprint-2
		USN-7	As a user, I can read about proud alumni of the university	I can access the details of alumni of the university	Medium	Sprint-2
		USN-8	As a user, I can predict my eligibility for admission at the university	I can get result as either eligible/not eligible	High	Sprint-2
	Courses offered	USN-9	As a user, I can see the courses offered by the university for PG students	I can access the course details	Medium	Sprint-3
	Events	USN-10	As a user, I can check various technical events about to happen in the university	I can register for the events	Low	Sprint-3
	E-books	USN-11	As a user, I can download and read e-books relating to visa formalities	I can download the e-books	High	Sprint-3

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 Sprint planning and estimation

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Landing Page	USN-1	As a user, I can enter the scores which are required by the university in order to predict the result.	2	High	4
Sprint-1	Choose University	USN-2	As a user, I will be able to view the list of universities that the students are eligible to apply.	5	High	4
Sprint-1	Choose Course	USN-3	As a user, I will be able to view the list of coursesthat the students are eligible to apply.	5	Medium	2
Sprint-1	Admission Process	USN-4	As a user, I will be able to view the details Of Admission process	2	Low	5
Sprint-1	Data collection	USN-5	As a admin,the historical data is collected and cleaned in order to achieve the optimal prediction. The correlation between the data is analyzed through visualization.	1	Medium	3
Sprint-1	Pre-requisite for Model Building	USN-6	As a admin,The Data is transformed into training set suitable to build the model.	2	Medium	1
Sprint-2	Model Building	USN-7	As a admin,amodel is built using various machine learning techniques,.	2	High	1
Sprint-2	Model Testing	USN-8	As a admin,the Built model has been checked for accuracy and other performance metrics.	1	High	2
Sprint-3	Integration	USN-9	As a admin,the frontend and the developed Machine Learning model is integrated using flask API	1	High	3
Sprint-4	Deployment in Cloud	USN-10	As a admin,the developed application is deployed in the cloud so that it can be accessed by anyone.	2	High	5

CHAPTER 7

CODING & SOLUTION

7.1 Model Building

7.1.1 Data Collection & Exploratory Data Analysis

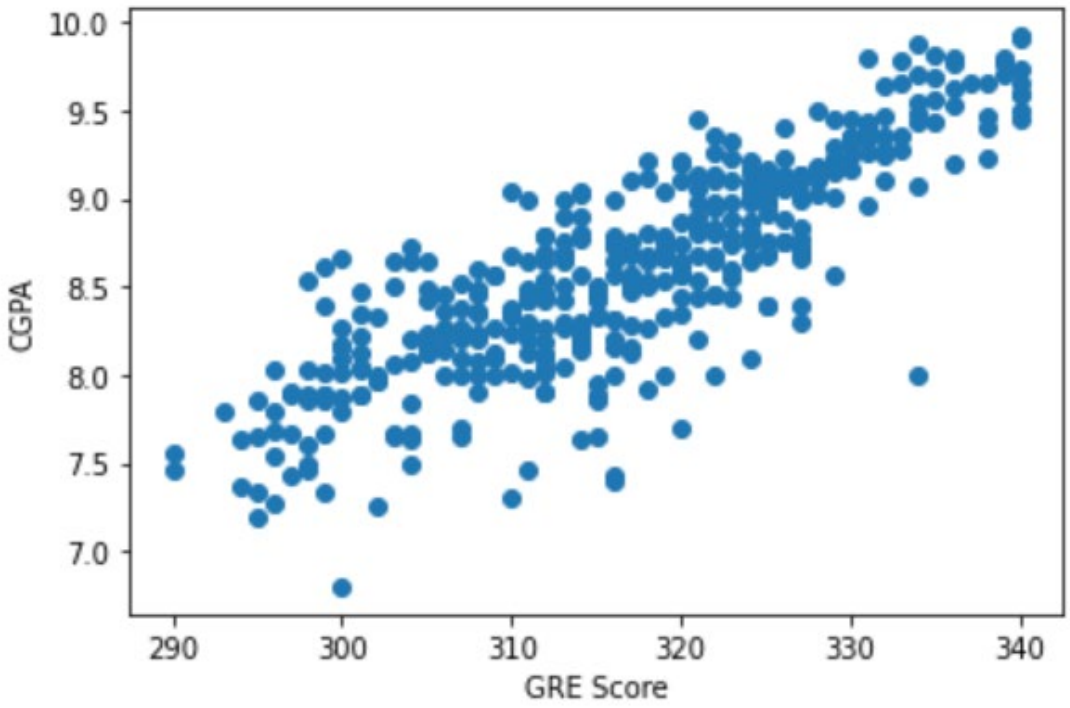
:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65
...	...	...	...	...	...	...	...	...	...
395	396	324	110	3	3.5	3.5	9.04	1	0.82
396	397	325	107	3	3.0	3.5	9.11	1	0.84
397	398	330	116	4	5.0	4.5	9.45	1	0.91
398	399	312	103	3	3.5	4.0	8.78	0	0.67
399	400	333	117	4	5.0	4.0	9.66	1	0.95

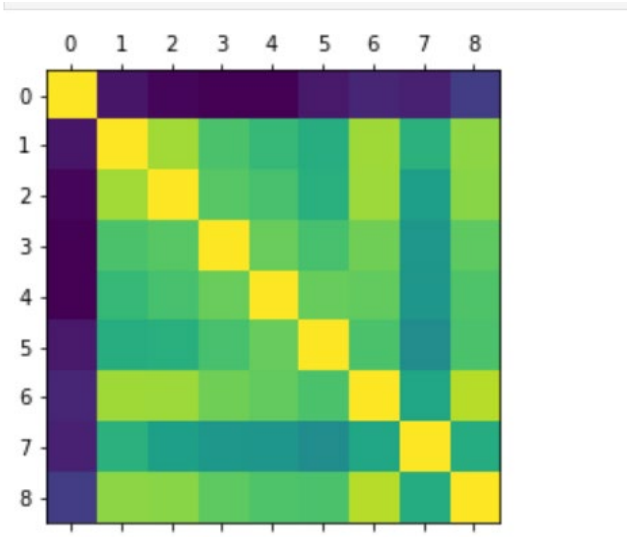
400 rows × 9 columns

7.1.2 Data Visualization

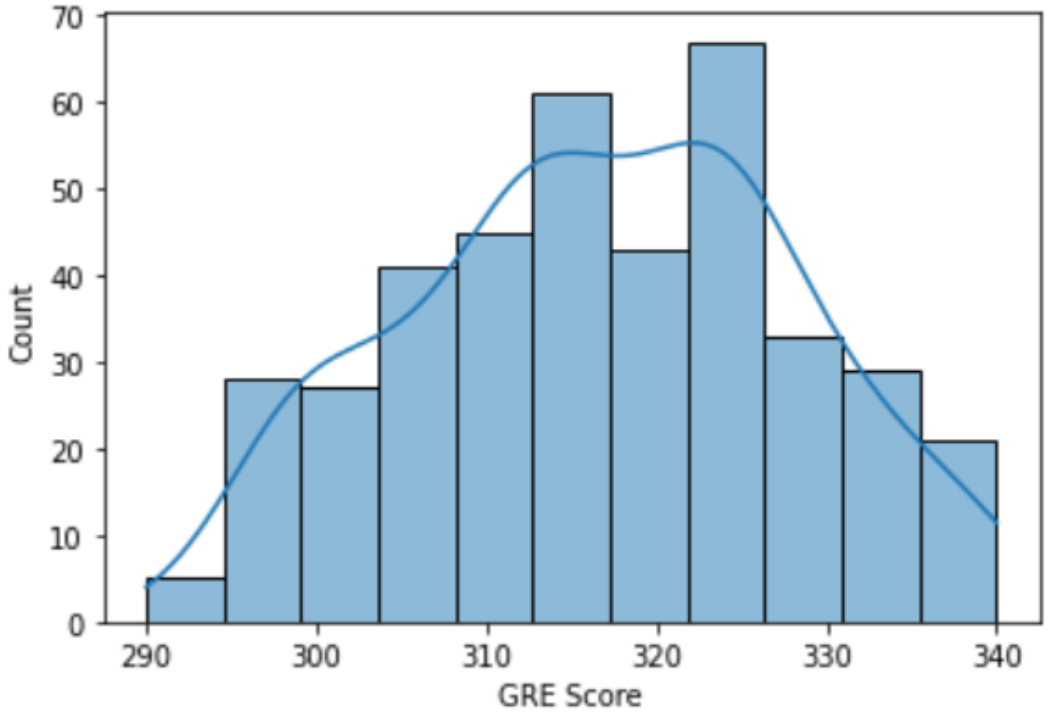
Data visualization helps to understand the data and also explain the data to others. Histogram, box plot, correlation matrix plot, scatter matrix plot, pair plot has been plotted.



Correlation Matrix:



Histogram:



7.1.3 Data Preprocessing & Splitting the dataset

In the first step data preprocessing is done. Preprocessing is the method by which we perform data cleaning i.e., raw dataset is converted into cleaned dataset. There are no missing values in the dataset. The dataset is divided into 80:20 ratio where 80% is for training data and 20% for testing data.

Splitting The Data Into Train And Test

In [40]:
```
from sklearn.model_selection import train_test_split
x=df[['GRE Score','TOEFL Score','University Rating','CGPA']]
y=df['Chance of Admit ']
```

In [41]:
```
y=(y>0.5).astype(int)
```

In [42]:
```
y
```

Out[42]:
```
0      1
1      1
2      1
3      1
4      1
      ..
395    1
396    1
397    1
398    1
399    1
Name: Chance of Admit , Length: 400, dtype: int32
```

In [43]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

7.1.4 Model Building and Hyper parameter tuning

Support Vector Machine

In [7]:
```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from decimal import Decimal
svm_clf = SVC(probability = True, kernel='linear')
svm_clf.fit(x_train,y_train)
```

Out[7]:
```
SVC(kernel='linear', probability=True)
```

Testing the model

In [8]:
```
y_pred = svm_clf.predict(x_test)

acc_svm = round(Decimal(accuracy_score(y_test, y_pred) * 100), 2)

print(f"Accuracy (SVM) : {acc_svm}%")
```

```
Accuracy (SVM) : 98.75%
```

Decision Tree

:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
drugTree = DecisionTreeClassifier(criterion="entropy", max_depth=4)

drugTree.fit(x_train, y_train)
predicted = drugTree.predict(x_test)

print(predicted)

print("DecisionTrees's Accuracy: ", metrics.accuracy_score(y_test, predicted))
```

```
[1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1]
DecisionTrees's Accuracy:  0.9625
```

Random Forest

```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators = 1000, random_state = 42)
# Train the model on training data
rf.fit(x_train,y_train);
```

```
# Use the forest's predict method on the test data
predictions = rf.predict(x_train)
# Calculate the absolute errors
errors = abs(predictions - y_train)
# Print out the mean absolute error (mae)
print('Mean Absolute Error:', round(np.mean(errors), 2), 'degrees.')
```

```
Mean Absolute Error: 0.05 degrees.
```

```
# Calculate mean absolute percentage error (MAPE)
mape = 100 * (errors / y_train)
# Calculate and display accuracy
accuracy = 100 - np.mean(mape)
print('Accuracy:', round(accuracy, 2), '%.')
```

```
Accuracy: 94 %
```

7.2 Flask Integration

Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet"  href="{{ url_for('static', filename='style.css') }}" />
  <!-- <link rel="stylesheet" href="style.css">-->
   <title>Model Deployment</title>


</head>
<body>
```

```
   <image src="{{ url_for('static', filename='images/header.jpeg') }}" alt="Girl in a jacket"
style="width: 1880px;height: 150px;">


      <center>
      <h1><b>UNIVERSITY ADMIT PREDICTION SYSTEM </b></h1>
      <h2>Enter your details and get probability of your admission</h2>
      <form action="{{ url_for('predict')}}" method="post">
      <table>
         <tr><th>Enter GRE score:</th>
            <td><input type="number" name="precp" placeholder="Enter GRE score" ></td></tr>
         <tr><th>Enter TOFEL score</th>
            <td><input type="number" name="maxT" placeholder="Enter TOFEL score" ></td></tr>
         <tr><th>University number</th>
            <td><select name="no">
               <option value="1">1-Stanford</option>
               <option value="2">2-Windsor</option>
               <option value="3">3-Michigan</option>
               <option value="4">4-Harvard</option>
               <option value="5">5-University of Toronto</option>
            </select></td>
            </tr></br>



                     <!--<tr><th>Enter SOP </th>
                        <td><input type="number" name="sop" placeholder="sop" ></td></tr> -->
                        <tr><th>Enter cgpa </th>
                           <td><input type="number" step="any" name="cgpa" placeholder="cgpa"
></td></tr>
                           <!-- <tr><th>Enter LOR </th>
                              <td><input type="number" name="lor" placeholder="lor" ></td></tr>

                              <tr><th>Reserach</th>
  <td><input type="radio" name="RESEARCH" value="RESEARCH">RESEARCH</td></tr>
  <tr><th></th>
    <td><input type="radio" name="NO RESEARCH" value="NO RESEARCH">NO
RESEARCH</td></tr>-->

</table>
      <br>
      <!--<button type="button">PREDICT</button>-->
      <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
      </form>
      <br>
      <br><br>
      </center>
```

```
</body>
</html>
```

Selected.html:

```
<!DOCTYPE html>
<html>
   <head>
      <style>
        body{ background-color:rgb(134, 222, 144);}
      </style>
      <title>
         SELECTED
      </title>
   </head>
   <body><center>

      <h1>
        <u>  Predicting chance of Admission</u>
      </h1>

      <h3>
         Prediction:<u>
            You Have a Chance



         </u>
      </h3>

      <img src="{{ url_for('static', filename='images/thumbs-up.jpeg') }}" alt="thumbsup"style="
height:50px; width:50px;">
      <br>
      <img src="{{ url_for('static', filename='images/selected.jpeg') }}" alt="rejected"style="
height:280px; width:350px;">
   </center>
   </body>
</html>
```

Rejected.html:

```
<!DOCTYPE html>
<html>
```

```
  <head>
    <style>
      body{ background-color:rgb(232, 139, 122);}
    </style>
    <title>
      REJECTED
    </title>
  </head>
  <body><center>

    <h1>
     <u>  Predicting chance of Admission</u>
    </h1>

    <h3>
      Prediction:<u>
        You Dont Have a Chance



      </u>
    </h3>

    <img src="{{ url_for('static', filename='images/thumbs-down.jpeg') }}" alt="thumbsdown"style="
height:50px; width:50px;">
    <br>
    <img src="{{ url_for('static', filename='images/rejected.jpeg') }}" alt="rejected"style="
height:280px; width:350px;">
  </center>
  </body>
</html>
```

Design.css

```
html, body {
   padding: 0;
   margin: 0;
}
button:hover {
  background-color: #4CAF50; /* Green */
  color: white;
}
button {
  transition-duration: 0.4s;
  background-color:rgb(68, 83, 202) ;
  border: none;
  color: rgb(240, 235, 238);
  padding: 5px 15px;
```

```
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}
h2{
  font-family: "Brush Script MT";
  font-size:20;
  padding: 5px 10px;
}

body{
  font-family: "Poppins", sans-serif;
  font-weight: 300;
  font-size: 20px;
  line-height: 1.7;
  color: #171426;
  background-color: rgb(149, 156, 205);
  overflow-x: hidden;

}


table{
  display: table;
  border-collapse: separate;
  box-sizing: border-box;
  text-indent: initial;
  border-spacing: 2px;
  padding: 0.5vmin;
  border-radius: 1em;
  border: 2px solid #e5dab6;
}

center{
  display: block;
  text-align: -webkit-center;
}



tr, th, td{
  padding: 0.5vmax;
  text-align: left;
}
```

```
input {
  background-color: #c4c3ca;
  color: #1f2029;
}
```

App.py

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle


# Create flask app
flask_app = Flask(__name__)
model = pickle.load(open('svm.pkl','rb'))

@flask_app.route("/")
def Home():
    return render_template("index.html")

@flask_app.route("/predict", methods = ["POST"])
def predict():
    float_features = [float(x) for x in request.form.values()]
    features = [np.array(float_features)]
    prediction = model.predict(features)
    if prediction==1:
        return render_template("goto.html")
    else:
        return render_template("rejected.html")

if __name__ == "__main__":
    flask_app.run(debug=True)
```

CHAPTER 8

RESULTS

8.1Performance Metrics

Accuracy:

- **SUPPORT VECTOR MACHINE:**

  Validation Accuracy :98.75%

- **DECISION TREE:**

  Validation Accuracy:96.25%

- **RANDOM FOREST:**

  Validation Accuracy:94%

CHAPTER 9

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- Predict eligibility for different universities
- High Accuracy
- User friendly

DISADVANTAGES:

- Not a generalized model
- Huge number of rules

CHAPTER10

CONCLUSION

In this study, a proposed methodology, a few algorithms, and their implementation are reviewed, as well as the stages required in model trained and discussed. The accuracy of 98 %obtained with the Support vector machine is the highest quantitative result to predict eligibility for admission into various universities

CHAPTER 11

FUTURE SCOPE

We plan to develop system add-ons in the future, and if we can obtain a structured dataset , we will be able to detect it much more quickly than with any other method. In the future work a web extension can be made so that the working will be much simplified for the endusers / customers.

Chapter 12

APPENDIX

12.1 Source Code

Flask Integration with scoring end point Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet"  href="{{ url_for('static', filename='style.css') }}" />
   <!-- <link rel="stylesheet" href="style.css">-->
    <title>Model Deployment</title>


</head>
<body>
    <image src="{{ url_for('static', filename='images/header.jpeg') }}" alt="Girl in a jacket"
style="width: 1880px;height: 150px;">


        <center>
        <h1><b>UNIVERSITY ADMIT PREDICTION SYSTEM </b></h1>
        <h2>Enter your details and get probability of your admission</h2>
        <form action="{{ url_for('predict')}}" method="post">
        <table>
            <tr><th>Enter GRE score:</th>
                <td><input type="number" name="precp" placeholder="Enter GRE score" ></td></tr>
            <tr><th>Enter TOFEL score</th>
                <td><input type="number" name="maxT" placeholder="Enter TOFEL score" ></td></tr>
            <tr><th>University number</th>
                <td><select name="no">
                    <option value="1">1-Stanford</option>
                    <option value="2">2-Windsor</option>
                    <option value="3">3-Michigan</option>
                    <option value="4">4-Harvard</option>
                    <option value="5">5-University of Toronto</option>
                </select></td>
                </tr></br>
```

```
                    <!--<tr><th>Enter SOP </th>
                       <td><input type="number" name="sop" placeholder="sop" ></td></tr> -->
                       <tr><th>Enter cgpa </th>
                          <td><input type="number" step="any" name="cgpa" placeholder="cgpa"
></td></tr>
                         <!-- <tr><th>Enter LOR </th>
                             <td><input type="number" name="lor" placeholder="lor" ></td></tr>

                          <tr><th>Reserach</th>
  <td><input type="radio" name="RESEARCH" value="RESEARCH">RESEARCH</td></tr>
  <tr><th></th>
    <td><input type="radio" name="NO RESEARCH" value="NO RESEARCH">NO
RESEARCH</td></tr>-->

</table>
       <br>
       <!--<button type="button">PREDICT</button>-->
       <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
       </form>
       <br>
       <br><br>
       </center>



</body>
</html>
```

Selected.html:

```
<!DOCTYPE html>
<html>
   <head>
      <style>
        body{ background-color:rgb(134, 222, 144);}
      </style>
      <title>
         SELECTED
      </title>
   </head>
   <body><center>

      <h1>
        <u>  Predicting chance of Admission</u>
      </h1>
```

```
      <h3>
         Prediction:<u>
            You Have a Chance



         </u>
      </h3>

      <img src="{{ url_for('static', filename='images/thumbs-up.jpeg') }}" alt="thumbsup"style="
height:50px; width:50px;">
      <br>
      <img src="{{ url_for('static', filename='images/selected.jpeg') }}" alt="rejected"style="
height:280px; width:350px;">
   </center>
   </body>
</html>
```

Rejected.html:

```
<!DOCTYPE html>
<html>
   <head>
      <style>
        body{ background-color:rgb(232, 139, 122);}
      </style>
      <title>
         REJECTED
      </title>
   </head>
   <body><center>

      <h1>
        <u>  Predicting chance of Admission</u>
      </h1>

      <h3>
         Prediction:<u>
            You Dont Have a Chance



         </u>
      </h3>

      <img src="{{ url_for('static', filename='images/thumbs-down.jpeg') }}" alt="thumbsdown"style="
```

```
height:50px; width:50px;">
        <br>
        <img src="{{ url_for('static', filename='images/rejected.jpeg') }}" alt="rejected"style="
height:280px; width:350px;">
    </center>
    </body>
</html>
```

Design.css

```
html, body {
    padding: 0;
    margin: 0;
}
button:hover {
  background-color: #4CAF50; /* Green */
  color: white;
}
button {
  transition-duration: 0.4s;
  background-color:rgb(68, 83, 202) ;
  border: none;
  color: rgb(240, 235, 238);
  padding: 5px 15px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  margin: 4px 2px;
  cursor: pointer;
}
h2{
  font-family: "Brush Script MT";
  font-size:20;
  padding: 5px 10px;
}

body{
  font-family: "Poppins", sans-serif;
  font-weight: 300;
  font-size: 20px;
  line-height: 1.7;
  color: #171426;
  background-color: rgb(149, 156, 205);
  overflow-x: hidden;

}
```

```
table{
  display: table;
  border-collapse: separate;
  box-sizing: border-box;
  text-indent: initial;
  border-spacing: 2px;
  padding: 0.5vmin;
  border-radius: 1em;
  border: 2px solid #e5dab6;
}

center{
  display: block;
  text-align: -webkit-center;
}



tr, th, td{
  padding: 0.5vmax;
  text-align: left;
}

input {
  background-color: #c4c3ca;
  color: #1f2029;
}
```

App.py

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle


# Create flask app
flask_app = Flask(__name__)
model = pickle.load(open('svm.pkl','rb'))

@flask_app.route("/")
def Home():
    return render_template("index.html")

@flask_app.route("/predict", methods = ["POST"])
def predict():
```

```
    float_features = [float(x) for x in request.form.values()]
    features = [np.array(float_features)]
    prediction = model.predict(features)
    if prediction==1:
        return render_template("goto.html")
    else:
        return render_template("rejected.html")

if __name__ == "__main__":
    flask_app.run(debug=True)
```

GitHub and project link

GitHub link - https://github.com/IBM-EPBL/IBM-Project-1964-1658421490