IBM PROJECT 2022

UNIVERSITY ADMIT ELIGIBILITY PREDICTOR

PROJECT REPORT DOCUMENTATION

TEAM ID: PNT2022TMID35368

1. INTRODUCTION

1. Project Overview

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.

2. Purpose

- Students often feel difficulty in shortlisting the universities to apply for which they tend to wonder if their profile matches the requirements of a certain university.
- Moreover, the cost of applying to a university is extremely high making it critical that students shortlist universities based on their profile.
- A university admission prediction system is quite useful for students to determine their chances of acceptance to a specific university.
- This system reduces dependence on educational consultancies, who charge loads of money to analyze a candidate's profile and determine the universities he/she should apply to.

2. LITERATURE SURVEY

1. Existing problem

There exists no tools or technologies for study-abroad aspirants to get their profiles evaluated for free. Hence, this Predictor aims to consider various factors involved in an application and use ML models to predict the chances of the individual.

2. **References**

1. Graduate Admission Prediction Using Machine Learning

   Aljasmi, et. all. talk about the student admission problem which is very important in educational institutions. This paper addresses machine learning models to predict the chance of a student to be admitted to a master's program. They propose a system that will assist students to know in advance if they have a chance to get accepted. The machine learning models used are multiple linear regression, k-nearest neighbor, random forest, and Multilayer Perceptron. Experiments show that the Multilayer Perceptron model surpasses other models.

2. HRSPCA: Hybrid recommender system for predicting college admission

   Ragab et.all., present a new college admission system using hybrid recommender based on data mining techniques and knowledge discovery rules, for tackling college admissions prediction problems. This is due to the huge numbers of students required to attend university colleges every year. The proposed HRSPCA system consists of two cascaded hybrid recommenders working together with the help of college predictor, for achieving high performance.

3. University Admissions Predictor Using Logistic Regression

   Fathiya and Sadath perform a novel study on a predictor for university admissions that allows students to assess their chances of being admitted to an institution. Real student data is gathered in order to construct this. The information is kept in the form of a training set that may be used by the logistic regression classifier that was designed to predict admissions.

4. A Machine Learning Approach for Graduate Admission Prediction

AlGhamdi et.all., evaluate three learning strategies of regression to predict the university rate given the students' profile; namely, linear regression, decision tree, and logistic regression model. This paper evaluates these models to select the best model in terms of the highest accuracy rate and the least error. It was determined that the Logistic Regression model shows the most accurate prediction and hence this model was employed to predict the future applicant's university chance of admission.

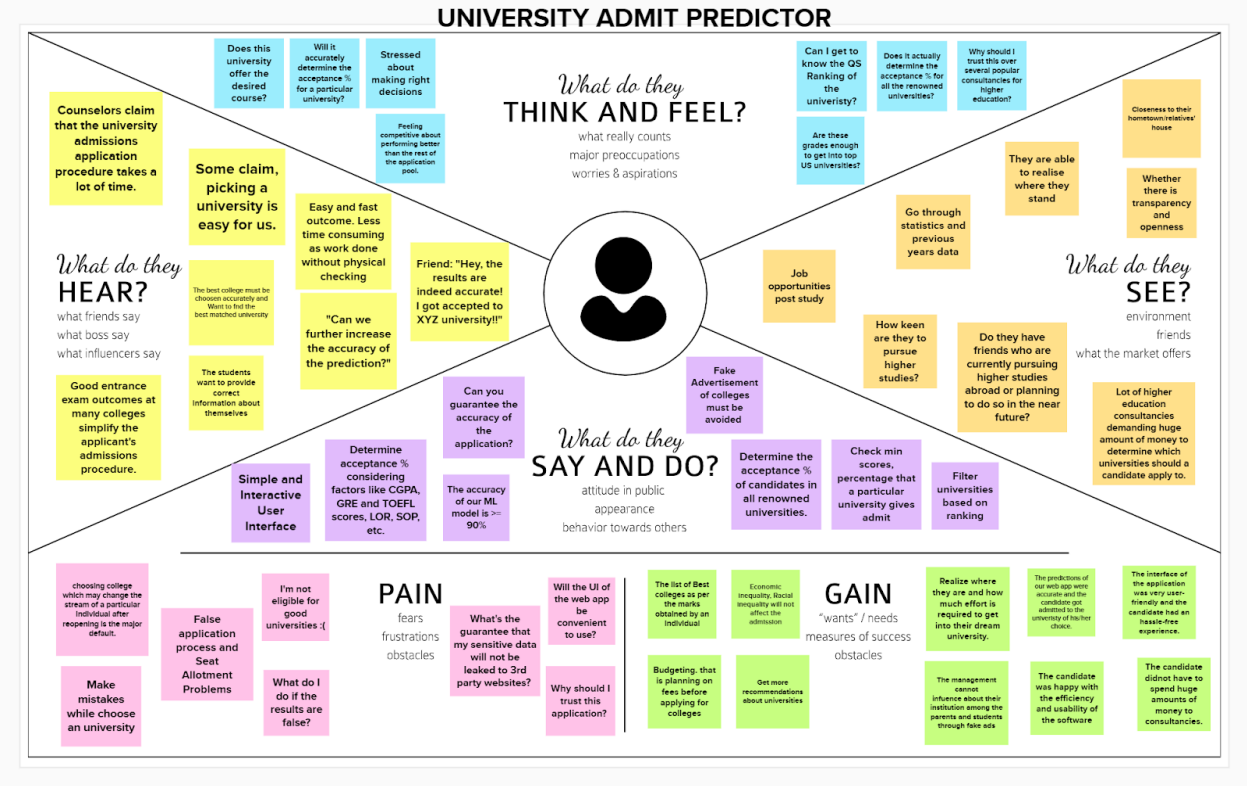
3. **Problem Statement Definition**

Concerns about getting into college are common among students. This project's goal is to assist students in narrowing down institutions based on their profiles. The anticipated results offer them a good indication of their prospects of admission to a particular university. This analysis ought to provide better insight for students who are or will be preparing for exams.

3. IDEATION & PROPOSED SOLUTION

1. **Empathy Map Canvas**

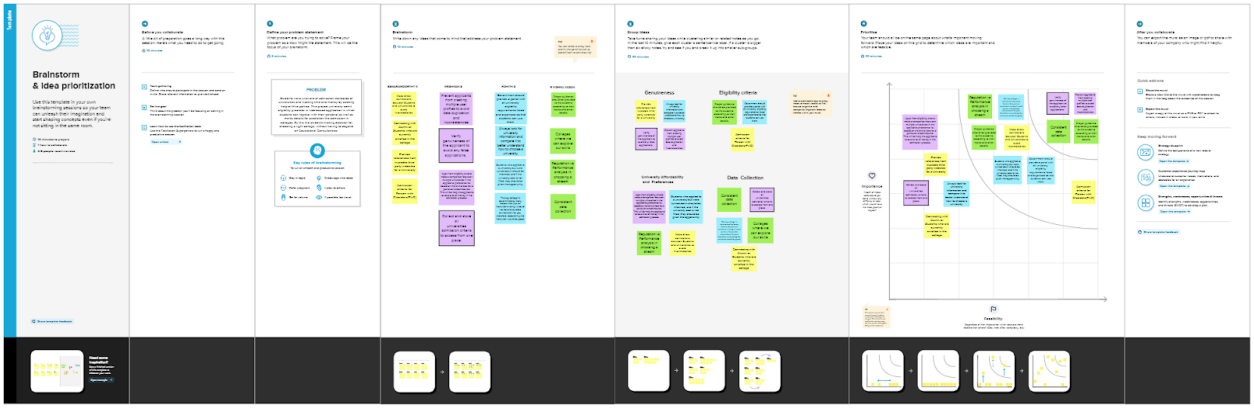
The below figure represents the Empathy Map of the project.



Source: Empathy Map

2. Ideation & Brainstorming

The below figure represents the Ideation & Brainstorming image of the project



Source: Brainstorming and Idea Prioritization

3. **Proposed Solution**

The below table represents the Proposed Solution.

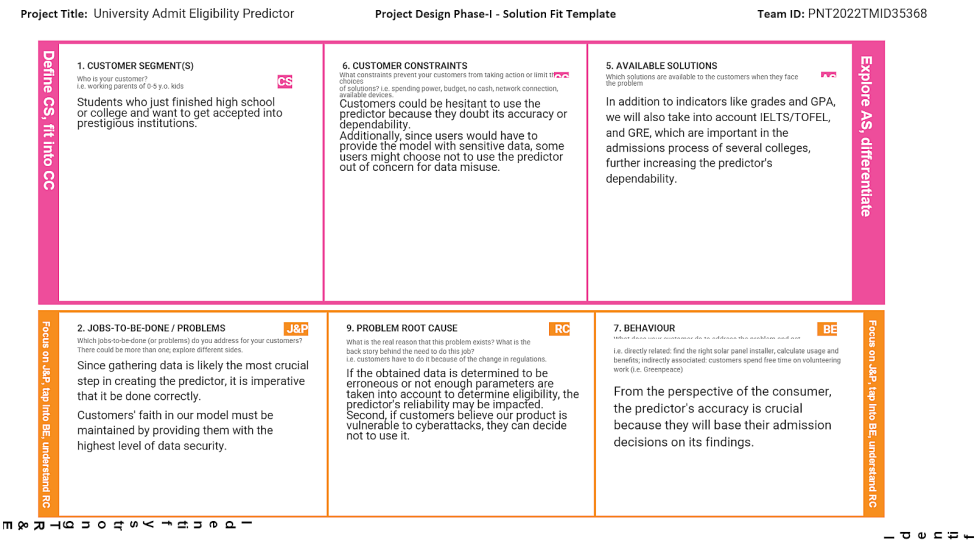
S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Students are often worried about their chances of admission to university. The aim of this project is to help students in shortlisting universities with their profiles. The predicted output gives them a fair idea about their admission chances in a particular university. This analysis should also help students who are currently preparing or will be preparing to get a better idea. It also aims to make a direct connection between the students and the universities and avoid any intermediaries.
2.	Idea / Solution description	This project intends to calculate the probability of acceptance in a particular grad-school after assessing the candidate's profile. The key attributes that will be considered for making the decisions are: i) GRE & TOEFL Scores ii) Undergrad CGPA iii) SOP & LOR iv) Corporate Work Experience / Research Experience v) Extracurriculars For determining the % of acceptance, we will be using various ML models such as Logistic Regression, Multiple Linear Regression, Decision Tree & Random Forest and assess which model gives the highest accuracy with the help of performance metrics like accuracyscore, precision and recall.

3.	Novelty / Uniqueness	● We intend to develop a novel deep learningbased hybrid model that has a better accuracy than the existing traditional ML models. ● The web-app will also provide feedback on the parameters where the candidate is lacking so that he can improve on those areas.
4.	Social Impact / Customer Satisfaction	● Students often feel difficult in shortlisting the universities to apply which they tend to wonder if their profile matches the requirement of a certain university. ● Moreover, the cost of applying to a university is extremely high making it critical that students shortlist universities based on their profile. ● A university admission prediction system is quite useful for students to determine their chances of acceptance to a specific university. ● This system reduces dependence on educational consultancies, who charge loads of money to analyse a candidate’s profile and determine the universities he/she should apply to.
5.	Business Model (Revenue Model)	● Advertisements of different universities could be placed in the web-app to generate revenue through ads. ● In the future, a separate premium plan could be created where the students can directly interact with the professors and alumni of the university through video calls.

6.	Scalability of the Solution	● A future update could have chat space where candidates, faculties, current students of the university and alumni can interact and candidates can get their doubts resolved instantly. ● To deal with huge volumes of data in the future (Both - applicants and university details), cloud-based storages (IBM cloud, AWS, GCP, AZURE) and NoSQL databases (MongoDB, Redis, etc.) could be used instead of the traditional RDBMS storage. ● Alternatively, distributed big-data processing techniques could be explored if the no. of users using the website increase exponentially during the course of time.

4. **Problem Solution fit**

The below images represent the Problem Solution Fit of the project



REQUIREMENT ANALYSIS

1. Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	A user Interface where the candidate enters his details for determining his chances of admit.	A user Interface where the candidate enters his details for determining his chances of admit.
FR-2	Univariate Analysis	Graphical visualization of the different attributes - count plot, dist plot etc.
FR-3	Model Building	Compare and determine which regression model has the best performance.
FR-4	Deployment of ML model to IBM cloud.	Push the notebook and CSV files as an asset in deployment space and deploy the model in IBM Watson.
FR-5	Access the model as a scorable endpoint	Access the deployed model as an API in the Streamlit webapp using the API key generated.
FR-6	Hosting the web app in	Host the web app in Streamlit's cloud platform.

	public cloud.	
FR-7	CI-CD pipeline	Link the repo & branch to Streamlit's cloud hosting platform to setup CI-CD pipeline.

2. **Non-Functional requirements**

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

NFR No.	**Non-Functional Requirement**	**Description**
NFR-1	Usability	A logical interface is required to make the system easy to use and to speed up typical processes. The mistake rate of users providing their information on the checkout page must not exceed 10%.
NFR-2	Security	Authorization access scenarios and definitions, as well as student record handover processes between universities. Utilize certain cryptographic techniques. When the application is validating the user or licence, communication must be limited.
NFR-3	Reliability	Data corruption is avoided by employing backup methods and strategies. At the moment of input, all data stored for user variables will be committed to the database.
NFR-4	Performance	The availability results of the requested college should be supplied to the student in little more than two seconds, and data retrieval should be trustworthy because each student will be granted a maximum of 10

		minutes, accessing the databas
NFR-5	Availability	The system should be available at all times, allowing the user easy access. If the hardware or database fails, a substitute page will be displayed, and the database should be obtained from the data folder.
NFR-6	Scalability	Determines the highest workloads under which the system will still run satisfactorily. Deals with the measurement of the system's reaction time under varied load circumstances.

5. PROJECT DESIGN

1. Data Flow Diagrams

The Data Flow Diagram can be viewed in the link below.

Data Flow Diagrams

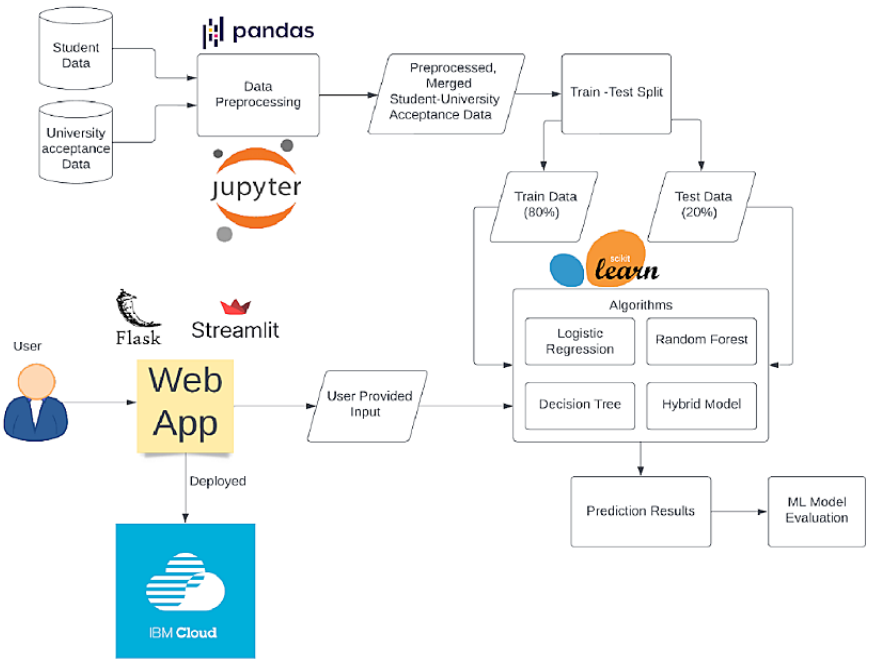
2. Solution & Technical Architecture

The below figure represents the Solution & Technical Architecture of the project

Technical Architecture

Date	18 October 2022
Team ID	PNT2022TMID35368
Project Name	University Admit Eligibility Predictor
Maximum Marks	4 Marks

System Architecture Diagram:



3. User Stories

This section contains the Project Components & Technologies, and Application Characteristics.

Table-1: Components & Technologies

SNO.	Component	Description	Technology
1.	User Interface	The Front-end part of the application for accepting user data.	Flask, Streamlit
2.	Dataset pre-processing	Removing inconsistencies in the dataset.	Pandas, Numpy, Python
3.	Application Logic	The core business logic of the application.	Python
4.	Database	For storing student & university details.	MySQL, IBM DB2, IBM Cloudant, etc.
5.	Data Visualization	Graphical visualization of student data, University's past acceptance trends, Heatmaps depicting the correlation of different attributes that play a crucial role in determining acceptance, etc	Matplotlib, Seaborn, Plotly
6.	File Storage	For storing the SOPs, LORs and other relevant PDF documents uploaded by the user.	IBM Cloud File Storage
7.	ML Model	Models to be used for prediction – Logistic Regression, DTree, Random Forest and a Hybrid Deep Learning based model.	Scikit-Learn
8.	Performance Metrics	Accuracy of the ML model on the trained and tested data.	Root Mean Squared Logarithmic Error (RMSLE), Mean Squared Error (MSE)
9.	Infrastructure	Cloud Server Configuration for hosting the web app.	IBM Cloud Hosting

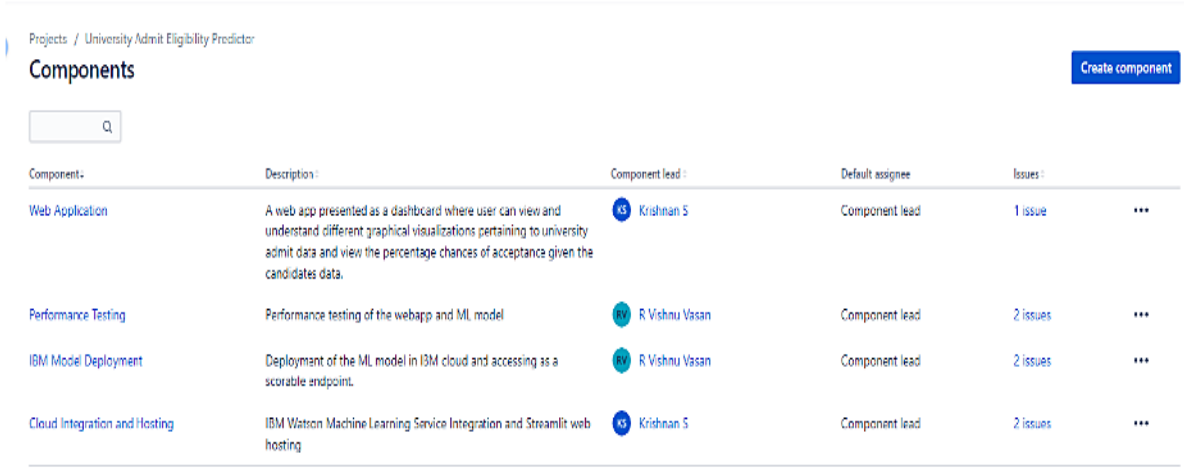


Table-2: Application Characteristics:

SNO.	Characteristics	Description	Technologies Used
1.	Security Implementations	Authenticating the users before making the predictions.	Cloud authentication services with modern, secure encryption schemes like SHA 256
2.	Availability	Since the web app is hosted on cloud, it can be accessed from any device, anywhere. Also, load balancing will be implemented using IBM cloud services to distribute the load across multiple servers.	IBM Cloud Hosting, IBM Load Balancer
3.	Performance	We will be implementing 4 different ML models – Logistic Regression, Decision Tree, Random Forest and a Hybrid model and then determine which model gives the highest accuracy after comparing the model-accuracy, precision and recall values.	Scikit-Learn, Root Mean Squared Logarithmic Error (RMSLE), Mean Squared Error (MSE)

4.	Scalable Architecture	The proposed architecture is scalable even if the no. of users registering the web app increases exponentially as the system has a cloud storage for storing the pdf documents, which can easily handle many requests. Also, the possibility of the website crashing is very minimal even if the number of users increase as IBM Load balancer takes care of distributing the load across the various servers.	IBM Cloud Services

6. PROJECT PLANNING & SCHEDULING

1. Sprint Planning & Estimation

The below figure represents the Sprint Planning & Estimation details.

2. Sprint Delivery Schedule

The below figure represents the Sprint Delivery Schedule details.

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	Story Points	Priority	Team Members
Sprint-1	Exploratory Data Analysis	US1	Perform data cleaning if required and perform univariate, bivariate and multivariate analysis.	1	Low	Rohith S
Sprint-1	Analysis of different regression models	US2	Compare the R2 scores of different fundamental regression models like Decision Trees, Random Forest, Multiple Linear Regression, Logistic Regression, etc and determine which model has the highest R^2 score.	2	Medium	Krishnan S
Sprint-2	Web App Development and model integration using pickle file	US3	Develop the web app using Streamlit predict the probability of acceptance given a test data for a candidate. Persist the model with highest R^2 score as a pickle file and integrate it with the web app.	3	High	Krishnan S
Sprint-3	Deploying the model in IBM cloud.	US4	Register in IBM cloud. Use IBM Watson ML service and IBM Watson Studio to deploy the Multiple Linear Regression Model. Test the deployed model with few examples.	3	High	Vishnu Vasan R, Eswaramoorty K
Sprint-4	Integrate the web app with the deployed model.	US5	Use the deployed model in IBM Watson through the scoring endpoint by making an API call with the IBM cloud API key.	2	Medium	Rohith S, Krishnan S
Sprint-4	Hosting the web app in Streamlit cloud platform.	US6	Connect the respective Github repo and branch to Streamlit cloud platform and set up CI-CD to automatically deploy new changes that's pushed to the repo.	1	Low	Krishnan S

PROJECT TRACKER, VELOCITY & BURNDOWN CHART: (4 MARKS)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story points completed (as on planned date)	Sprint release date (Actual)
Sprint-1	3	6 days	24 Oct 2022	29 Oct 2022	3	30 Oct 2022
Sprint-2	3	6 days	31 Oct 2022	05 Nov 2022	3	06 Nov 2022
Sprint-3	3	6 days	07 Nov 2022	12 Nov 2022	3	9 Nov 2022
Sprint-4	3	6 days	14 Nov 2022	19 Nov 2022	3	11 Nov 2022

3. **Reports from JIRA**

   The below link details about the respective reports from JIRA

   JIRA Spring Dashboard

7. CODING & SOLUTIONING

1. Model Deployment in IBM Watson ML Service

In this module, we run the jupyter notebook in IBM Watson studio before deploying the model in IBM cloud using the Watson Machine Learning service. The deployed model was used in the web app via an API call, utilising the power of the cloud.

Program Code Snippet:

CREATING DEPLOYMENT SPACE

```
In [56]: def guid_from_space_name(client, space_name):
             space = client.spaces.get_details()
             return (next(item for item in space['resources'] if item['entity']['name'] == space_name)['metadata']['id'])

In [57]: space_uid = guid_from_space_name(client, 'Regression-Models')
         print("Space UID = " + space_uid)

         Space UID = a5d3879c-fce2-421a-a352-9e2bbb589332

In [58]: client.set.default_space(space_uid)

Out[58]: 'SUCCESS'

In [59]: client.software_specifications.list()

         ----------------------------  ----------------------------------  ----
```

Creating model deployment pipeline

In [60]:
```
#Set Python Version
software_spec_uid = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
software_spec_uid
```

Out[60]:
```
'12b83a17-24d8-5082-900f-0ab31fbfd3cb'
```

In [61]:
```
model_details = client.repository.store_model(model = multiple_lin_reg, meta_props={
    client.repository.ModelMetaNames.NAME: "UAEP_Multiple_Linear_Regression",
    client.repository.ModelMetaNames.TYPE: "scikit-learn_1.0",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
)

model_id = client.repository.get_model_id(model_details)
```

In [62]:
```
model_id
```

Out[62]:
```
'8083e827-e81f-40d1-84ab-20d511771869'
```

Deployment Test Code Snippet:

```
import requests


API_KEY = "<Your API Key>"
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}


payload_scoring   =   {"input_data":   [{"field":   [["GRE   Score","TOEFL
```

```
Score","University    Rating","SOP","LOR    ","CGPA",    "Research"]],
"values": [[326, 110, 2, 3.5, 4, 9.23, 1]]}]}

response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/uaep_deployment/predictions
?version=2022-11-12', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
print("Scoring response")
print(response_scoring.json())
```

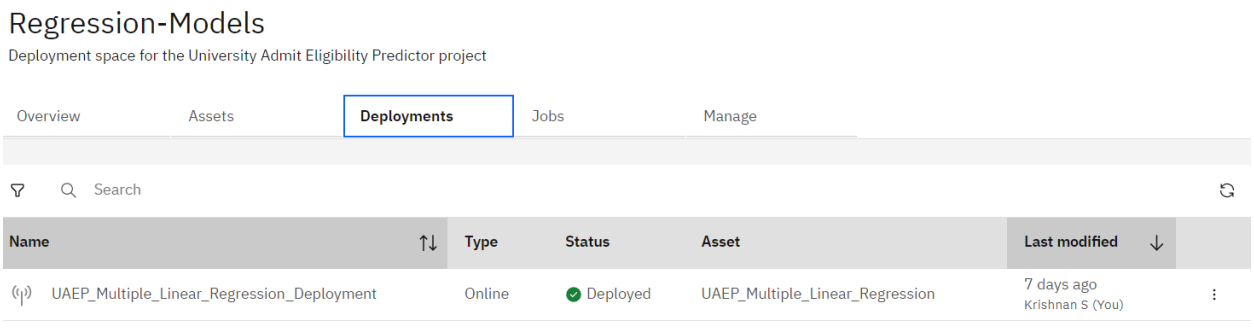
Output:

```
Scoring response
{'predictions': [{'fields': ['prediction'], 'values': [[[0.8448151378927107]]]}]}
```

```
probability = response_scoring.json()['predictions'][0]['values'][0][0][0]
probability
```

```
0.8448151378927107
```

Deployment Snapshot:



2. **Making the probabilistic prediction**

This function takes in the candidate's gre score, toefl score, cgpa, sop, lor ratings and a boolean field representing whether he has prior research internship experience or not. Using the deployed multiple linear regression model, we estimate the probability of acceptance to universities having the specified rank. If the probability percentage is over 66.67 %, the candidate stands a very good chance to get addmitted to the university. Otherwise, we display "Low Chances", implying that the candidate shouldn't apply to that particular university.

Program Code:

```
def pred(gre, toefl, sop, lor, cgpa, resc, univ_rank):


        # Preprocessing user input
        # ielts = convert_toefl_to_ielts(toefl)


        if resc == 'Yes':
            resc = 1
        else:
            resc = 0


                payload_scoring = {"input_data": [{"field": [["GRE
Score","TOEFL    Score","University    Rating","SOP","LOR    ","CGPA",
"Research"]],
        "values": [[gre, toefl, univ_rank, sop, lor, cgpa, resc]]}]}


                    response_scoring   =   requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/uaep_deployment/predictions
```

```
?version=2022-11-12', json=payload_scoring,
        headers={'Authorization': 'Bearer ' + mltoken})


                                                    prediction      =
response_scoring.json()['predictions'][0]['values'][0][0]
```