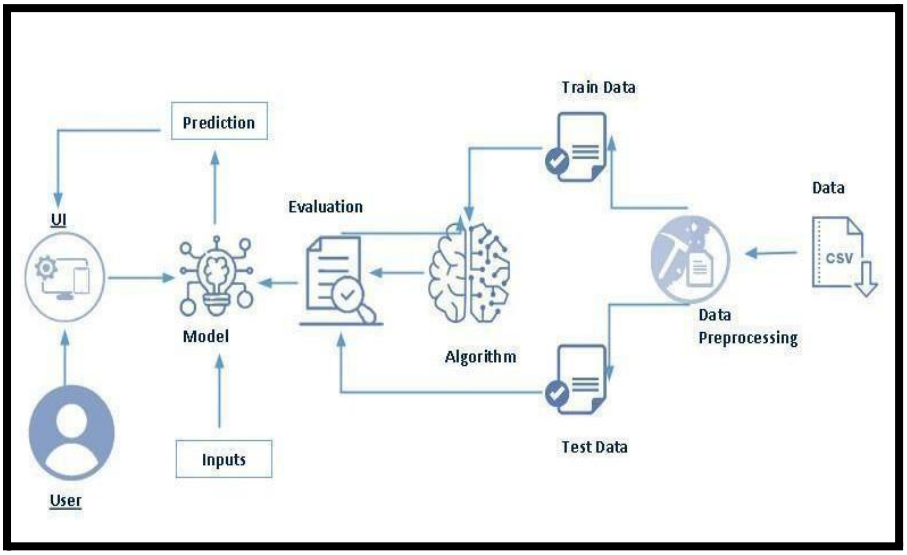
Intelligent Admissions: The Future of UniversityDecision Making with Machine Learning

1.1 OVERVIEW:

Artificial intelligence (AI) has gradually become accepted by colleges and universities as an effective tool for automating a number of tasks effectively and efficiently. AI-generated emails can remind students about important deadlines, prompt them to register for classes, turn in assignments and pay their fees on time. And, in a particularly controversial use, AI-based software is increasingly able to detect plagiarized assignments.

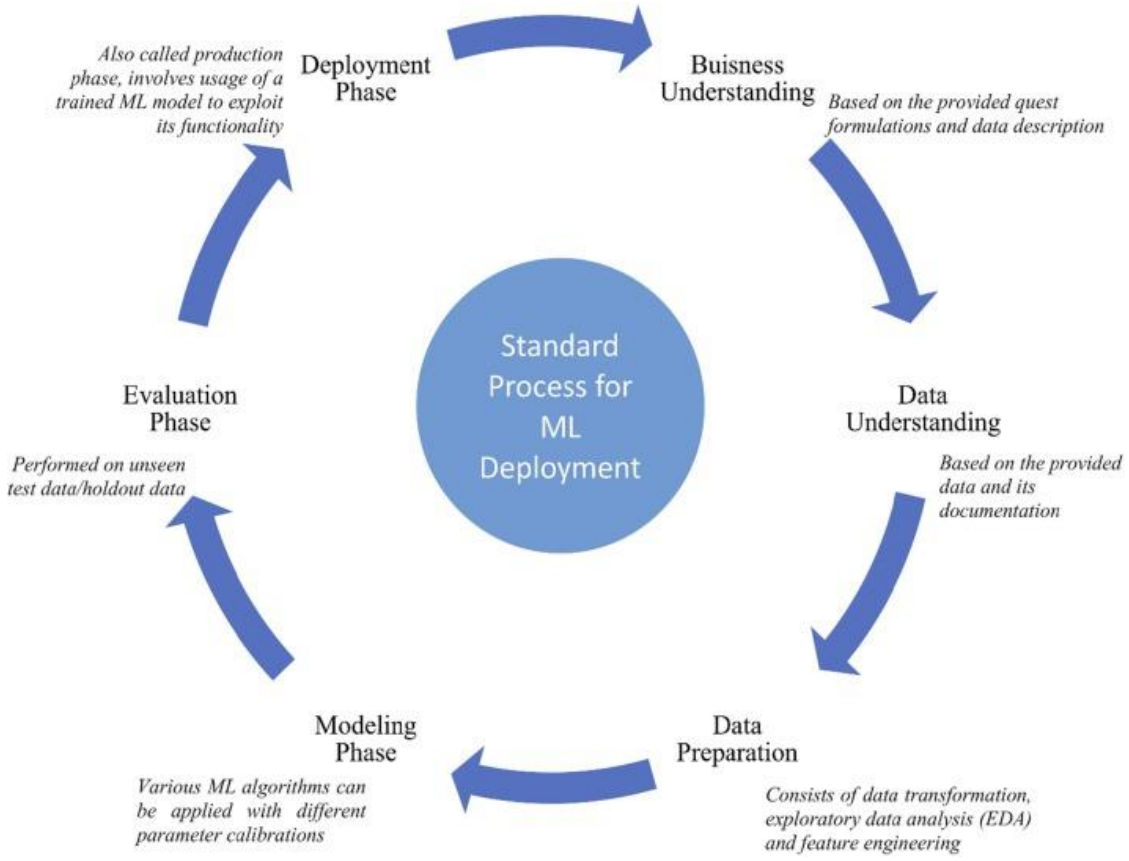
AI expands into these core university practices, new concerns are also being raised about the tool's threats to personal privacy and its vulnerability to systematic bias.

Technical Architecture:



TechnicalAbout the Project:

Machine learning is merely based on predictions made based on experience. It enables machines to make data-driven decisions, which is more efficient than explicitly programming to carry out certain tasks. These algorithms are designed in a fashion that gives exposure to new data that can help organisations learn and improve their strategies.



A PROJECT DESCRIPTION:

University admission is the process by which students are selected to attend a college or university. The process typically involves several steps, including submitting an application, taking entrance exams, and participating in interviews or other evaluations.

Students are often worried about their chances of admission in University. the university admission process for students can be demanding, but by being well-informed, prepared, and organized, students can increase their chances of being admitted to the university of their choice.

The aim of this project is to help students in short listing universities with their profiles. Machine learning algorithms are then used to train a model on this data, which can be used to predict the chances of future applicants being admitted. With this project, students can make more informed decisions about which universities to apply to, and universities can make more efficient use of their resources by focusing on the most promising applicants.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.

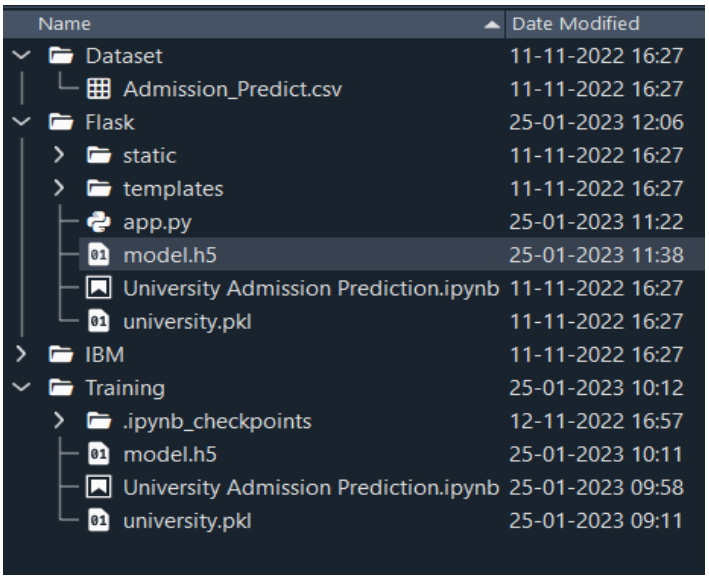
Project Flow:

- User interacts with the UI to enter the input.
- Entered input is analysed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the UITo accomplish this, we have to complete all the activities listed below,

- Define Problem / Problem Understanding
  - Specify the business problem
  - Business requirements
  - Literature Survey
  - Social or Business Impact.
- Data Collection & Preparation
  - Collect the dataset
  - Data Preparation
- Exploratory Data Analysis
  - Descriptive statistical
  - Visual Analysis
- Model Building
  - Training the model in multiple algorithms
  - Testing the model
- Performance Testing & Hyperparameter Tuning
  - Testing model with multiple evaluation metrics
  - Comparing model accuracy before & after applying hyperparameter tuning
- Model Deployment
  - Save the best model
  - Integrate with Web Framework
- Project Demonstration & Documentation
  - Record explanation Video for project end to end solution
  - Project Documentation-Step by step project development procedure

Project Structure:

Create the Project folder which contains files as shown below



- We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.
- model.h5 is our saved model. Further we will use this model for flask integration.
- Training folder contains a model training file.

1.2 PURPOSE:

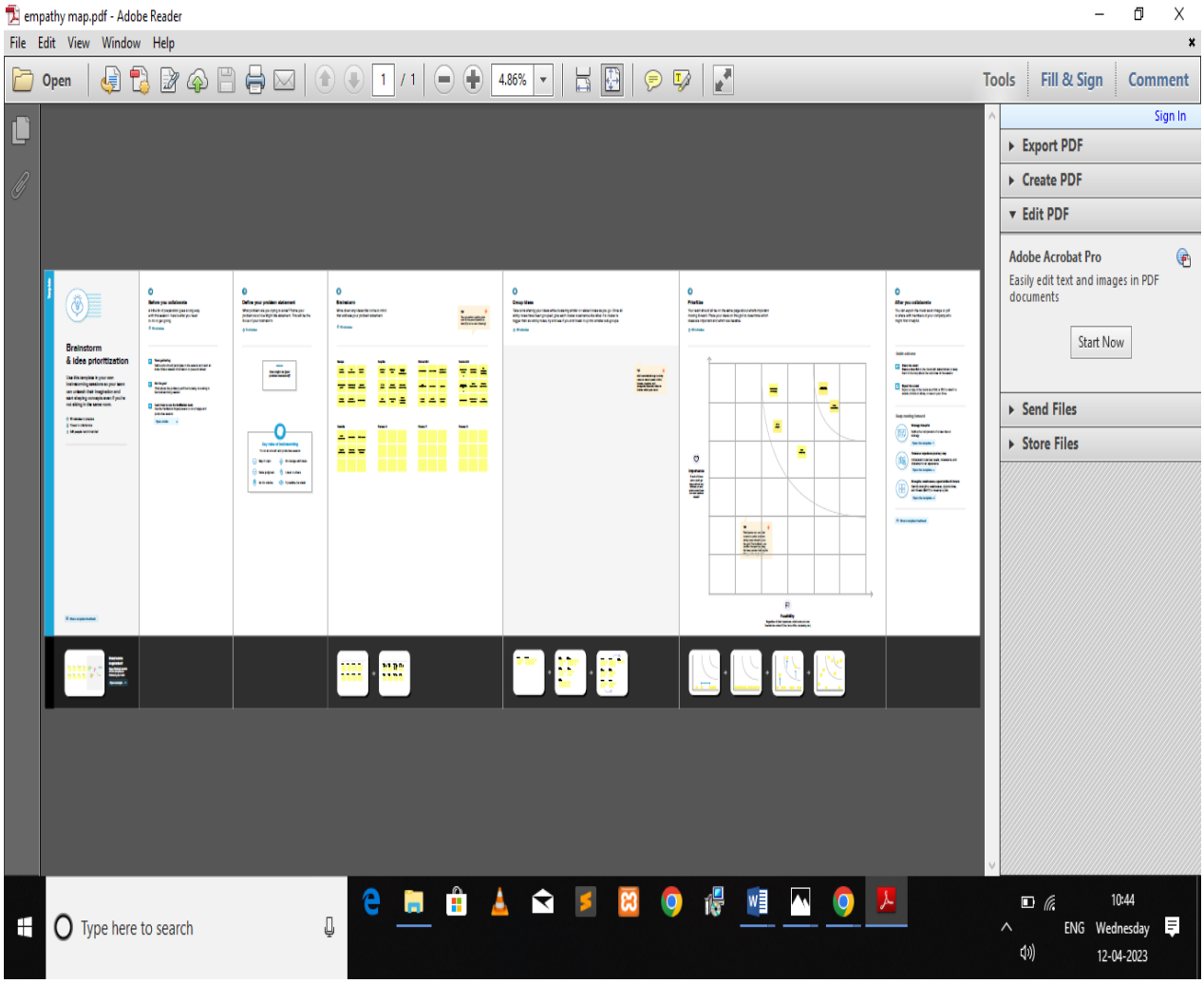
Machine Learning has an additional benefit of processing large chunks of data that is sometimes tiresome for men to do and eventually lead to a failure in making the right decision. It is easily adaptable to new and complex data. After processing the data, it is capable of analyzing any flaws or errors. These also help in creating effective plans of Actions for improvement. There is a co-relation between inputs and outputs in the process of decision-making. These points are extremely useful for ventures that work mainly around risk management.

2. PROBLEM DEFINITION AND DESIGN THINKING

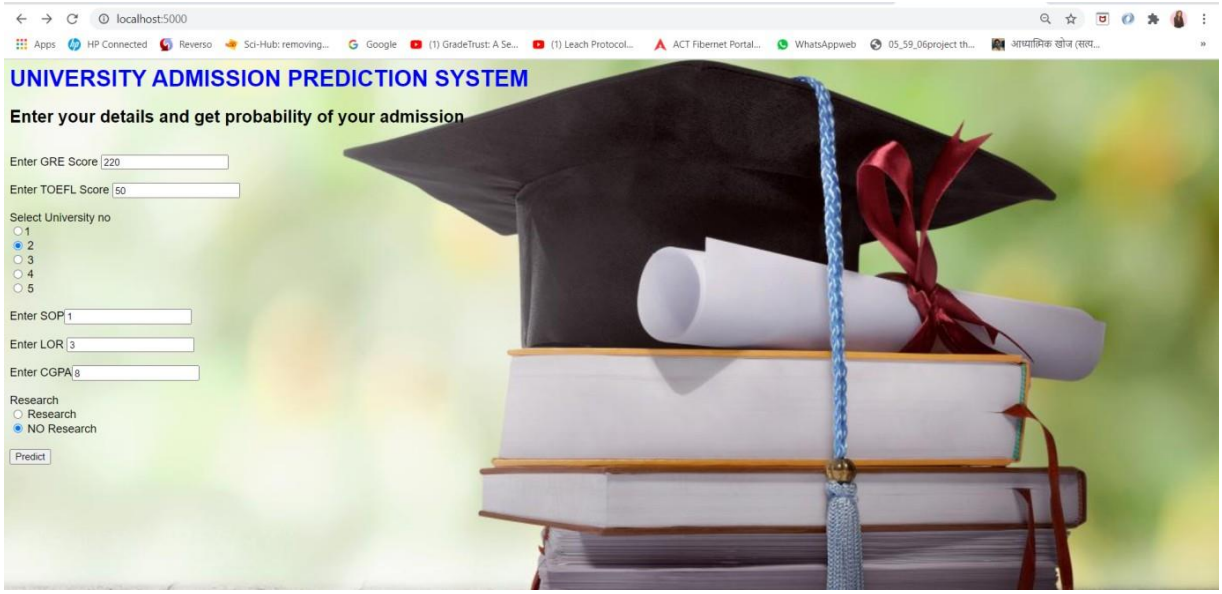
Machine learning has become an increasingly popular tool in recent years, given its ability to automatically detect patterns in data and make predictions about future events. This can be extremely useful for making decisions in a wide range of domains, from financial trading to medical diagnoses.

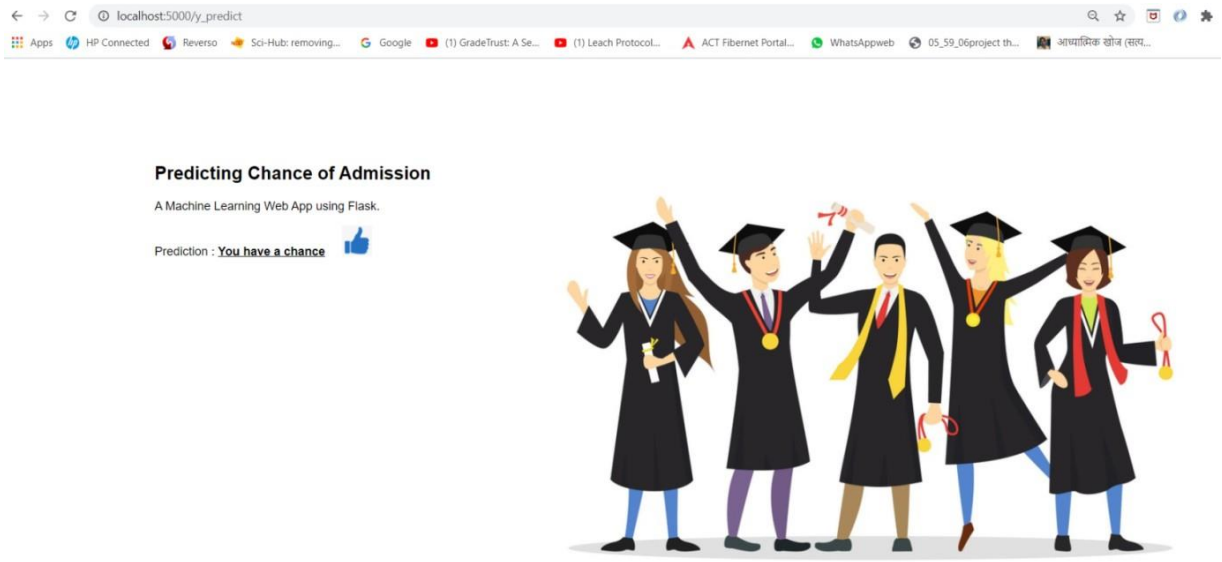
2.1 EMPATHY MAP

2.2 Ideation and Brainstorming Map:



3.RESULT:


localhost:5000
UNIVERSITY ADMISSION PREDICTION SYSTEM
Enter your details and get probability of your admission
Enter GRE Score 220
Enter TOEFL Score 50
Select University no
1
2
3
4
5
Enter SOP 1
Enter LOR 3
Enter CGPA 8
Research
Research
NO Research
Predict


localhost:5000/y_predict
Predicting Chance of Admission
A Machine Learning Web App using Flask.
Prediction : You have a chance

4.ADVANTAGE AND DISADVANTAGE:

Advantage:

1. Providing better information: Since machine learning technology can sift through extremely large amounts of data, it is able to also provide better information to decision makers.

2. Automating the process: In many industries, it is simply not possible for human beings to make optimal decisions all of the time. This is especially true in industries where the data is constantly changing, such as financial markets. In these cases, machine learning algorithms can be used to automatically make decisions as trends change and evolve.

3. Improving accuracy: By identifying patterns in data that humans may not be able to see, machine learning can drastically improve the accuracy of its predictions. It can also create models that simulate different decision scenarios and help identify the best course of action. And as new data becomes available, machine learning can be used to constantly update and refine decision models.

Disadvantage:

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions

coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

5.APPLICATION

1. Decisions in business operations

Machine Learning algorithms come to the rescue in areas built on a constant flow of heterogeneous data, whether it is several financial reports, payrolls, procurement, the analysis of employee productivity, or predicting further churn rates.

Overall, AI, in terms of inner business processes, is able to leverage business intelligence and make a company data-driven in many aspects, including decision making.

2. Complex problem-solving

The potential of AI in decision making is robust, but you can solve multilayer and complex problems, too.

Artificial Intelligence here gathers tons of different data and conducts an interdisciplinary study. Eventually, there's a way to leverage anything from product development stages to digital marketing approaches of product promotion.

Also, it's a way to optimize various types of predictions and risk management. For example, you can predict and optimize pricing with the help of AI tools.

3. Strategic changes

AI allows better planning of production, managing all restrictions, reducing shortcomings in operations, and improving manufacturing.

It also helps to anticipate and adequately plan product customization, enhance postponement processes, and maintain efficiency with high levels of customer satisfaction.

4. Customer-related decisions

AI can be valuable for customer service management, personalized customer communication, evaluation of customer behavior, predicting consumer trends and patterns.

Artificial intelligence enables automatic recognition and profiling of potential customers.

5. Performance assessment

Firstly, it relates to people's performance evaluation and afterward decisions. AI is capable of minimizing human errors and making employee performance data more transparent.

AI can also recommend online courses, training, and development programs to employees based on their performance history.

6.CONCLUSION

Student admission problem is very important in educational institutions. In this project addresses machine learning models to predict the chance of a student to be admitted. This will assist students to know in advance if they have a chance to get accepted. Machine learning models were performed to predict the opportunity of a student to get admittedto a master's program. The machine learning models included are multiple linear regression, random forest, Multiple Linear Regression with Backward Elimination and random forest regression with backward elimination. Experiments show that the Linear Regression model surpasses other models.

Our aim would be to predict the "Chance of Admit" based on the different parameters that are provided in the dataset. We will achieve this aim by using the Linear Regression model. Based on the data that we have, we will split out data into training and testing sets. The Training set will have features and labels on which our model would be trained. The label here is the "Chance of Admit". If you think from a no-technical standpoint then label is basically the output that we want and features are the parameters that drive us towards the output. Once our model is trained, we willuse the trained model and run it on the test set and predict the output. Then we will compare the predicted results with the actual results that we have to see how our model performed. This whole process of training the model using features and known labels and later testing it to predict the output is called Supervised Learning.

7. FUTURE SCOPE

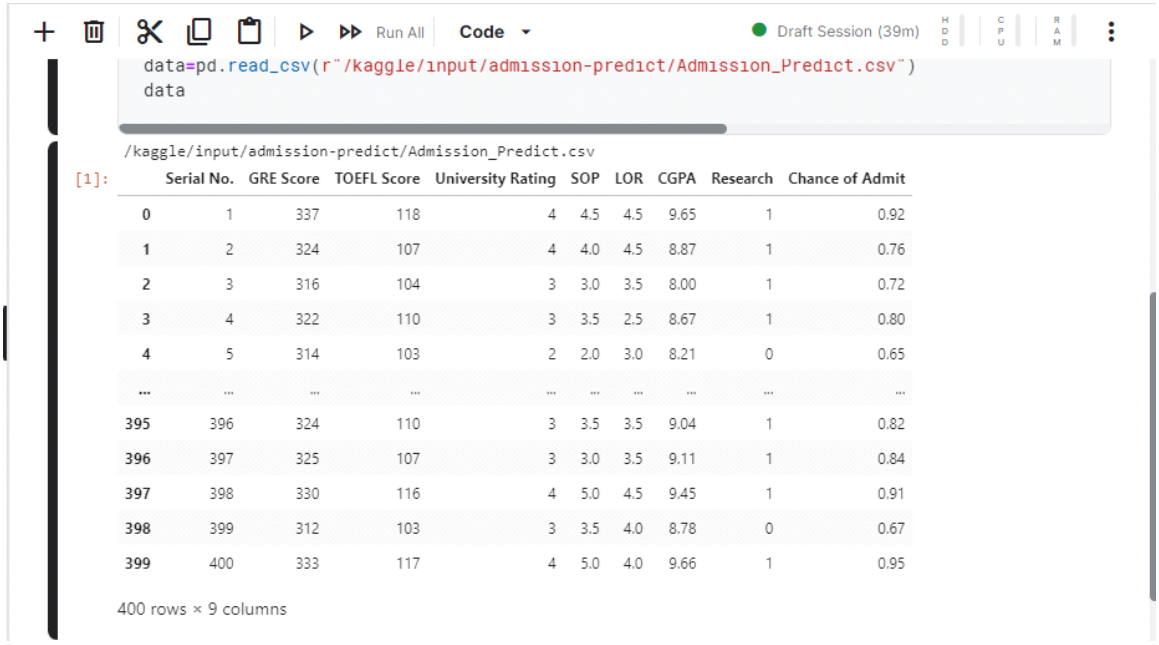
With the help of machine learning services like SDKs and APIs, developers are able to include and hone the intelligent capabilities into their applications. This will empower machines to apply the various things they come across, and accordingly carry out an array of duties like vision recognition, speech detection, and understanding of speech and dialect.

8.APPENDIX

Source Code:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline
```

data=pd.read_csv('Admission_Predic
t.csv



```
data=pd.read_csv(r"/kaggle/input/admission-predict/Admission_Predict.csv")
data
```

```
/kaggle/input/admission-predict/Admission_Predict.csv
```

[1]:

	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

data.info()

```
/kaggle/input/admission-predict/Admission_Predict.csv
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Serial No.         400 non-null    int64
 1   GRE Score          400 non-null    int64
 2   TOEFL Score        400 non-null    int64
 3   University Rating  400 non-null    int64
 4   SOP                400 non-null    float64
 5   LOR                400 non-null    float64
 6   CGPA               400 non-null    float64
 7   Research           400 non-null    int64
 8   Chance of Admit    400 non-null    float64
dtypes: float64(4), int64(5)
memory usage: 28.2 KB
```

+ Code + Markdown

data.isnull().any()

```
[3]: Serial No.           False
     GRE Score            False
     TOEFL Score          False
     University Rating    False
     SOP                  False
     LOR                  False
     CGPA                 False
     Research             False
     Chance of Admit      False
     dtype: bool
```

data=data.rename(columns ={'Chance of Admit': 'Chance of Admit'})

```
data=data.rename(columns={'Chance of Admit ':'Chance of Admit'})
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Serial No.         400 non-null    int64
 1   GRE Score          400 non-null    int64
 2   TOEFL Score        400 non-null    int64
 3   University Rating  400 non-null    int64
 4   SOP                400 non-null    float64
 5   LOR                400 non-null    float64
 6   CGPA               400 non-null    float64
 7   Research           400 non-null    int64
 8   Chance of Admit    400 non-null    float64
dtypes: float64(4), int64(5)
memory usage: 28.2 KB
```

```
data.describe()
```

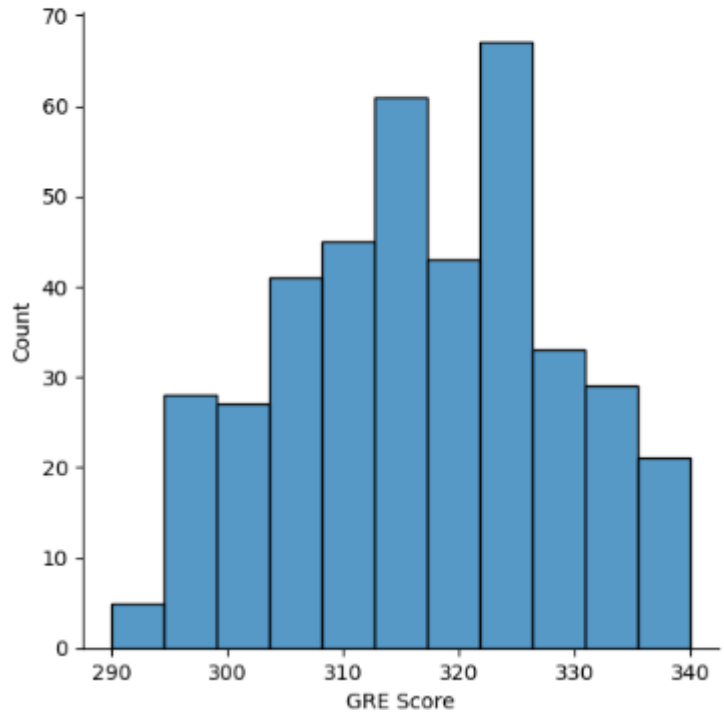
	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	200.500000	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.547500	0.724350
std	115.614301	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.498362	0.142609
min	1.000000	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.000000	0.340000
25%	100.750000	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.000000	0.640000
50%	200.500000	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.000000	0.730000
75%	300.250000	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.000000	0.830000
max	400.000000	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.000000	0.970000

Visual analysis

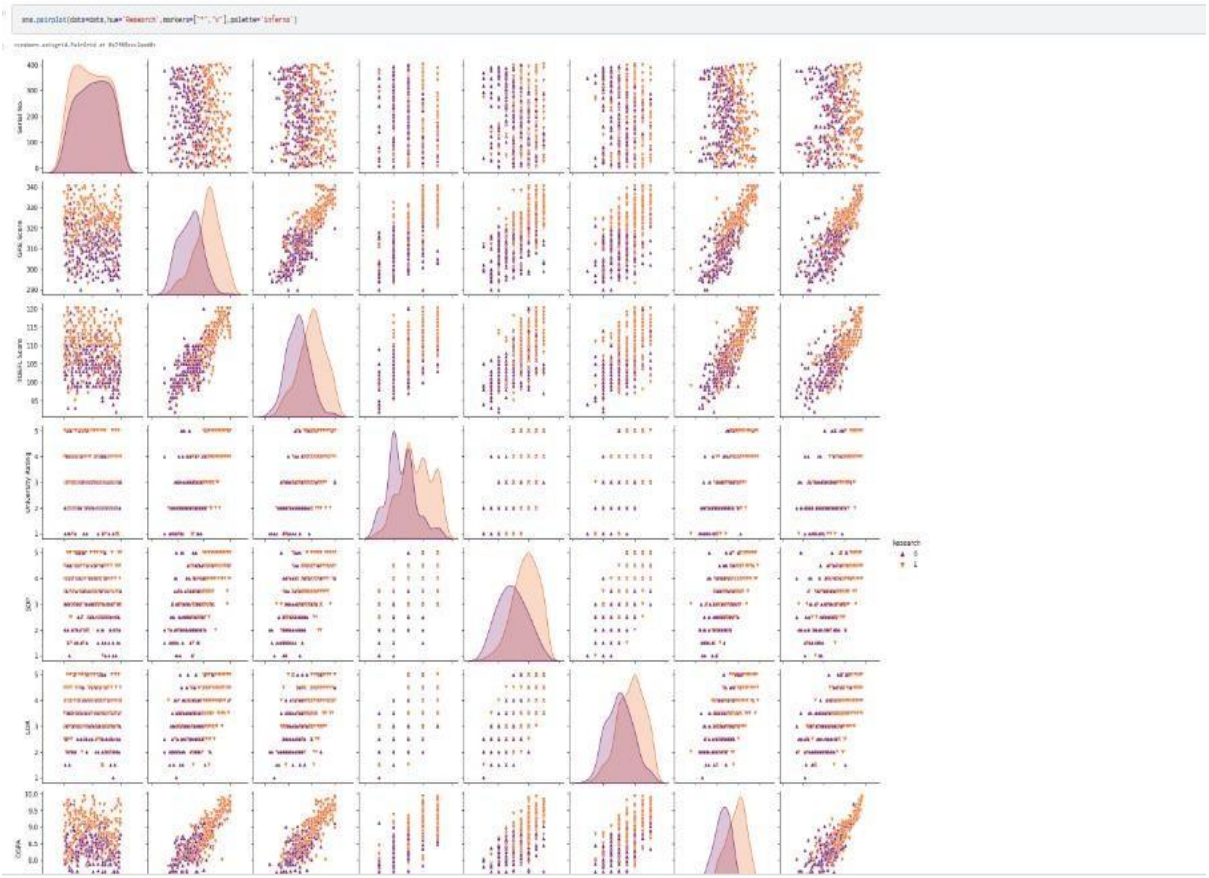
Univariate analysis

```
sns.displot(data['GRE Score'])
```

```
<seaborn.axisgrid.FacetGrid at 0x7f660df3ab10>
```

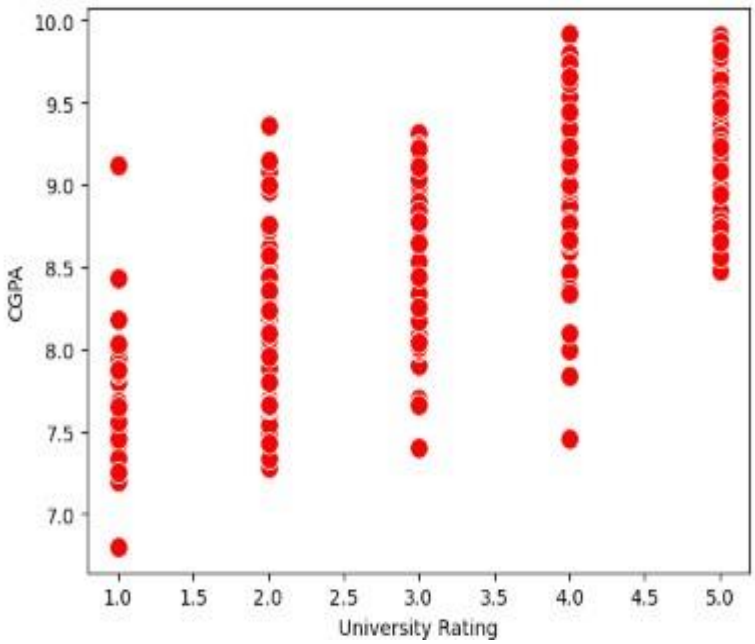


Bivariate analysis

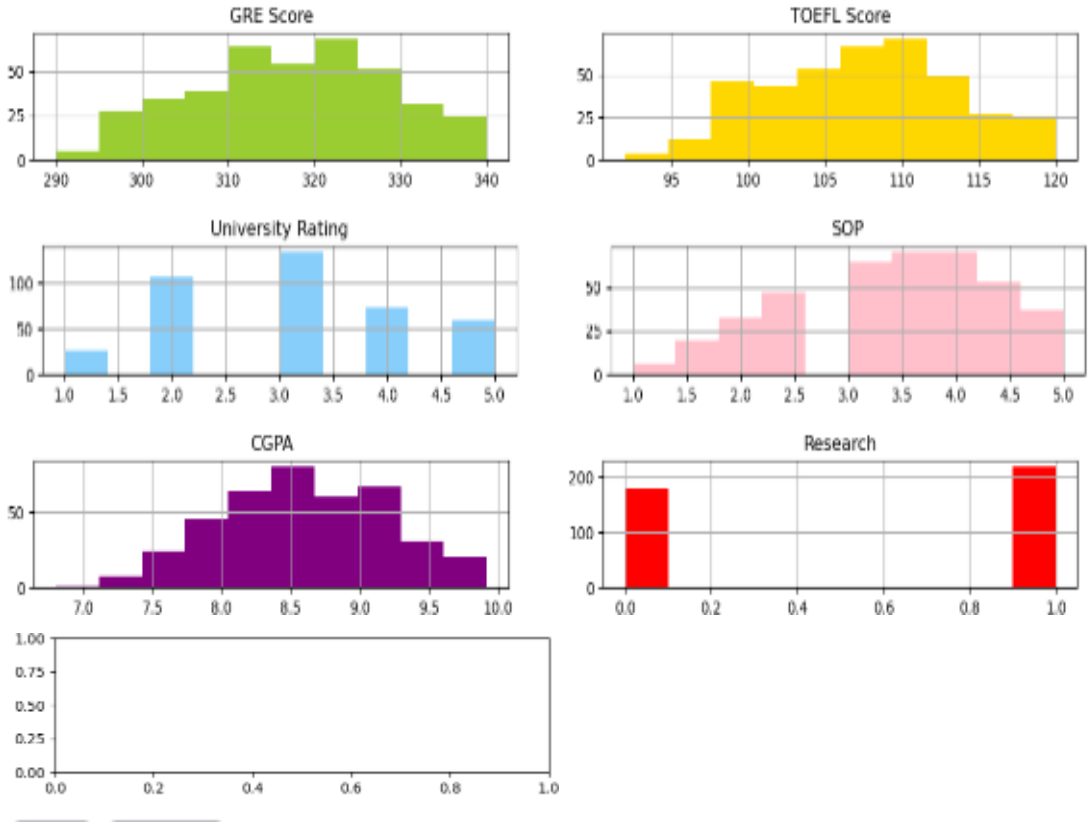


```
sns.scatterplot(x='University Rating',y='CGPA',data=data,color='Red',s=100)
```

```
<AxesSubplot:xlabel='University Rating', ylabel='CGPA'>
```



```
category =['GRE Score','TOEFL Score','University Rating','SOP','CGPA','Research']
color=['yellowgreen','gold','lightskyblue','pink','purple','red','orange','gray']
start = True
for i in np.arange(4):
    fig = plt.figure(figsize=(14,8))
    plt.subplot2grid((4,2),(i,0))
    data[category[2*i]].hist(color=color[2*i],bins=10)
    plt.title(category[2*i])
    plt.subplot2grid((4,2),(i,1))
    data[category[2*i+1]].hist(color=color[2*i+1],bins=10)
    plt.title(category[2*i+1])

    plt.subplots_adjust(hspace = 0.7, wspace = 0.2)
    plt.show()
```



Scaling the Data

```
from sklearn.preprocessing import MinMaxScaler

sc = MinMaxScaler()

x=sc.fit_transform(x)

data()
```

Splitting data into x and y

```
x=data.iloc[:,0:7].values

x
```

```
array([[  1.  , 337.  , 118.  , ...,   4.5 ,   4.5 ,   9.65],
       [  2.  , 324.  , 107.  , ...,   4.  ,   4.5 ,   8.87],
       [  3.  , 316.  , 104.  , ...,   3.  ,   3.5 ,   8.  ],
       ...,
       [398.  , 330.  , 116.  , ...,   5.  ,   4.5 ,   9.45],
       [399.  , 312.  , 103.  , ...,   3.5 ,   4.  ,   8.78],
       [400.  , 333.  , 117.  , ...,   5.  ,   4.  ,   9.66]])
```

40]:
```
y=data.iloc[:,7:].values

y
```

40_
```
array([[1.  , 0.92],
       [1.  , 0.76],
       [1.  , 0.72],
       [1.  , 0.8 ],
       [0.  , 0.65],
       [1.  , 0.9 ],
       [1.  , 0.75],
       [0.  , 0.68],
       [0.  , 0.5 ],
       [0.  , 0.45],
       [1.  , 0.52],
       [1.  , 0.84],
       [1.  , 0.78],
       [1.  , 0.62],
       [1.  , 0.61],
       [0.  , 0.54],
       [0.  , 0.66],
       [1.  , 0.65],
       [0.  , 0.63],
       [0.  , 0.62],
       [1.  , 0.64],
       [0.  , 0.7 ],
       [1.  , 0.94],
       [1.  , 0.95],
       [1.  , 0.97],
       [1.  , 0.94],
       [0.  , 0.76],
       [1.  , 0.44],
       [0.  , 0.46],
       [0.  , 0.54],
       [1.  , 0.65],
       [1.  , 0.74],
       [1.  , 0.91],
       [1.  , 0.9 ],
       [1.  , 0.94],
       [1.  , 0.88],
       [0.  , 0.64],
       [0.  , 0.58],
       [0.  , 0.52],
       [0.  , 0.48],
       [1.  , 0.46],
```

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,random_state=101)

#random_state acts as the seed for the random number generator during the split
```

Let us convert it into classification problem

```
y_train=(y_train>0.5)

y_train
```

```
array([[ True,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [ True, False],
       [ True,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [False, False],
       [ True, False],
       [False,  True],
       [ True,  True],
       [False,  True],
       [False, False],
       [False,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [ True,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [False,  True],
       [ True,  True],
       [False,  True],
       [False,  True],
       [False,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [False,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [False,  True],
       [ True,  True],
       [False, False],
       [False,  True],
       [ True,  True],
       [False,  True],
       [ True,  True]])
```

Intelligent Admissions: The Future of University Decision Making with Machine Learning

```
from sklearn.linear_model.logistic import LogisticRegression
cls =LogisticRegression(random_state =0)

lr=cls.fit(x_train, y_train)
```

```
C:\Users\Tulasi\anaconda3\lib\site-packages\sklearn\utils\validation.py:760: DataConversionWarn
array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

```
y_pred =lr.predict(x_test)
y_pred
```

ANN model

```
#Libraries to train Neural network
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.layers import Dense, Activation, Dropout
from tensorflow.keras.optimizers import Adam
```

```
# Initialize the model
model=keras.Sequential()

# Add input layer
model.add(Dense(7,activation ='relu',input_dim=7))

# Add hidden layers
model.add(Dense(7,activation='relu'))

# Add output layer
model.add(Dense(1,activation='linear'))

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 7)                 56
_________________________________________________________________
dense_1 (Dense)              (None, 7)                 56
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 8
=================================================================
Total params: 120
Trainable params: 120
Non-trainable params: 0
```

```
]: model.fit(x_train, y_train, batch_size = 20, epochs = 100)
Epoch 1/100
16/16 [==============================] - 0s 2ms/step - loss: 1.7298 - accuracy: 0.0781
Epoch 2/100
16/16 [==============================] - 0s 1ms/step - loss: 1.3143 - accuracy: 0.0844
Epoch 3/100
16/16 [==============================] - 0s 1ms/step - loss: 1.0439 - accuracy: 0.1344
Epoch 4/100
16/16 [==============================] - 0s 1ms/step - loss: 0.8401 - accuracy: 0.3219
Epoch 5/100
16/16 [==============================] - 0s 1ms/step - loss: 0.6683 - accuracy: 0.5656
Epoch 6/100
16/16 [==============================] - 0s 1ms/step - loss: 0.5238 - accuracy: 0.7531
Epoch 7/100
16/16 [==============================] - 0s 1ms/step - loss: 0.3918 - accuracy: 0.8844
Epoch 8/100
16/16 [==============================] - 0s 1ms/step - loss: 0.2865 - accuracy: 0.9250
Epoch 9/100
16/16 [==============================] - 0s 1ms/step - loss: 0.2254 - accuracy: 0.9312
Epoch 10/100
```

Testing the model

```
]: model.compile(loss = 'binary_crossentropy', optimizer = 'adam',metrics = ['accuracy'])

]: model.fit(x_train, y_train, batch_size = 20, epochs = 100)

]: from sklearn.metrics import accuracy_score


   # Make predictions on the training data
   train_predictions = model.predict(x_train)

   print(train_predictions)

]: # Get the training accuracy
   train_acc = model.evaluate(x_train, y_train, verbose=0)[1]

   print(train_acc)

   0.9281250238418579

]:
   # Get the test accuracy
   test_acc = model.evaluate(x_test, y_test, verbose=0)[1]


   print(test_acc)

   0.875

]: print(classification_report(y_test,pred))
```

```
[ ]:
   pred=model.predict(x_test)
   pred = (pred>0.5)
   pred
```

```
array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True])
```

Model Deployment::Save the best model:

```
  # Save the model in HDF5 format
    model.save('model.h5')
```

```
  File "/tmp/ipykernel_27/3721887466.py", line 2
    model.save('model.h5')
    ^
IndentationError: unexpected indent
```

Integrate with Web Framework:

```
import numby as np
from flask import Flask, request,jsonify, render_template
import pickle
app = Flask(__name__)
# Import necessary libraries
from tensorflow.keras.models import load_model

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

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
/tmp/ipykernel_27/4117061979.py in <module>
----> 1 import numby as np
      2 from flask import Flask, request,jsonify, render_template
      3 import pickle
      4 app = Flask(__name__)
      5 # Import necessary libraries

ModuleNotFoundError: No module named 'numby'
```

```
#load model trained model
#Load your trained model
model = load_ model('model.h5')
```

```
  File "/tmp/ipykernel_27/552793924.py", line 3
    model = load_ model('model.h5')
                    ^
SyntaxError: invalid syntax
```

```
@app.route('/')
def home():
  return render_template('Demo2.html')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
/tmp/ipykernel_27/3305153803.py in <module>
----> 1 @app.route('/')
      2 def home():
      3   return render_template('Demo2.html')

NameError: name 'app' is not defined
```

Retrieves the value from UI:

```
@app.route('/')
def home():
  return render_template('Demo2.html')

@app.route('/y_predict',methods=['POST'])
def y_predict():

  For rendering results on HTML GUI

#min max scaling
min1=[290.0, 92.0, 1.0, 1.0, 1.0, 6.8, 0.0]
max1=[340.0, 120.0, 5.0, 5.0, 9.92, 1.0]
k= [float(x) for x in request.from.values()]
p=[]
for i in range(7):
    l=(k[i]-min1[i])/(max1[i]-min1[i])
    p.append(l)
prediction = model.predict([p])
print(prediction)
output=prediction[0]
if(output==False):
  return render_template('noChance.html', prediction_text='You Dont have a chance of getting
else:
   return render_template('Chance.html', prediction_text='You  have a chance of getting admis
if__name__ == "__main__":
   app.run(debug=False)
```

```
  File "/tmp/ipykernel_27/3016622240.py", line 8
    For rendering results on HTML GUI
            ^
SyntaxError: invalid syntax
```