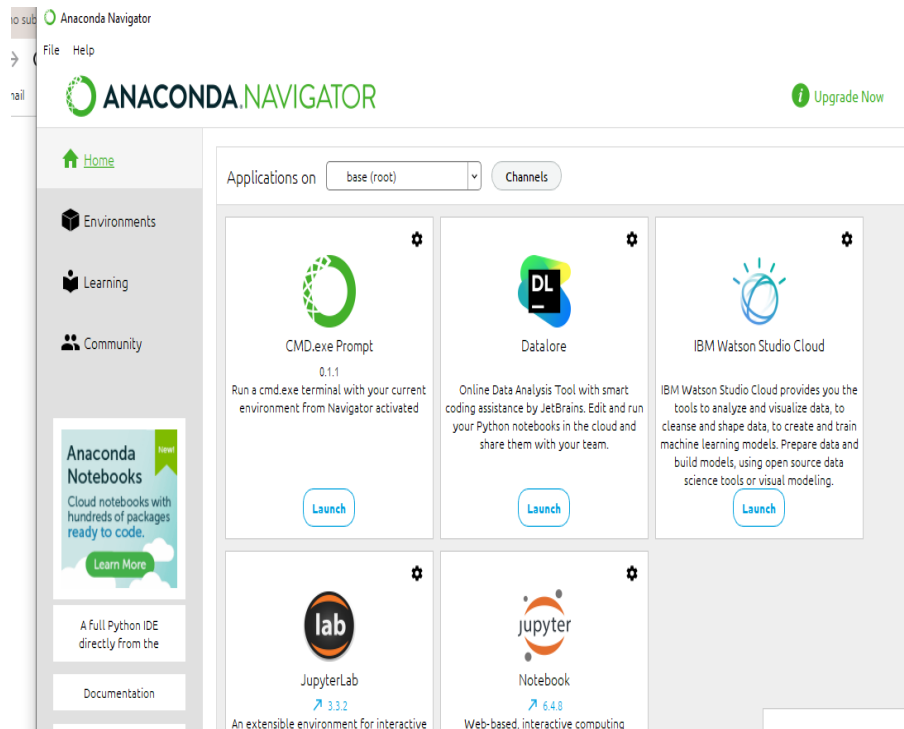


Pre-Requisites and Model Building Screenshots

Date	26 November 2022
Team ID	PNT2022TMID30201
Project Name	University Admit Eligibility Predictor

Pre-Requisites: Installing Anaconda



Importing libraries and the dataset :

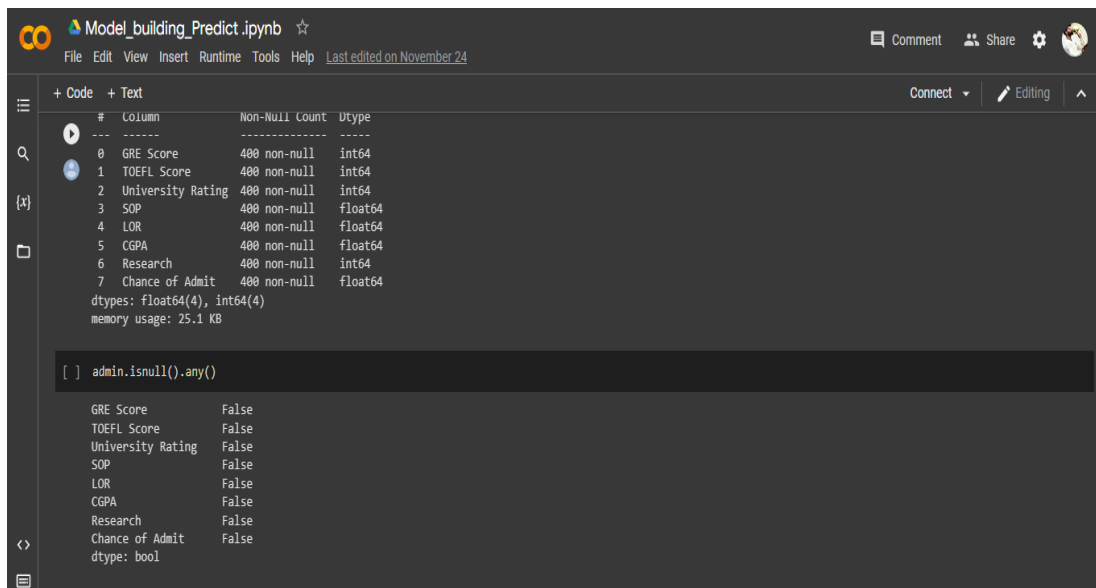
```
Model_building_Predict.ipynb
File Edit View Insert Runtime Tools Help Last edited on November 24

+ Code + Text
Connect Editing

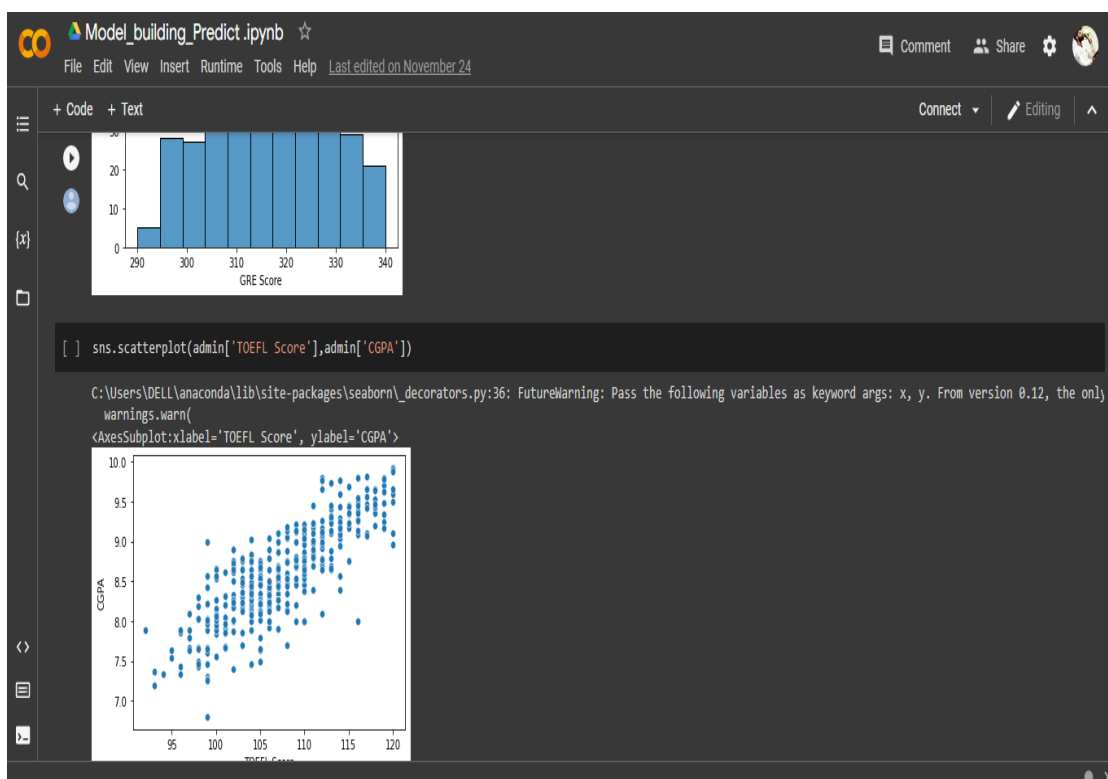
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

[ ] admin = pd.read_csv(r"C:\Users\DELL\Desktop\IBM\Admission_Predict.csv")
```

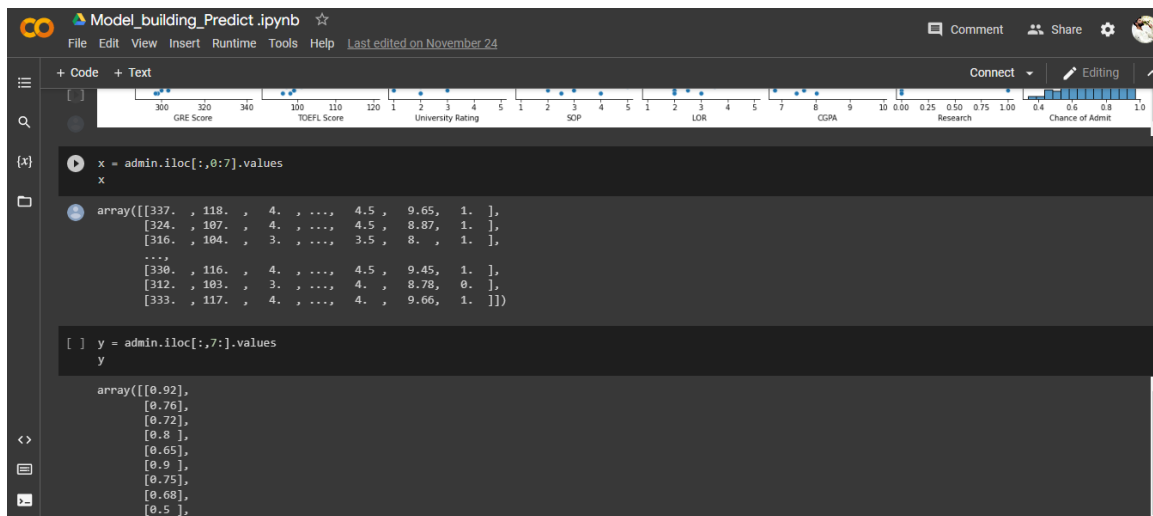
Handling the missing values :



Visualization :



Splitting Dependent and Independent columns :



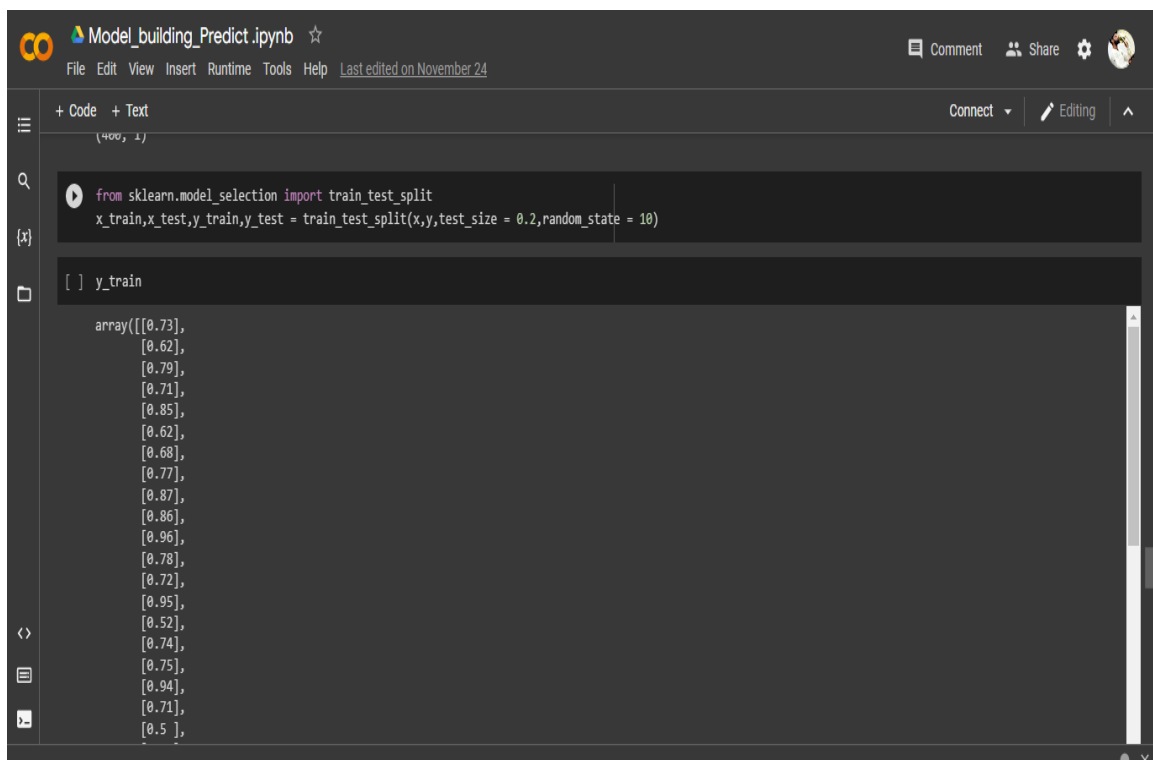
```
x = admin.iloc[:,0:7].values
x

array([[337., 118., 4., ..., 4.5, 9.65, 1. ],
       [324., 107., 4., ..., 4.5, 8.87, 1. ],
       [316., 104., 3., ..., 3.5, 8., 1. ],
       ...,
       [330., 116., 4., ..., 4.5, 9.45, 1. ],
       [312., 103., 3., ..., 4., 8.78, 0. ],
       [333., 117., 4., ..., 4., 9.66, 1. ]])

[ ] y = admin.iloc[:,7:].values
y

array([[0.92],
       [0.76],
       [0.72],
       [0.8 ],
       [0.65],
       [0.9 ],
       [0.75],
       [0.68],
       [0.5 ]])
```

Splitting the data into train and test :

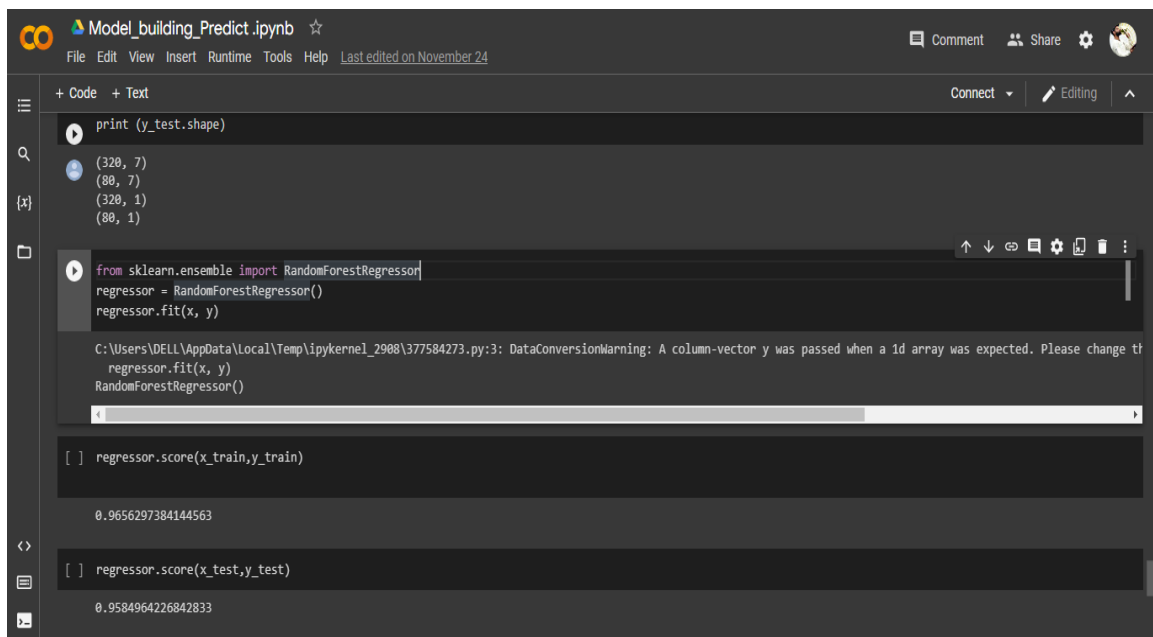


```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 10)

[ ] y_train

array([[0.73],
       [0.62],
       [0.79],
       [0.71],
       [0.85],
       [0.62],
       [0.68],
       [0.77],
       [0.87],
       [0.86],
       [0.96],
       [0.78],
       [0.72],
       [0.95],
       [0.52],
       [0.74],
       [0.75],
       [0.94],
       [0.71],
       [0.5 ]])
```

Training and testing the model :



A Jupyter Notebook titled "Model_building_Predict.ipynb" showing the training and testing of a RandomForestRegressor model. The notebook has a dark theme and includes a sidebar with icons for file explorer, search, and other functions. The code is executed in three cells. The first cell prints the shape of the test data. The second cell trains the model, with a warning message about a column-vector y. The third cell prints the scores for both training and testing data.

```
print (y_test.shape)
```

```
(320, 7)
(80, 7)
(320, 1)
(80, 1)
```

```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor()
regressor.fit(x, y)
```

```
C:\Users\DELL\AppData\Local\Temp\ipykernel_2908\377584273.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change it to a 1d array using y.ravel().
regressor.fit(x, y)
RandomForestRegressor()
```

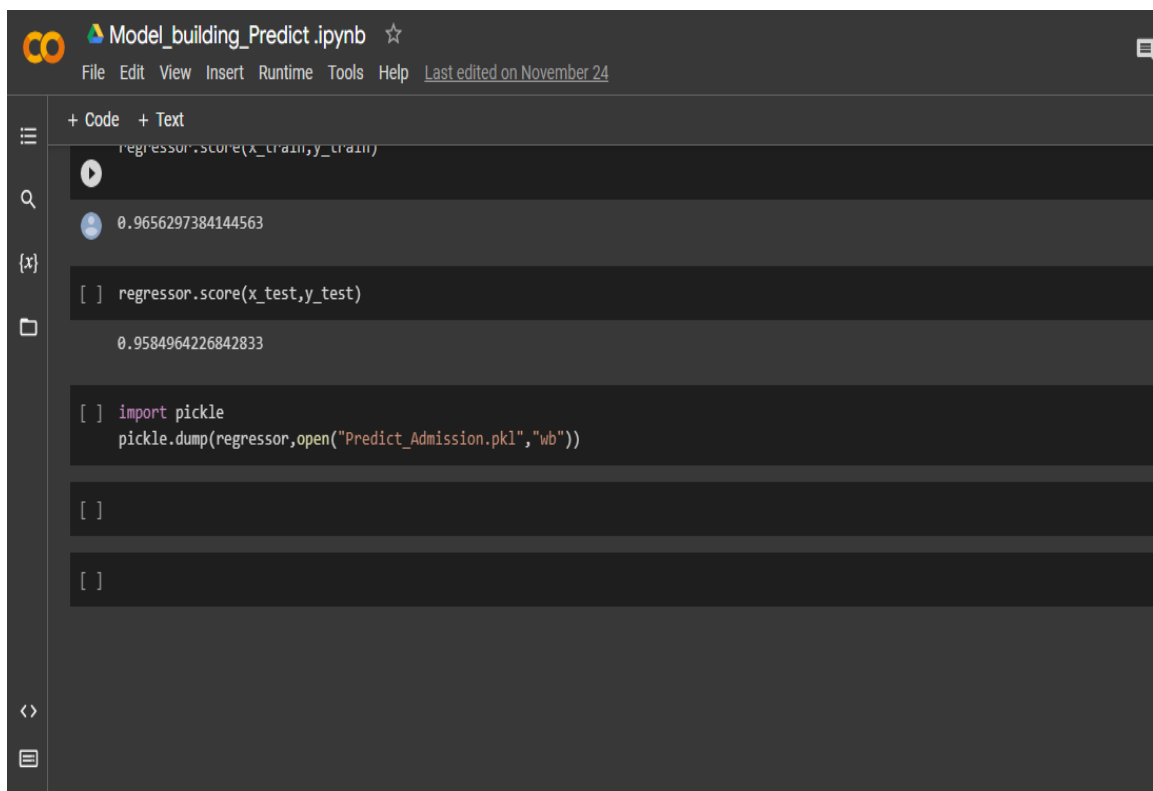
```
[ ] regressor.score(x_train,y_train)
```

```
0.9656297384144563
```

```
[ ] regressor.score(x_test,y_test)
```

```
0.9584964226842833
```

Save the model :



A Jupyter Notebook titled "Model_building_Predict.ipynb" showing the saving of the trained model. The notebook has a dark theme and includes a sidebar with icons for file explorer, search, and other functions. The code is executed in three cells. The first cell prints the training score. The second cell prints the testing score. The third cell saves the model using pickle.dump.

```
regressor.score(x_train,y_train)
```

```
0.9656297384144563
```

```
[ ] regressor.score(x_test,y_test)
```

```
0.9584964226842833
```

```
[ ] import pickle
pickle.dump(regressor,open("Predict_Admission.pkl","wb"))
```

```
[ ]
```

```
[ ]
```

Application building :

```
103         return render_template('Login.html', msg="No Account found! \n Please Signup")
104
105
106 @app.route("/predict", methods=['GET','POST'])
107 def predict():
108     if request.method == 'POST' :
109         g = int(request.form["gre"])
110         t = int(request.form["toefl"])
111         r = int(request.form["university_rating"])
112         s = float(request.form["sop"])
113         l = float(request.form["lor"])
114         c = float(request.form["cgpa"])
115         re = request.form["research"]
116         if re == "Research":
117             re = 1
118         else:
119             re = 0
120         S = [[g,t,r,s,l,c,re]]
121         print("*****")
122         payload_scoring = {"input_data": [{"field": ["GRE Score","TOEFL Score","University Rating","SOP","LOR","CGPA","Research Score"]}]}
123         response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/581582bf-4a79-41d8-a47b-85fc2')
124         print("Scoring response")
125         predictions = response_scoring.json()
126         output= predictions['predictions'][0]['values'][0][0]
127         r =round(output,2)
128         print(r)
129         predict = int(r*100)
130         p = '%'
131         #output = output*100
132         #z = str(output[0])
133
134         if r == 1:
```