Importing Libraries

In [127…

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

Loading the Datasets

In [128…

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def___iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includ
# You might want to remove those credentials before you share the notebook.
cos_client  =  ibm_boto3.client(service_name='s3',
    ibm_api_key_id='z0u1vzt8qo_0xobAJTt1SWEn6yZmKSWG7oH3mQR1KjUF',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket  =  'universityadmissionprediction-donotdelete-pr-d6ymzrsmcxijbu'
object_key = 'Admission_Predict.csv'

body  =  cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing___iter__method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter___=  types.MethodType(___iter__, bo

data = pd.read_csv(body)
data.head()
data
```

Out[128…

	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

Analysing the datas

In [129…

```
data.head(10)
```

Out[129…

	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

5	6	330	115	5	4.5	3.0	9.34	1	0.90
6	7	321	109	3	3.0	4.0	8.20	1	0.75
7	8	308	101	2	3.0	4.0	7.90	0	0.68
8	9	302	102	1	2.0	1.5	8.00	0	0.50
9	10	323	108	3	3.5	3.0	8.60	0	0.45

In [130…

```
data.tail(10)
```

Out[130…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
390	391	314	102	2	2.0	2.5	8.24	0	0.64
391	392	318	106	3	2.0	3.0	8.65	0	0.71
392	393	326	112	4	4.0	3.5	9.12	1	0.84
393	394	317	104	2	3.0	3.0	8.76	0	0.77
394	395	329	111	4	4.5	4.0	9.23	1	0.89
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

In [131…

```
data.dtypes
```

Out[131…

```
Serial No.             int64
GRE Score              int64
TOEFL Score            int64
University Rating      int64
SOP                  float64
LOR                  float64
```

```
CGPA                 float64
Research               int64
Chance of Admit      float64
dtype: object
```

In [132…

```
data.describe()
```

Out[132…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Re
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.
mean	200.500000	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.
std	115.614301	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.
min	1.000000	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.
25%	100.750000	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.
50%	200.500000	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.
75%	300.250000	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.
max	400.000000	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.

In [133…

```
data.skew()
```

Out[133…

```
Serial No.            0.000000
GRE Score            -0.062893
TOEFL Score           0.057216
University Rating     0.171260
SOP                  -0.275761
LOR                  -0.106991
CGPA                 -0.065991
Research             -0.191582
Chance of Admit      -0.353448
dtype: float64
```

In [134…

```
data.info()
```

```
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
```

In [ ]:

```

```

Handling Missing Values from the DataSets

In [135…

```
data.isnull() #method returns datafram object where all values replaced with boo
```

Out[135…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
395	False	False	False	False	False	False	False	False	False
396	False	False	False	False	False	False	False	False	False

397	False	False	False	False	False	False	False	False	False
398	False	False	False	False	False	False	False	False	False
399	False	False	False	False	False	False	False	False	False

400 rows × 9 columns

In [136…

```
data.notnull()
```

Out[136…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	True	True	True	True	True	True	True	True	True
1	True	True	True	True	True	True	True	True	True
2	True	True	True	True	True	True	True	True	True
3	True	True	True	True	True	True	True	True	True
4	True	True	True	True	True	True	True	True	True
...	...	...	...	...	...	...	...	...	...
395	True	True	True	True	True	True	True	True	True
396	True	True	True	True	True	True	True	True	True
397	True	True	True	True	True	True	True	True	True
398	True	True	True	True	True	True	True	True	True
399	True	True	True	True	True	True	True	True	True

400 rows × 9 columns

In [137…

```
data.sum()
```

Out[137…

```
Serial No.            80200.00
GRE Score            126723.00
TOEFL Score           42964.00
University Rating      1235 00
```

```
SOP                    1360.00
LOR                    1381.00
CGPA                   3439.57
Research                219.00
Chance of Admit         289.74
dtype: float64
```

In [138…

```
data.isna().sum()
```

Out[138…

```
Serial No.           0
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
dtype: int64
```

Data Visualization:

1.Univariate Analysis

2.Bivariate Analysis

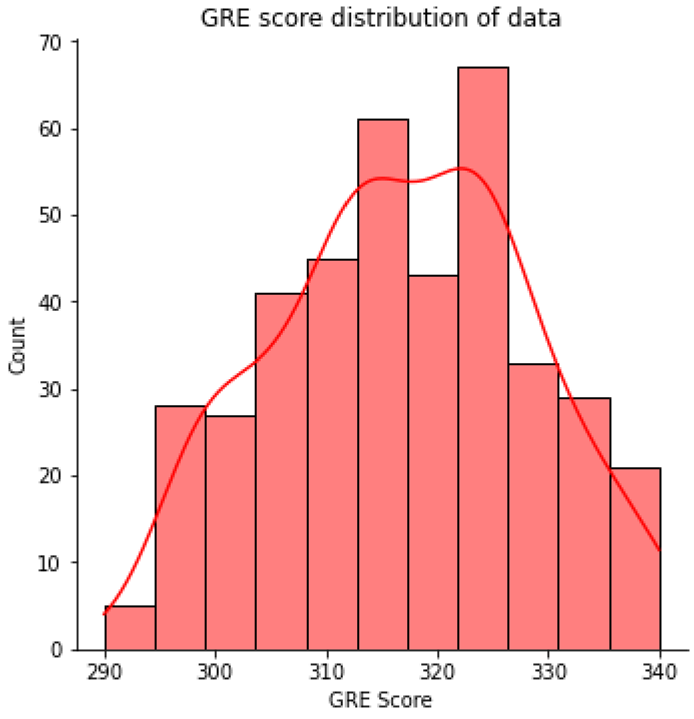
3.Multi-Variate Analysis

Univariate Analysis

In [139…

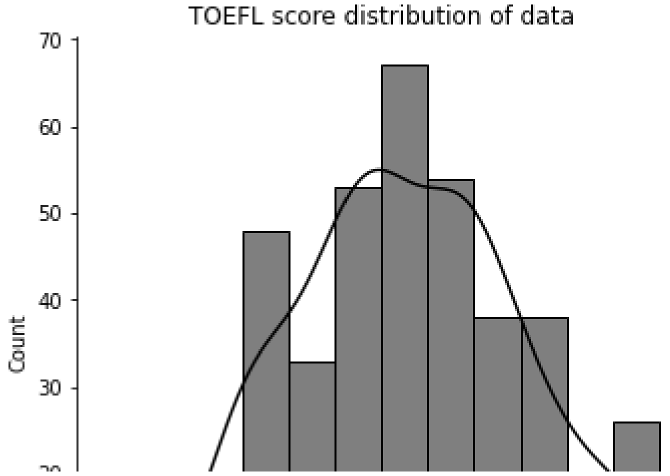
```
sns.displot(x=data["GRE Score"], kde=True, color='Red')
```

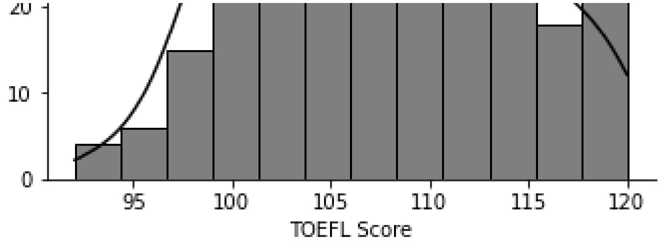
```
plt.title("GRE score distribution of data");
```



In [140…

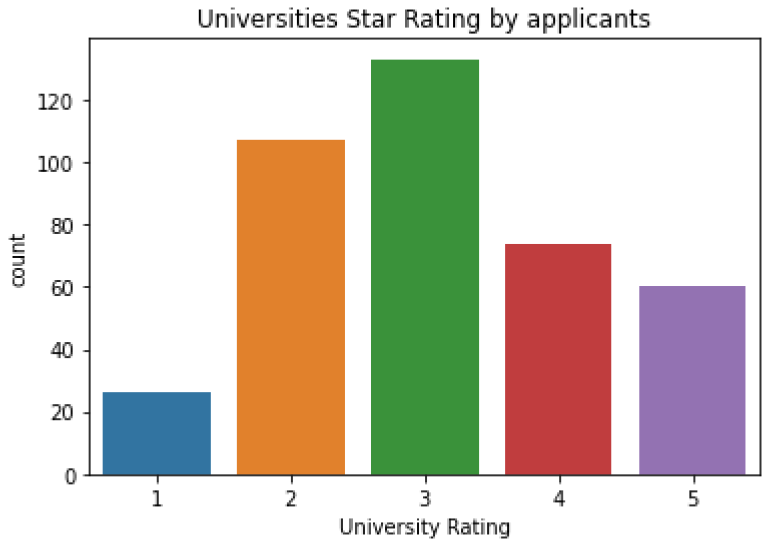
```
sns.displot(x=data["TOEFL Score"], kde=True, color='Black')
plt.title("TOEFL score distribution of data");
```





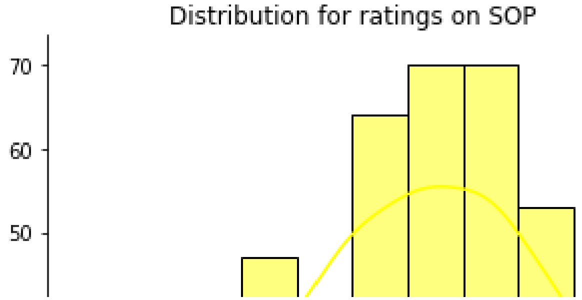
In [141…

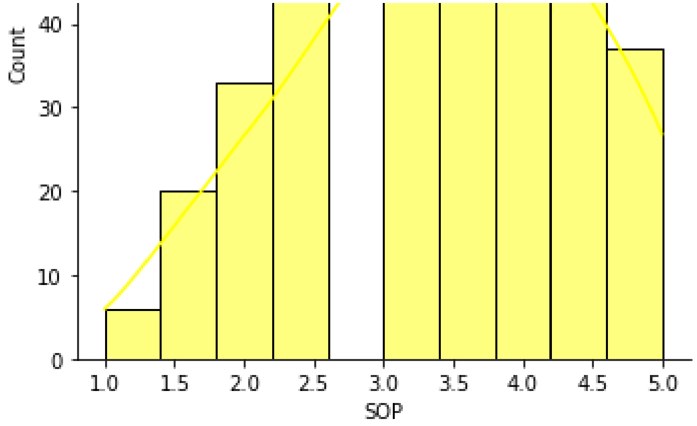
```
sns.countplot(x=data["University  Rating"]);
plt.title("Universities Star Rating by applicants");
```



In [142…

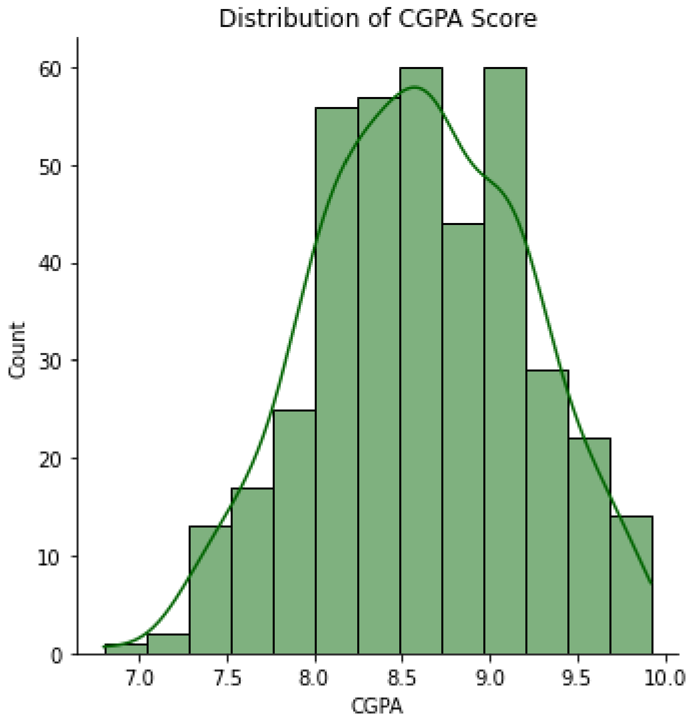
```
sns.displot(x=data["SOP"],  kde=True,  color='yellow');
plt.title("Distribution for ratings on SOP");
```





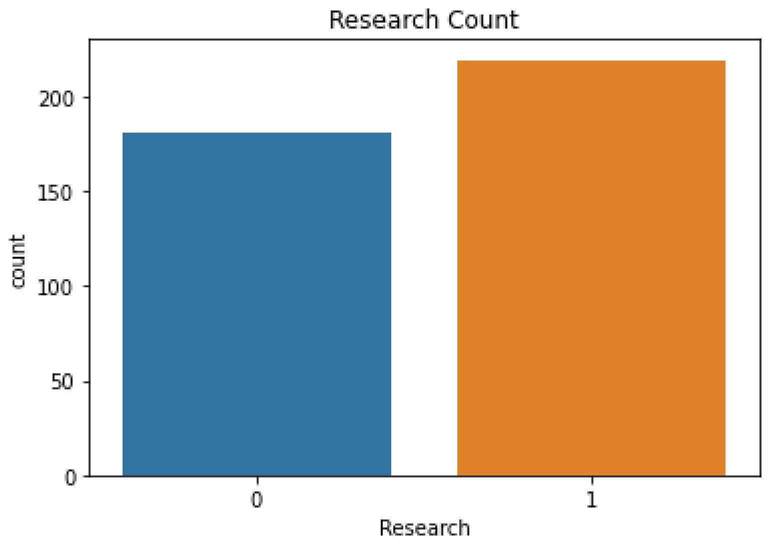
In [143…

```
sns.displot(x=data["CGPA"],  kde=True,  color='Darkgreen');
plt.title("Distribution of CGPA Score");
```



In [144…

```
sns.countplot(x=data["Research"]);
plt title("Research Count");
```



Bivariate Analysis

In [145…

```
cols = data.columns
features = [i for i in data.columns if i != 'Chance of Admit']
label = 'Chance of Admit'
features
```

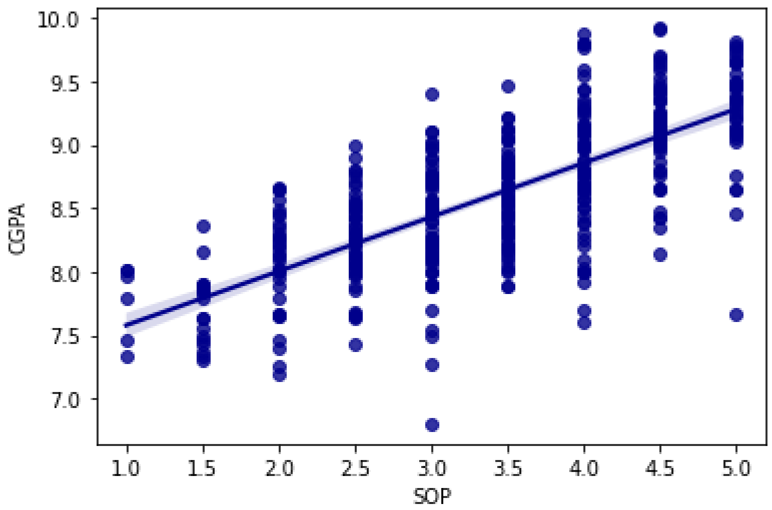
Out[145…

```
['Serial No.',
 'GRE Score',
 'TOEFL Score',
 'University Rating',
 'SOP',
 'LOR ',
 'CGPA',
 'Research',
 'Chance of Admit ']
```

In [146…

```
sns.regplot(data['SOP'],data['CGPA'],color='Darkblue')
```

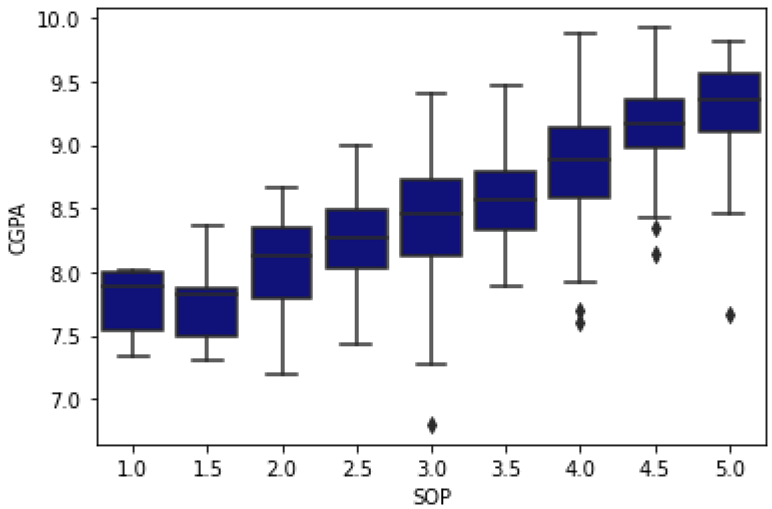
Out[146…



In [147…

```
sns.boxplot(data['SOP'],data['CGPA'],color='Darkblue')
```

Out[147…

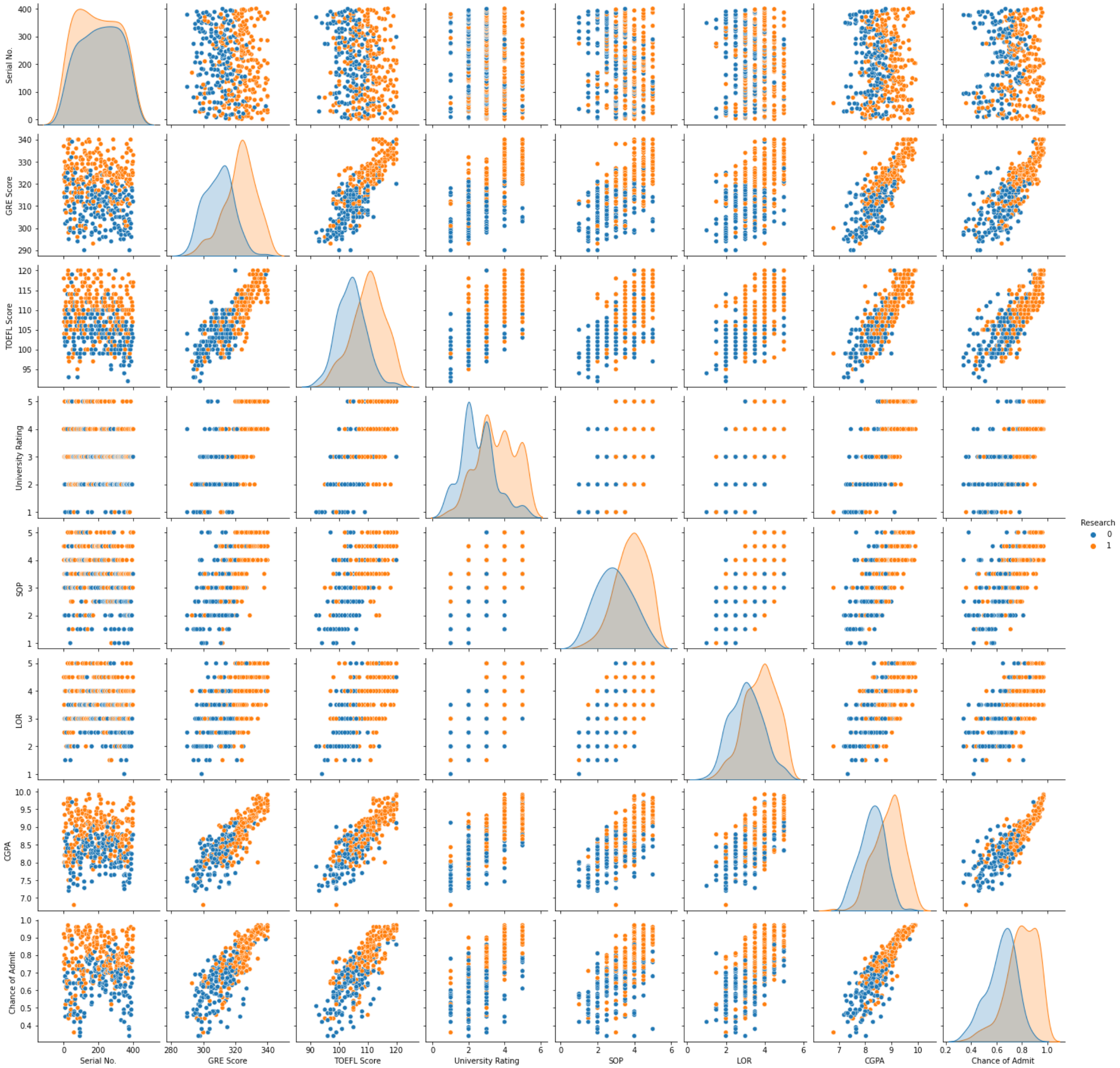


Multi-Variate Analysis

In [148…

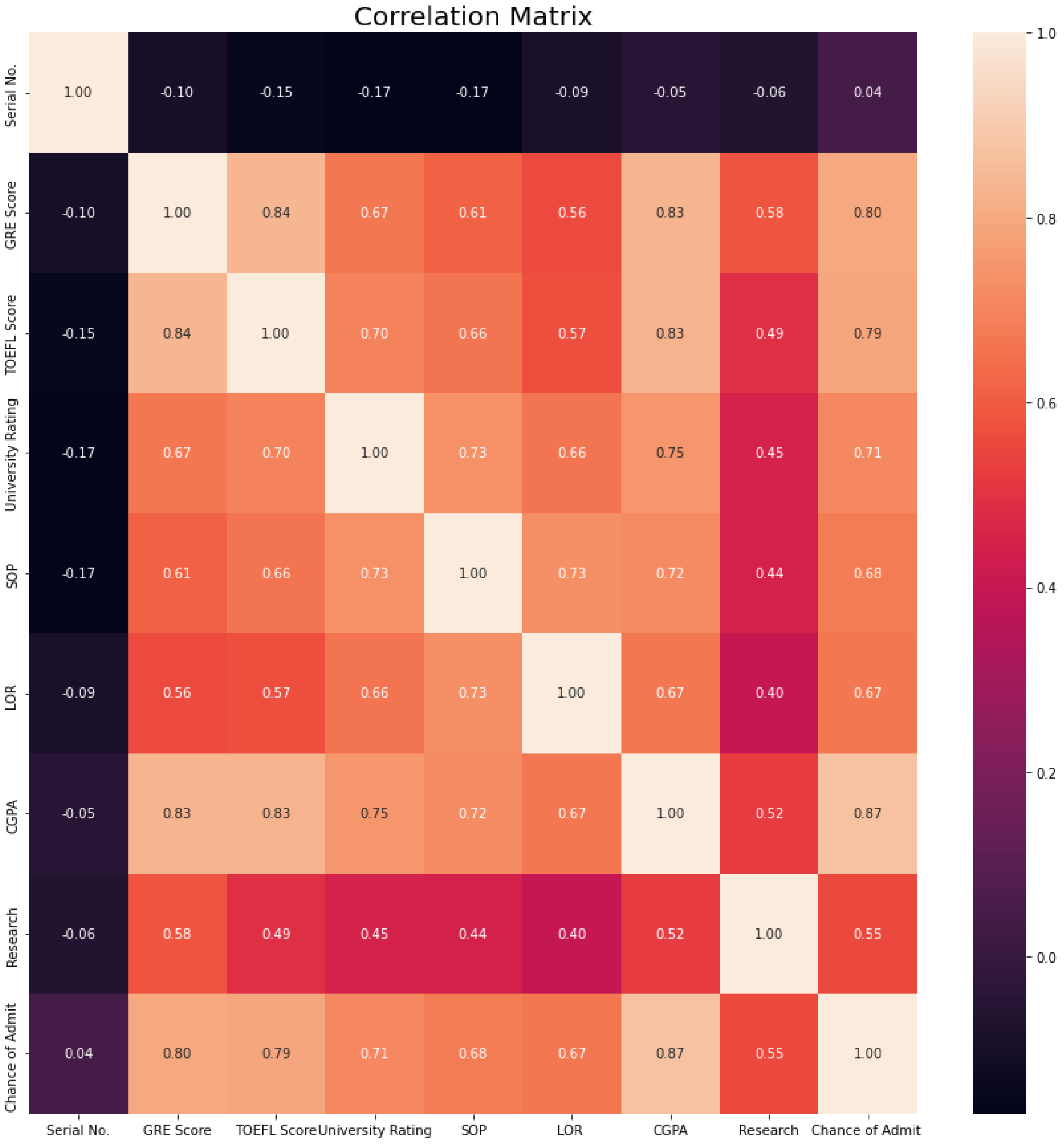
```
sns.pairplot(data,hue='Research')
```

Out[148…



In [149…

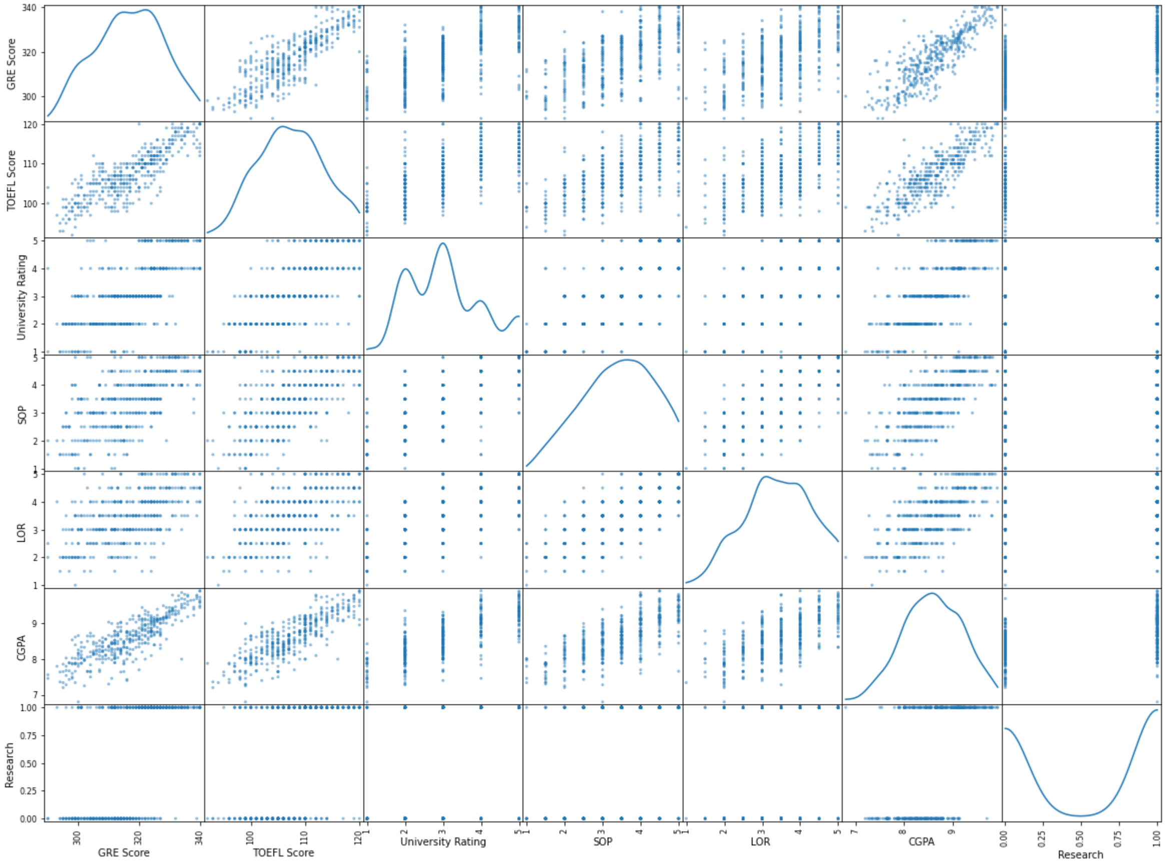
```
corr_matrix = data.corr()
plt.figure(figsize = (15, 15))
sns.heatmap(corr_matrix,annot=True,fmt='0.2f')
plt.title("Correlation Matrix", fontsize = 20)
plt.show()
```



In [150…

```
pd.plotting.scatter_matrix(data.loc[:,"GRE   Score":"Research"],diagonal="kde",fig
plt.show
```

Out[150…



Split the data into dependent and independent variables

In [151…

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

Out[151…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA
0	1	337	118	4	4 5	4 5	9 65

1	2	324	107	4	4.0	4.5	8.87
2	3	316	104	3	3.0	3.5	8.00
3	4	322	110	3	3.5	2.5	8.67
4	5	314	103	2	2.0	3.0	8.21
...	...	...	...	...	...	...	...
395	396	324	110	3	3.5	3.5	9.04
396	397	325	107	3	3.0	3.5	9.11
397	398	330	116	4	5.0	4.5	9.45
398	399	312	103	3	3.5	4.0	8.78
399	400	333	117	4	5.0	4.0	9.66

400 rows × 7 columns

In [152...

```
x.head()
```

Out[152...

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA
0	1	337	118	4	4.5	4.5	9.65
1	2	324	107	4	4.0	4.5	8.87
2	3	316	104	3	3.0	3.5	8.00
3	4	322	110	3	3.5	2.5	8.67
4	5	314	103	2	2.0	3.0	8.21

In [153...

```
y = data.iloc[:,8:9 ]
y
```

Out[153...

	Chance of Admit
0	0.92

1	0.76
2	0.72
3	0.80
4	0.65
...	...
395	0.82
396	0.84
397	0.91
398	0.67
399	0.95

400 rows × 1 columns

In [154...

```
y.head()
```

Out[154...

	Chance of Admit
0	0.92
1	0.76
2	0.72
3	0.80
4	0.65

In [155...

```
print(f'x contains: {x.shape[0]} rows and {x.shape[1]} columns')
```

```
x contains: 400 rows and 7 columns
```

In [156...

```
print(f'y contains: {y.shape}')
```

```
y contains: (400, 1)
```

In [157…

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
x[x.columns] = scaler.fit_transform(x[x.columns])
```

In [158…

```
x.head()
```

Out[158…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA
0	0.000000	0.94	0.928571	0.75	0.875	0.875	0.913462
1	0.002506	0.68	0.535714	0.75	0.750	0.875	0.663462
2	0.005013	0.52	0.428571	0.50	0.500	0.625	0.384615
3	0.007519	0.64	0.642857	0.50	0.625	0.375	0.599359
4	0.010025	0.48	0.392857	0.25	0.250	0.500	0.451923

Splitting The Data Into Train And Test

In [159…

```
X=data.drop(['Chance of Admit '],axis=1) #input data_set
y=data['Chance of Admit '] #output labels
```

In [160…

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15)
```

In [161…

```
x_train
```

Out[161…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA
5	0.012531	0.80	0.821429	1.00	0.875	0.500	0.814103
76	0.190476	0.74	0.714286	0.50	0.500	0.500	0.615385
360	0.902256	0.64	0.642857	0.50	0.750	1.000	0.589744

371	0.929825	0.68	0.642857	0.50	0.625	0.500	0.775641
204	0.511278	0.16	0.464286	0.50	0.625	0.750	0.557692
...	...	...	...	...	...	...	...
247	0.619048	0.42	0.428571	0.25	0.375	0.625	0.538462
106	0.265664	0.78	0.678571	0.75	0.875	0.875	0.762821
307	0.769424	0.70	0.714286	0.75	0.750	0.750	0.705128
156	0.390977	0.50	0.464286	0.50	0.250	0.375	0.493590
52	0.130326	0.88	0.857143	0.75	0.750	0.500	0.384615

340 rows × 7 columns

In [162…

```
y_train
```

Out[162…

```
5      0.90
76     0.74
360    0.85
371    0.89
204    0.69
       ...
247    0.71
106    0.87
307    0.80
156    0.70
52     0.78
Name: Chance of Admit , Length: 340, dtype: float64
```

In [163…

```
x_test
```

Out[163…

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA
18	0.045113	0.56	0.642857	0.50	0.750	0.500	0.641026
214	0.536341	0.82	0.892857	0.75	0.875	1.000	0.839744
386	0.967419	0.24	0.321429	0.25	0.375	0.625	0.371795

74	0.185464	0.48	0.500000	0.50	0.500	1.000	0.673077
230	0.576441	0.46	0.428571	0.50	0.750	0.875	0.592949
337	0.844612	0.84	0.928571	1.00	1.000	1.000	0.855769
1	0.002506	0.68	0.535714	0.75	0.750	0.875	0.663462
105	0.263158	0.52	0.642857	0.50	0.750	0.875	0.634615
95	0.238095	0.28	0.285714	0.75	0.125	0.375	0.333333
102	0.255639	0.48	0.500000	0.25	0.750	0.625	0.464744
183	0.458647	0.48	0.642857	0.50	0.750	0.750	0.641026
73	0.182957	0.48	0.571429	0.75	0.875	0.750	0.717949
133	0.333333	0.66	0.714286	1.00	0.750	0.875	0.634615
0	0.000000	0.94	0.928571	0.75	0.875	0.875	0.913462
227	0.568922	0.44	0.642857	0.25	0.625	0.500	0.554487
118	0.295739	0.12	0.250000	0.25	0.500	0.625	0.153846
273	0.684211	0.44	0.250000	0.00	0.000	0.125	0.387821
84	0.210526	1.00	0.821429	1.00	0.875	0.875	0.849359
82	0.205514	0.60	0.642857	1.00	1.000	0.875	0.775641
53	0.132832	0.68	0.714286	0.75	0.750	0.375	0.416667
181	0.453634	0.30	0.535714	0.25	0.375	0.375	0.519231
356	0.892231	0.74	0.607143	0.50	0.625	0.750	0.631410
8	0.020050	0.24	0.357143	0.00	0.250	0.125	0.384615
274	0.686717	0.50	0.285714	0.00	0.250	0.375	0.368590
3	0.007519	0.64	0.642857	0.50	0.625	0.375	0.599359
317	0.794486	0.20	0.250000	0.00	0.000	0.375	0.387821
338	0.847118	0.66	0.571429	1.00	0.750	0.750	0.621795
312	0.781955	0.42	0.535714	0.75	0.875	0.875	0.705128
201	0.503759	0.50	0.642857	0.25	0.625	0.500	0.532051

239	0.598997	0.18	0.285714	0.00	0.125	0.250	0.349359
111	0.278195	0.62	0.607143	0.75	0.750	0.750	0.602564
379	0.949875	0.42	0.250000	0.00	0.375	0.500	0.522436
235	0.588972	0.72	0.678571	1.00	0.875	0.750	0.778846
108	0.270677	0.82	0.857143	1.00	1.000	1.000	0.826923
43	0.107769	0.84	0.892857	0.75	0.875	0.750	0.737179
330	0.827068	0.74	0.750000	0.50	0.625	0.500	0.596154
172	0.431078	0.64	0.642857	0.75	0.750	1.000	0.746795
188	0.471178	0.82	0.821429	1.00	0.875	0.625	0.820513
373	0.934837	0.62	0.607143	0.50	0.500	0.500	0.557692
144	0.360902	0.70	0.714286	0.25	0.500	0.625	0.692308
86	0.215539	0.50	0.500000	0.50	0.875	0.625	0.519231
207	0.518797	0.40	0.357143	0.50	0.625	0.750	0.391026
110	0.275689	0.30	0.571429	1.00	0.500	0.500	0.538462
319	0.799499	0.74	0.750000	0.75	0.625	0.500	0.605769
175	0.438596	0.60	0.678571	0.75	0.875	0.625	0.663462
32	0.080201	0.96	0.928571	0.75	0.500	0.875	0.833333
225	0.563910	0.12	0.250000	0.25	0.375	0.375	0.394231
380	0.952381	0.64	0.428571	0.50	0.625	0.750	0.653846
278	0.696742	0.36	0.392857	0.25	0.500	0.625	0.541667
352	0.882206	0.26	0.285714	0.25	0.500	0.625	0.403846
252	0.631579	0.56	0.285714	0.25	0.375	0.625	0.557692
46	0.115288	0.78	0.785714	1.00	0.750	1.000	0.801282
178	0.446115	0.38	0.571429	0.50	0.375	0.500	0.423077
49	0.122807	0.74	0.678571	0.75	0.500	0.750	0.512821
246	0.616541	0.52	0.464286	0.50	0.500	0.625	0.618590

185	0.463659	0.74	0.750000	0.75	0.875	0.875	0.740385
103	0.258145	0.54	0.428571	0.25	0.875	0.750	0.535256
167	0.418546	0.46	0.357143	0.50	0.250	0.500	0.471154
80	0.200501	0.44	0.464286	0.50	0.250	0.500	0.391026
128	0.320802	0.72	0.714286	0.50	0.625	0.500	0.737179

In [164…

```
y_test
```

Out[164…

```
18     0.63
214    0.94
386    0.46
74     0.74
230    0.73
337    0.94
1      0.76
105    0.69
95     0.42
102    0.62
183    0.75
73     0.84
133    0.79
0      0.92
227    0.64
118    0.47
273    0.52
84     0.94
82     0.92
53     0.72
181    0.71
356    0.79
8      0.50
274    0.58
3      0.80
317    0.58
338    0.81
312    0.78
201    0.72
239    0.59
```

```
111    0.69
379    0.71
235    0.88
108    0.93
43     0.87
330    0.80
172    0.86
188    0.93
373    0.79
144    0.80
86     0.72
207    0.66
110    0.61
319    0.80
175    0.85
32     0.91
225    0.61
380    0.78
278    0.66
352    0.64
252    0.71
46     0.86
178    0.72
49     0.78
246    0.72
185    0.89
103    0.57
167    0.64
80     0.50
128    0.84
Name: Chance of Admit , dtype: float64
```

MODELING AND TRAINING

In [165…

```
from sklearn.ensemble import GradientBoostingRegressor
model = GradientBoostingRegressor()
model.fit(x_train,y_train)
```

Out[165…

```
GradientBoostingRegressor()
```

In [166…

```
model.score(x_test,y_test)
```

Out[166…

```
0.8629183215319991
```

In [167…

```
y_predict=model.predict(x_test)
```

In [168…

```
from sklearn.metrics import mean_squared_error, r2_score,mean_absolute_error
import numpy as np
print('Mean Absolute Error:', mean_absolute_error(y_test, y_predict))
print('Mean Squared Error:', mean_squared_error(y_test, y_predict))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_predict)))
```

```
Mean Absolute Error: 0.03666253362648755
Mean Squared Error: 0.0024410096406552915
Root Mean Squared Error: 0.04940657487273623
```

In [169…

```
y_train = (y_train>0.5)
y_test = (y_test>0.5)
```

In [170…

```
from sklearn.linear_model._logistic import LogisticRegression

lore = LogisticRegression(random_state=0, max_iter=1000)

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

In [171…

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

In [172…

```
from sklearn.metrics import accuracy_score, recall_score, roc_auc_score, confusi

print('Accuracy Score:', accuracy_score(y_test, y_pred))
print('Recall Score:', recall_score(y_test, y_pred))
print('ROC AUC Score:', roc_auc_score(y_test, y_pred))
print('Confussion Matrix:\n', confusion_matrix(y_test, y_pred))
```

```
Accuracy Score: 0.9166666666666666
Recall Score: 1.0
```

```
ROC AUC Score: 0.5
Confussion Matrix:
 [[ 0  5]
 [ 0 55]]
```

SAVING THE MODEL

In [173…

```
import pickle
```

In [174…

```
pickle.dump(lr, open("university.pkl", "wb")) #logistic regression model
```

HOSTING THE MODEL

In [193…

```
import pickle
```

In [194…

```
lr = pickle.load(open("university.pkl", "rb")) #logistic regression model
```

In [176…

```
!pip install -U ibm-watson-machine-learning
```

```
Requirement already satisfied: ibm-watson-machine-learning in /opt/conda/envs/Pyt
hon-3.9/lib/python3.9/site-packages (1.0.256)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/
lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/l
ib/python3.9/site-packages (from ibm-watson-machine-learning) (4.8.2)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.
9/site-packages (from ibm-watson-machine-learning) (1.26.7)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.
9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.3.4)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python
3.9/site-packages (from ibm-watson-machine-learning) (21.3)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.
9/site-packages (from ibm-watson-machine-learning) (0.3.3)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python
```

```
3.9/site-packages (from ibm-watson-machine-learning) (0.8.9)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python
3.9/site-packages (from ibm-watson-machine-learning) (2.26.0)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.
9/site-packages (from ibm-watson-machine-learning) (2022.9.24)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/
Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-mach
ine-learning) (2.11.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-lea
rning) (0.10.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python
-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-le
arning) (2.11.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Pyt
hon-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==
2.11.*->ibm-watson-machine-learning) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/pyt
hon3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning) (2
021.3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/py
thon3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning)
(1.20.3)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python
3.9/site-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.11.0->ibm-
cos-sdk==2.11.*->ibm-watson-machine-learning) (1.15.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Pytho
n-3.9/lib/python3.9/site-packages (from requests->ibm-watson-machine-learning)
(2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/pyt
hon3.9/site-packages (from requests->ibm-watson-machine-learning) (3.3)
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python
3.9/site-packages (from importlib-metadata->ibm-watson-machine-learning) (3.6.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python
-3.9/lib/python3.9/site-packages (from packaging->ibm-watson-machine-learning)
(3.0.4)
```

In [177…

```
from ibm_watson_machine_learning import APIClient
import json
```

Authenticate and set space

In [178…

```
wml_credentials = {
    "apikey":"d2-3twnS_oRmh5yFAxs0aK7kpqJcN3_1qQy6IX3-P413",
    "url":"https://us-south.ml.cloud.ibm.com"
}
```

In [179…

```
wml_client  =  APIClient(wml_credentials)
```

In [180…

```
wml_client.spaces.list()
```

```
Note: 'limit' is not provided. Only first 50 records will be displayed if the num
ber of records exceed 50
------------------------------------  ------------------------------  ----------
------------------------------------
ID                                    NAME                            CREATED
77330267-b27a-47ac-8c71-e7808afcd79e  UNIVERSITY_ADMISSION_PREDICTION  2022-11-01
T13:20:22.473Z
------------------------------------  ------------------------------  ----------
--------------
```

In [181…

```
SPACE_ID="77330267-b27a-47ac-8c71-e7808afcd79e"
```

In [182…

```
wml_client.set.default_space(SPACE_ID)
```

Out[182…

```
'SUCCESS'
```

In [183…

```
wml_client.software_specifications.list()
```

```
-----------------------------  ------------------------------------  ----
NAME                           ASSET_ID                              TYPE
default_py3.6                  0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12      020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt     069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6        09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12     09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9      0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6          0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
```

```
shiny-r3.6                     0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod   1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6              10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl      111e41b3-de2d-5422-a4d6-bf776828c4b7  base
runtime-22.1-py3.9             12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6        154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                   1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6         1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6           1c9e5454-f216-59dd-a20e-474a5cdf5988  base
pytorch-onnx_rt22.1-py3.9-edt  1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6           1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod   217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda        26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                       295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8            2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6          2b73a275-7cbf-420b-a912-eae7f436e0bc  base
kernel-spark3.3-py3.9          2b7961e2-e3b1-5a8c-a491-482c8368839a  base
pytorch_1.2-py3.6              2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt     32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37           36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                390d21f8-e58b-4fac-9c55-d7ceda621326  base
xgboost_0.82-py3.6             39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt     40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
default_r36py38                41c247d3-45f8-5a71-b065-8580229facf0  base
autoai-ts_rt22.1-py3.9         4269d26e-07ba-5d40-8f66-2d495b0c71f7  base
autoai-obm_3.0                 42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
pmml-3.0_4.3                   493bcb95-16f1-5bc5-bee8-81b8af80e9c7  base
spark-mllib_2.4-r_3.6          49403dff-92e9-4c87-a3d7-a42d0021c095  base
xgboost_0.90-py3.6             4ff8d6c2-1343-4c18-85e1-689c965304d3  base
pytorch-onnx_1.1-py3.6         50f95b2a-bc16-43bb-bc94-b0bed208c60b  base
autoai-ts_3.9-py3.8            52c57136-80fa-572e-8728-a5e7cbb42cde  base
spark-mllib_2.4-scala_2.11     55a70f99-7320-4be5-9fb9-9edb5a443af5  base
spark-mllib_3.0                5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9  base
autoai-obm_2.0                 5c2e37fa-80b8-5e77-840f-d912469614ee  base
spss-modeler_18.1              5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b  base
cuda-py3.8                     5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e  base
autoai-kb_3.1-py3.7            632d4b22-10aa-5180-88f0-f52dfb6444d7  base
pytorch-onnx_1.7-py3.8         634d3cdc-b562-5bf9-a2d4-ea90a478456b  base
spark-mllib_2.3-r_3.6          6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c  base
tensorflow_2.4-py3.7           65e171d7-72d1-55d9-8ebb-f813d620c9bb  base
spss-modeler_18.2              687eddc9-028a-4117-b9dd-e57b36f1efa5  base
-----------------------------  ------------------------------------  ----
```

```
Note: Only first 50 records were displayed. To display more use 'limit' paramete
r.
```

Save and Deploy the model

In [184…

```
import sklearn
sklearn.__version__
```

Out[184…

```
'1.0.2'
```

In [185…

```
MODEL_NAME = 'UNIVERSITY_ADMISSION_PREDICTION'
DEPLOYMENT_NAME = 'UNIVERSITY_ADMISSION_PREDICTION'
DEMO_MODEL = model
```

In [186…

```
# Set Python Version
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-2
```

In [187…

```
# Setup model meta
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

In [188…

```
#Save model
model_details = wml_client.repository.store_model(
    model=DEMO_MODEL,
    meta_props=model_props,
    training_data=x_train,
    training_target=y_train
)
```

In [189…

```
model_details
```

Out[189

```
{'entity': {'hybrid pipeline software specs': [],
  'label_column': 'Chance of Admit ',
  'schemas': {'input': [{'fields': [{'name': 'Serial No.', 'type': 'float64'},
      {'name': 'GRE Score', 'type': 'float64'},
      {'name': 'TOEFL Score', 'type': 'float64'},
      {'name': 'University Rating', 'type': 'float64'},
      {'name': 'SOP', 'type': 'float64'},
      {'name': 'LOR ', 'type': 'float64'},
      {'name': 'CGPA', 'type': 'float64'}],
     'id': '1',
     'type': 'struct'}],
   'output': []},
  'software_spec': {'id': '12b83a17-24d8-5082-900f-0ab31fbfd3cb',
   'name': 'runtime-22.1-py3.9'},
  'type': 'scikit-learn_1.0'},
 'metadata': {'created_at': '2022-11-03T12:56:02.525Z',
  'id': '8c39cf12-4f7f-4e2f-be42-561ef2534cb1',
  'modified_at': '2022-11-03T12:56:06.118Z',
  'name': 'UNIVERSITY_ADMISSION_PREDICTION',
  'owner': 'IBMid-667000C4DA',
  'resource_key': 'cacc40ce-7250-45b4-9780-7a384b44ca2f',
  'space_id': '77330267-b27a-47ac-8c71-e7808afcd79e'},
 'system': {'warnings': []}}
```

In [190…

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

Out[190…

```
'8c39cf12-4f7f-4e2f-be42-561ef2534cb1'
```

In [191…

```
# Set meta
deployment_props = {
    wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE: {}
}
```

In [192…

```
# Deploy
deployment = wml_client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)
```

```
################################################################################
######

Synchronous deployment creation for uid: '8c39cf12-4f7f-4e2f-be42-561ef2534cb1' s
tarted

################################################################################
######


initializing
Note: online_url is deprecated and will be removed in a future release. Use servi
ng_urls instead.
.
ready


----------------------------------------------------------------------------------
---------------
Successfully finished deployment creation, deployment_uid='523f7b82-a7a0-4066-afa
b-c0f74dc8b976'
----------------------------------------------------------------------------------
---------------
```

In [ ]: