Importing the required libraries

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

Load the Dataset

```
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 includes your credentials.
# You might want to remove those credentials before you share the
notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='CkS2-uN0UofbsGnqQSz-I3ssQ8fUHOT2gSLJuMQrJ9Vc',
    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 = 'universityadmiteligibilitypredict-donotdelete-pr-
i7u8fj33wgfayc'
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__, body )

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

```
   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

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

Analyse the Data

```
df.head()
```

```
   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

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

```
df.tail()
```

```
     Serial No.  GRE Score  TOEFL Score  University Rating  SOP  LOR
CGPA  \
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

     Research  Chance of Admit
395         1              0.82
396         1              0.84
397         1              0.91
398         0              0.67
399         1              0.95
```

Drop the serial No column

```
df.drop("Serial No.",axis=1,inplace=True)
```

```
df.head()
```

```
   GRE Score  TOEFL Score  University Rating  SOP  LOR   CGPA
Research  \
0        337          118                  4  4.5   4.5  9.65
1
1        324          107                  4  4.0   4.5  8.87
1
2        316          104                  3  3.0   3.5  8.00
1
3        322          110                  3  3.5   2.5  8.67
1
4        314          103                  2  2.0   3.0  8.21
0

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

Checking for null value

```
df.isnull().sum()
```

```
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
```

```
Chance of Admit      0
dtype: int64
```

getting information about the dataframe

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   GRE Score          400 non-null    int64  
 1   TOEFL Score        400 non-null    int64  
 2   University Rating  400 non-null    int64  
 3   SOP                400 non-null    float64
 4   LOR                400 non-null    float64
 5   CGPA               400 non-null    float64
 6   Research           400 non-null    int64  
 7   Chance of Admit    400 non-null    float64
dtypes: float64(4), int64(4)
memory usage: 25.1 KB
```

statistical summary of the dataframe

```
df.describe()
```

```
        GRE Score  TOEFL Score  University Rating         SOP
LOR   \
count  400.000000   400.000000         400.000000  400.000000
400.000000
mean   316.807500   107.410000           3.087500    3.400000
3.452500
std     11.473646     6.069514           1.143728    1.006869
0.898478
min    290.000000    92.000000           1.000000    1.000000
1.000000
25%    308.000000   103.000000           2.000000    2.500000
3.000000
50%    317.000000   107.000000           3.000000    3.500000
3.500000
75%    325.000000   112.000000           4.000000    4.000000
4.000000
max    340.000000   120.000000           5.000000    5.000000
5.000000

             CGPA    Research  Chance of Admit
count  400.000000  400.000000        400.000000
mean     8.598925    0.547500          0.724350
std      0.596317    0.498362          0.142609
```

```
min      6.800000    0.000000          0.340000
25%      8.170000    0.000000          0.640000
50%      8.610000    1.000000          0.730000
75%      9.062500    1.000000          0.830000
max      9.920000    1.000000          0.970000
```

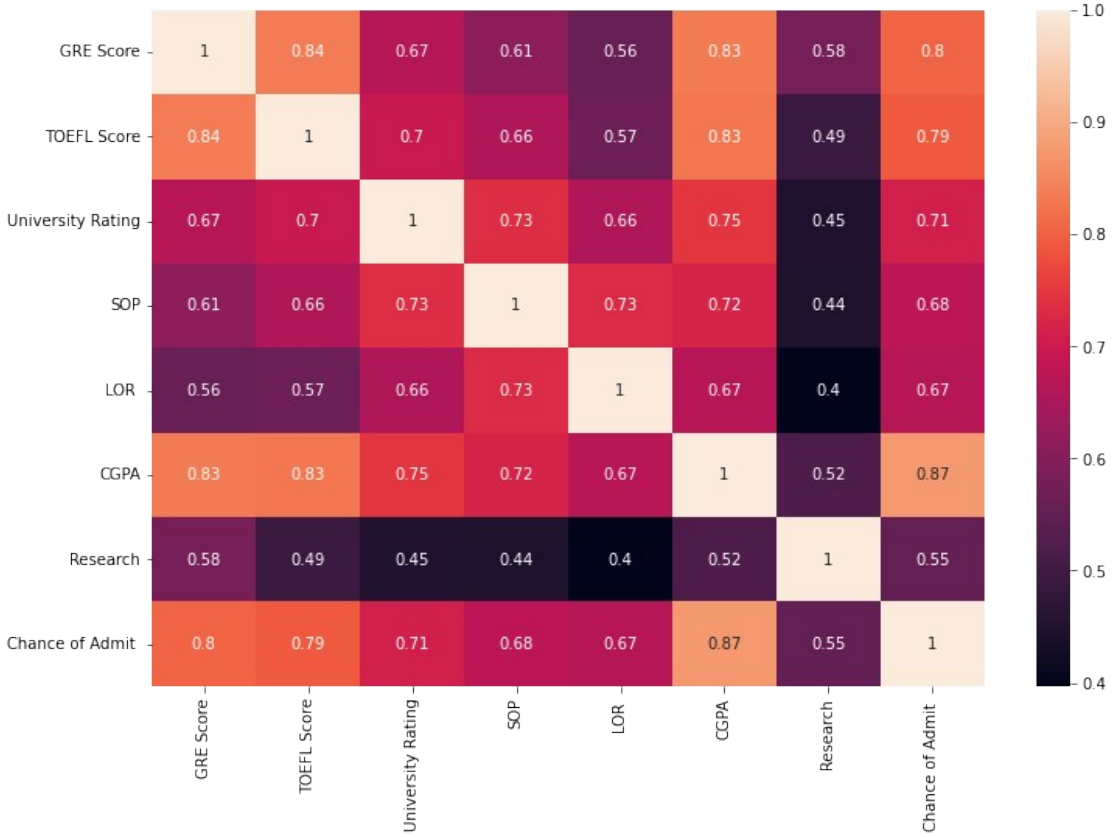
To find the correlation of columns

```
corr_matrix=df.corr()
```

```
corr_matrix
```

```
                   GRE Score  TOEFL Score  University Rating       SOP
\
GRE Score           1.000000     0.835977           0.668976  0.612831

TOEFL Score         0.835977     1.000000           0.695590  0.657981

University Rating   0.668976     0.695590           1.000000  0.734523

SOP                 0.612831     0.657981           0.734523  1.000000

LOR                 0.557555     0.567721           0.660123  0.729593

CGPA                0.833060     0.828417           0.746479  0.718144

Research            0.580391     0.489858           0.447783  0.444029

Chance of Admit     0.802610     0.791594           0.711250  0.675732


                        LOR       CGPA  Research  Chance of Admit
GRE Score          0.557555  0.833060  0.580391          0.802610
TOEFL Score        0.567721  0.828417  0.489858          0.791594
University Rating  0.660123  0.746479  0.447783          0.711250
SOP                0.729593  0.718144  0.444029          0.675732
LOR                1.000000  0.670211  0.396859          0.669889
CGPA               0.670211  1.000000  0.521654          0.873289
Research           0.396859  0.521654  1.000000          0.553202
Chance of Admit    0.669889  0.873289  0.553202          1.000000
```

```
#plotting the correlation matrix as a heatmap
fig = plt.figure(figsize=(12,8))
sns.heatmap(corr_matrix,annot=True)
plt.show()
```

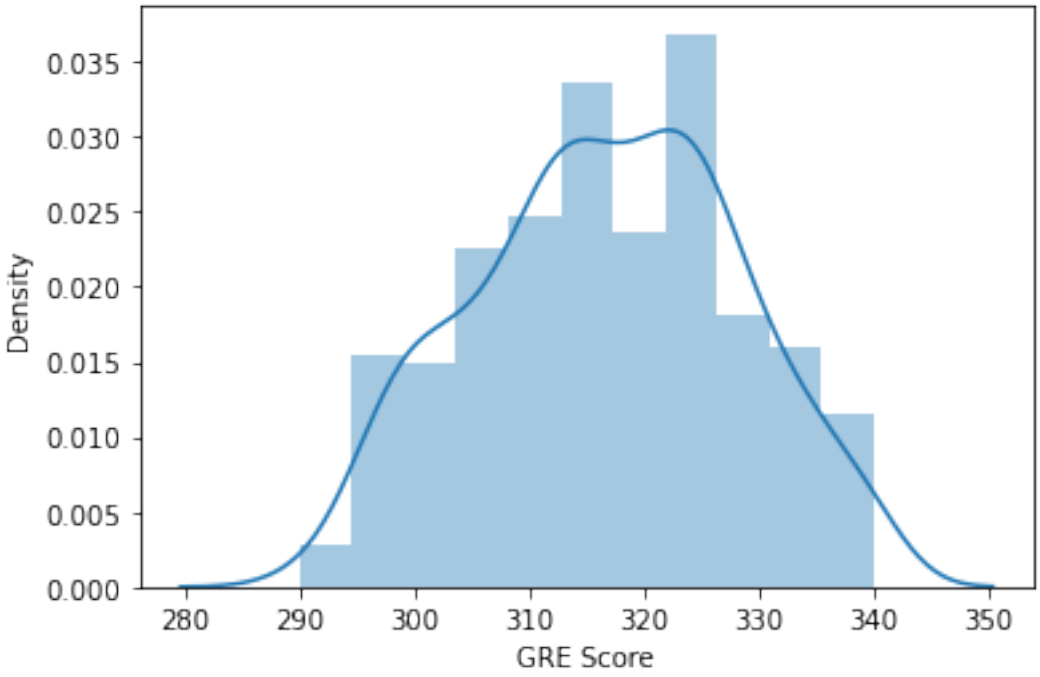


Data Visualization

Univarient Analysis

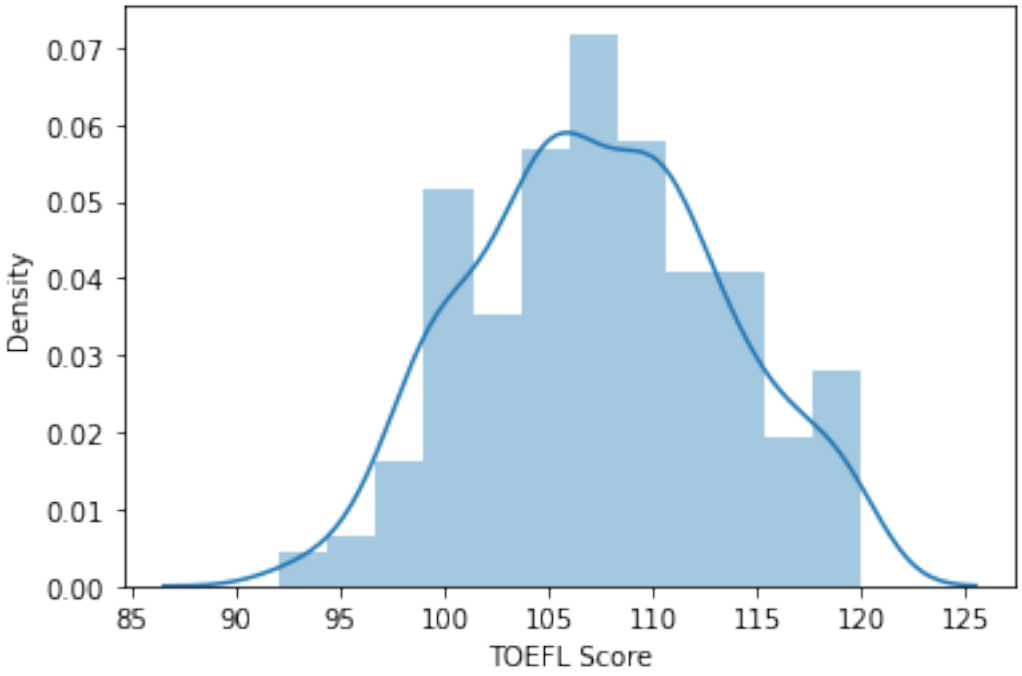
```
sns.distplot(df["GRE Score"])
```

```
<AxesSubplot:xlabel='GRE Score', ylabel='Density'>
```



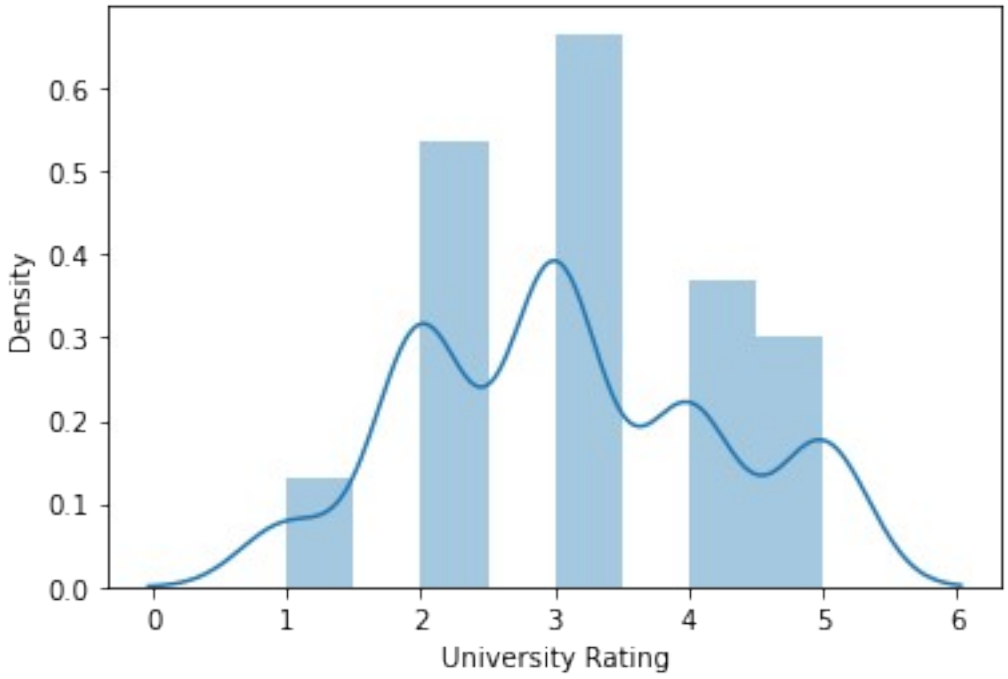

```
sns.distplot(df["TOEFL Score"])
```

```
<AxesSubplot:xlabel='TOEFL Score', ylabel='Density'>
```



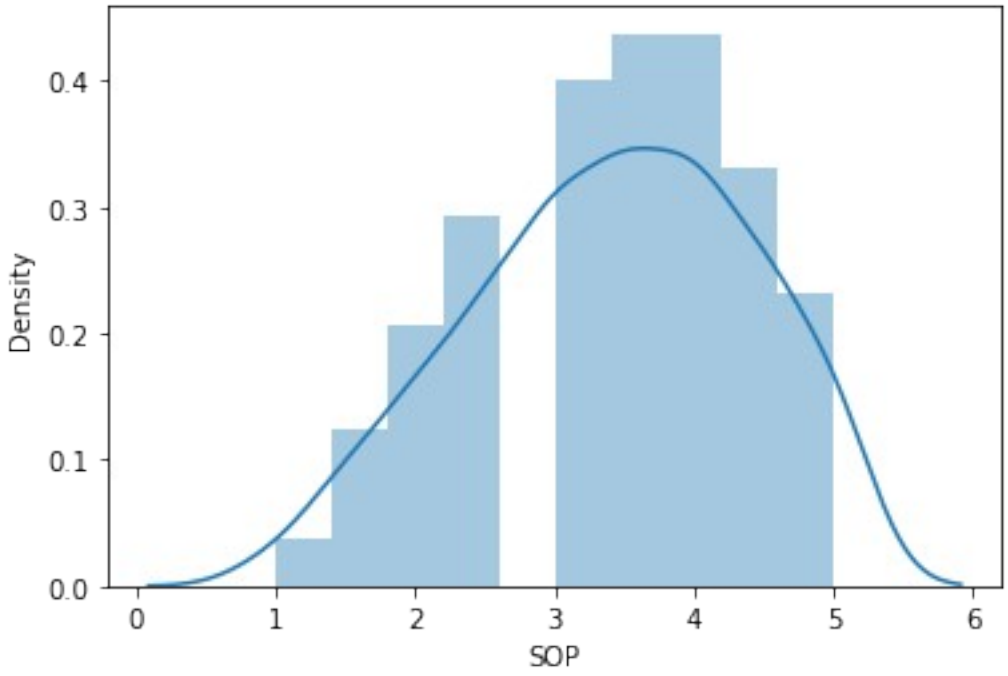

```
sns.distplot(df["University Rating"])
```

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



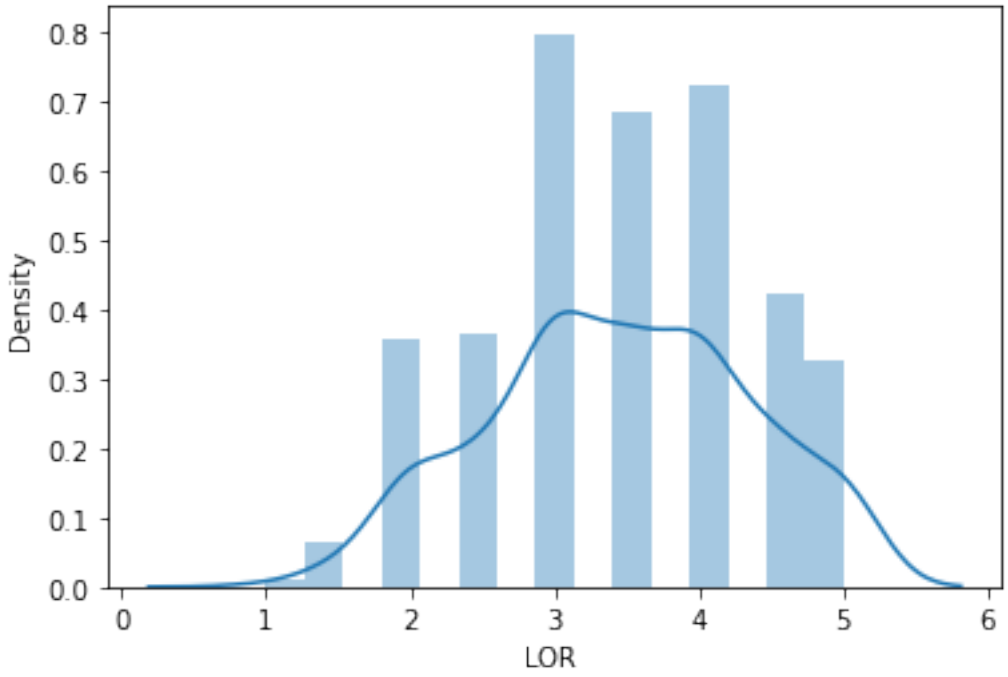

```
sns.distplot(df["SOP"])
```

```
<AxesSubplot:xlabel='SOP', ylabel='Density'>
```



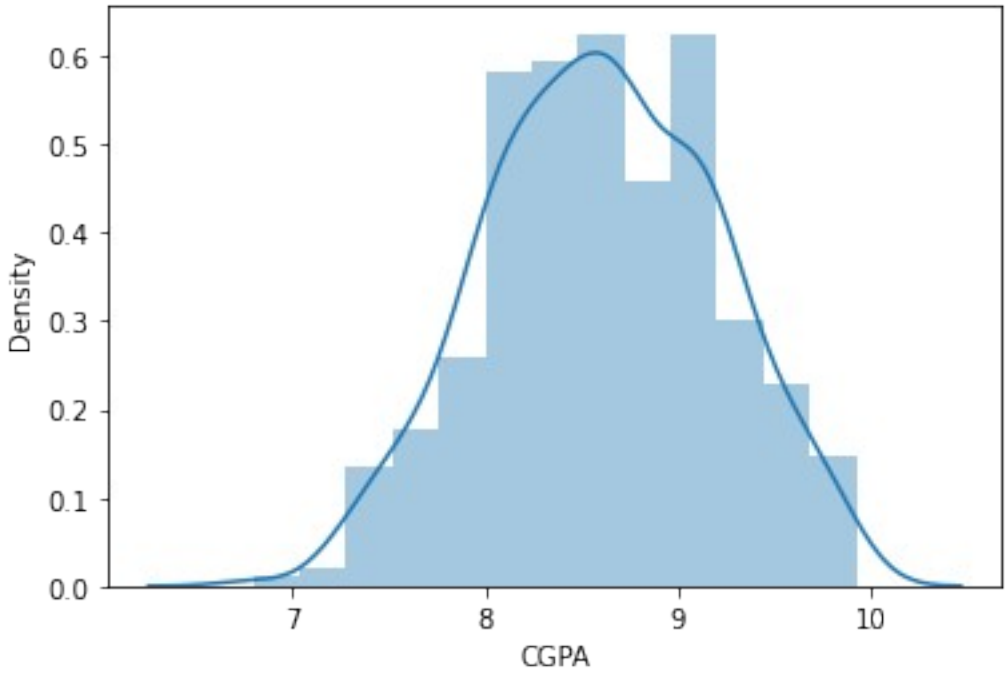

```
sns.distplot(df["LOR "])
```

```
<AxesSubplot:xlabel='LOR ', ylabel='Density'>
```



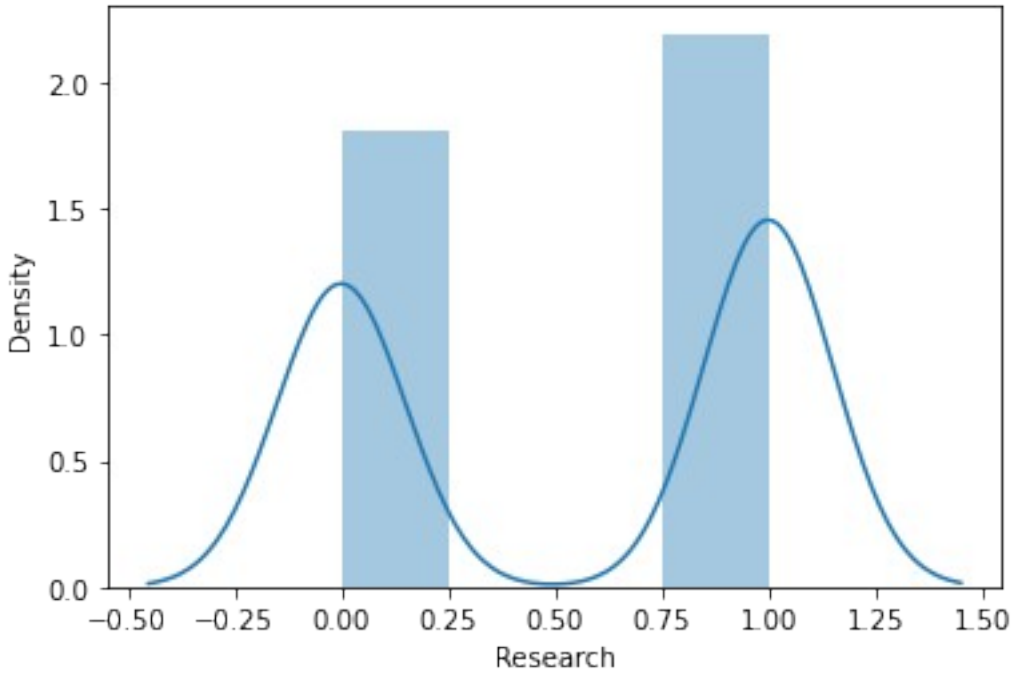

```
sns.distplot(df["CGPA"])
```

```
<AxesSubplot:xlabel='CGPA', ylabel='Density'>
```



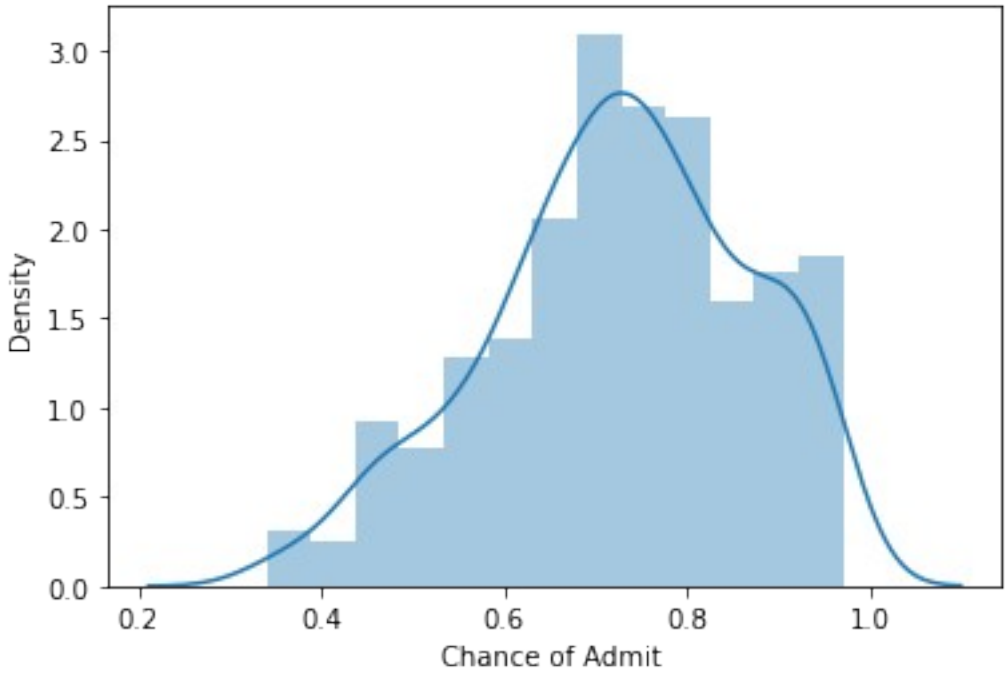

```
sns.distplot(df["Research"])
```

```
<AxesSubplot:xlabel='Research', ylabel='Density'>
```




```
sns.distplot(df["Chance of Admit "])
```

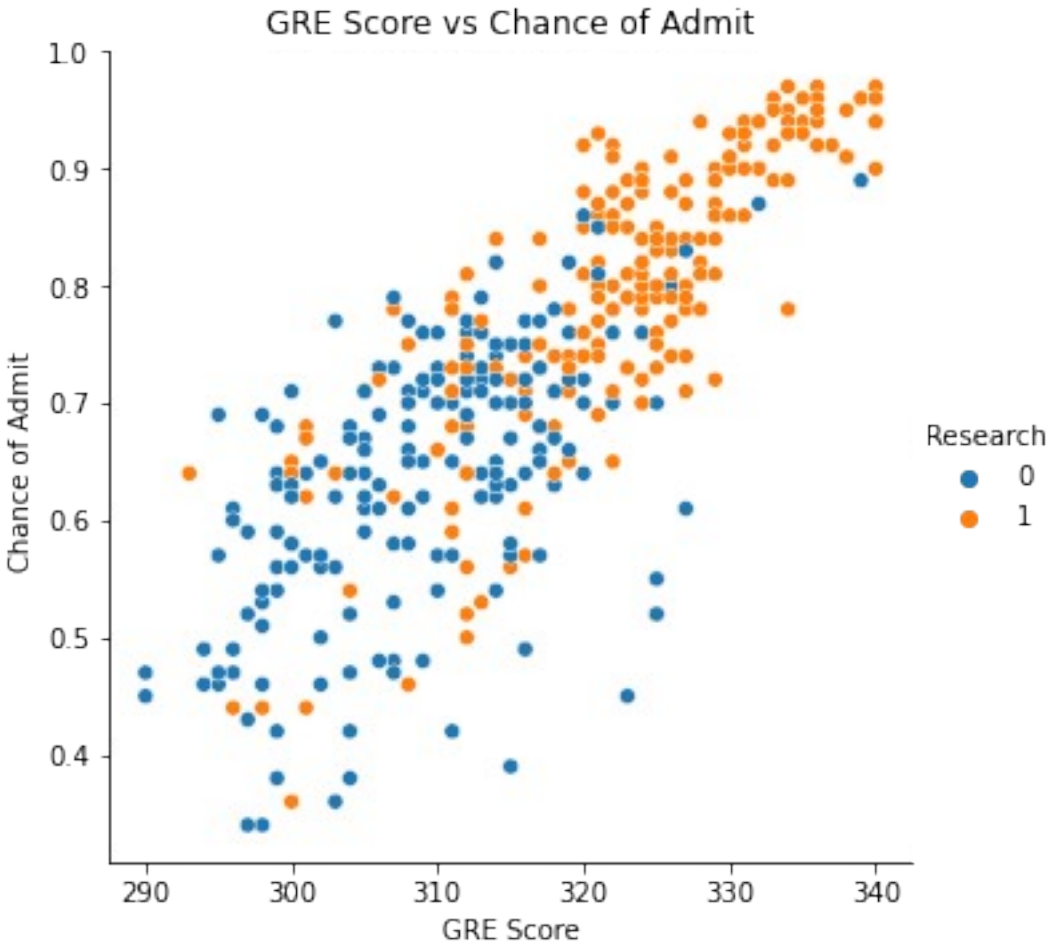
```
<AxesSubplot:xlabel='Chance of Admit ', ylabel='Density'>
```



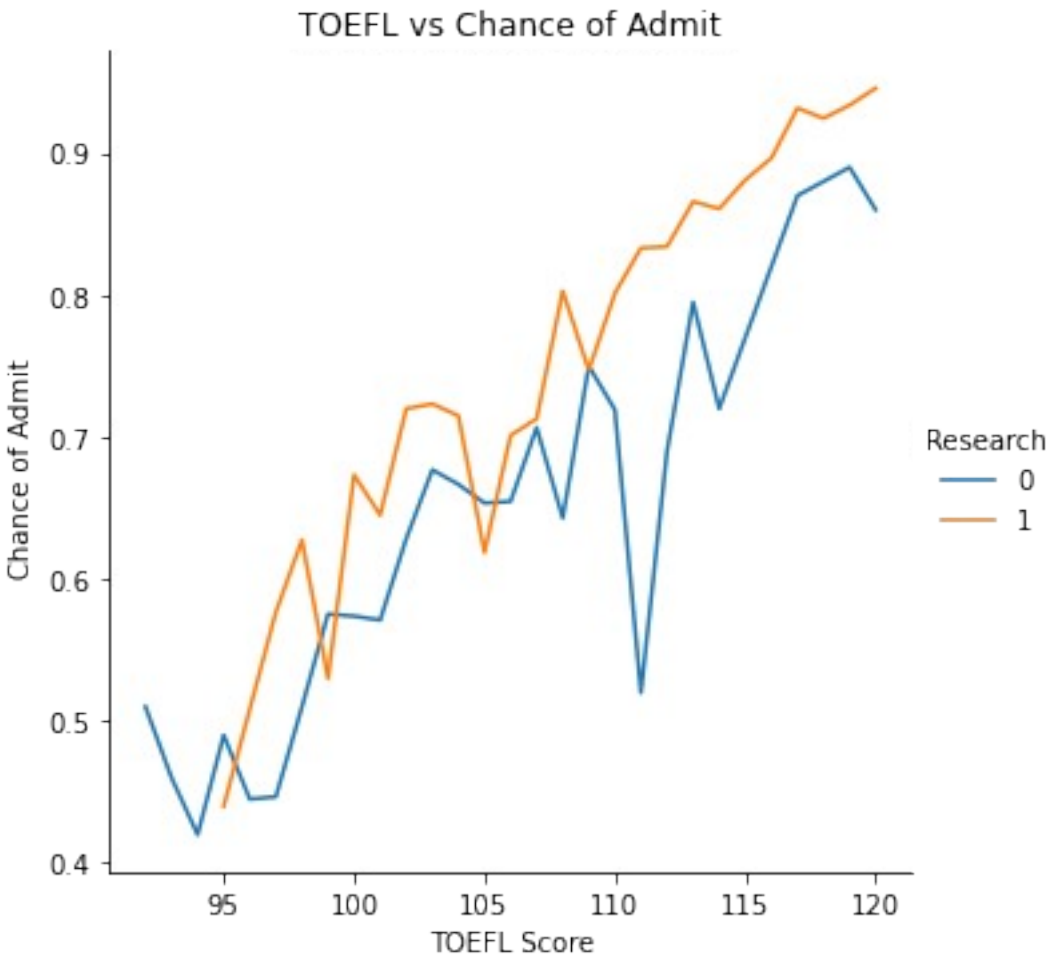
Bivarient Analysis

plotting data which have high correlation

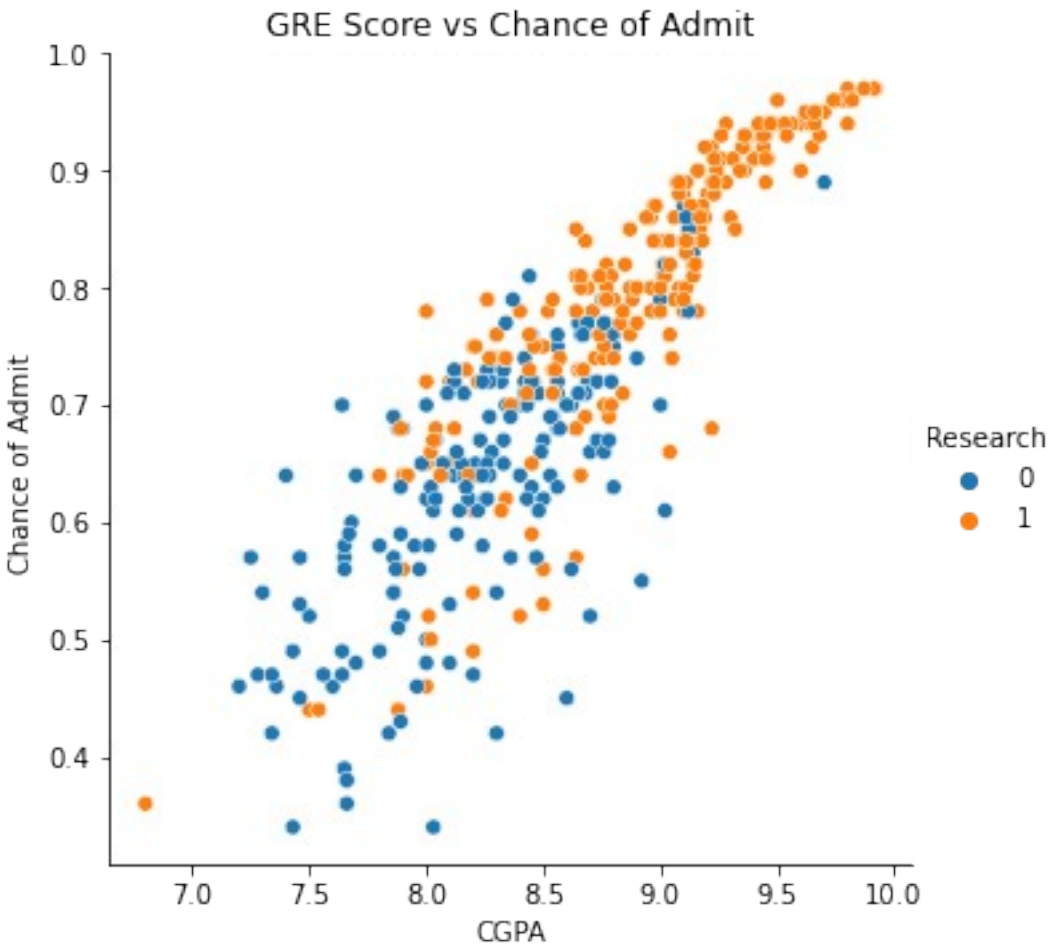
```
sns.relplot(data=df,x="GRE Score",y="Chance of Admit ",hue="Research")
plt.title("GRE Score vs Chance of Admit")
plt.show()
```



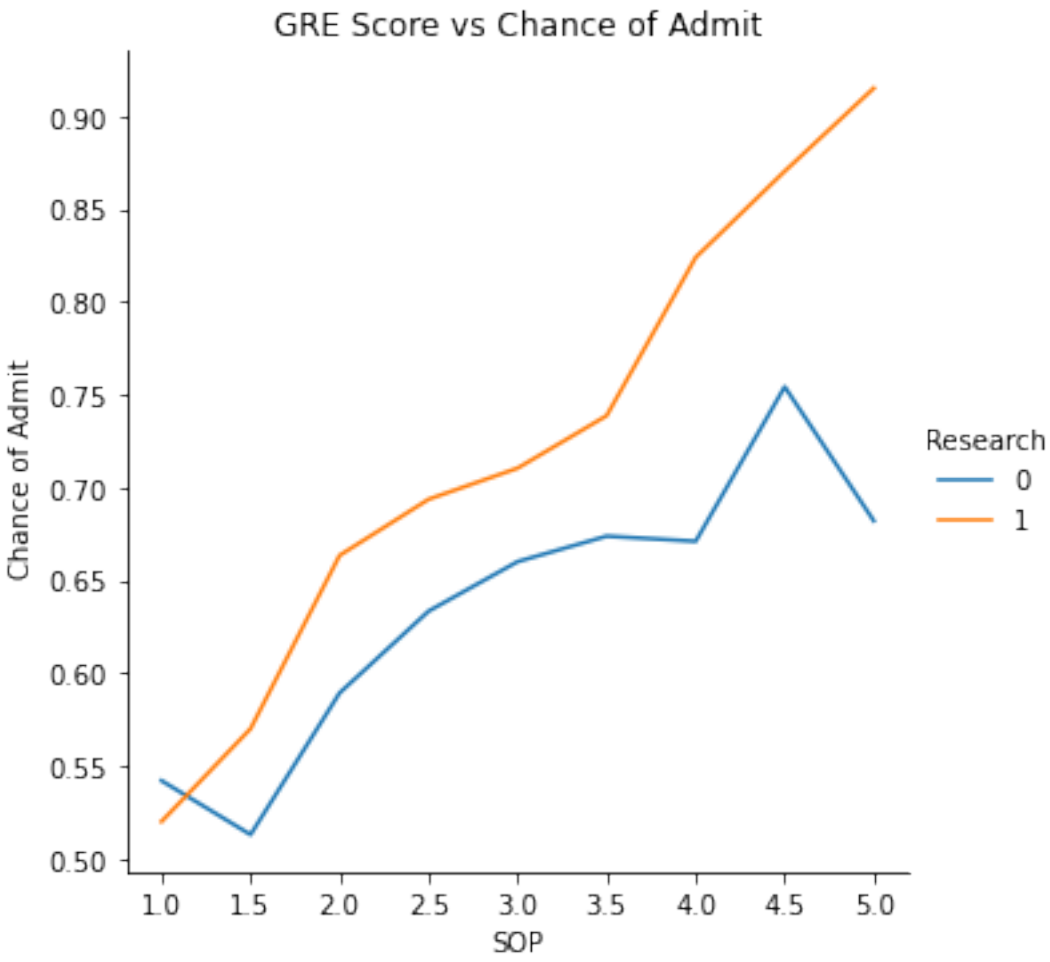

```
sns.relplot(data=df,x="TOEFL Score",y="Chance of Admit
",hue="Research",kind="line",ci=None)
plt.title("TOEFL vs Chance of Admit")
plt.show()
```



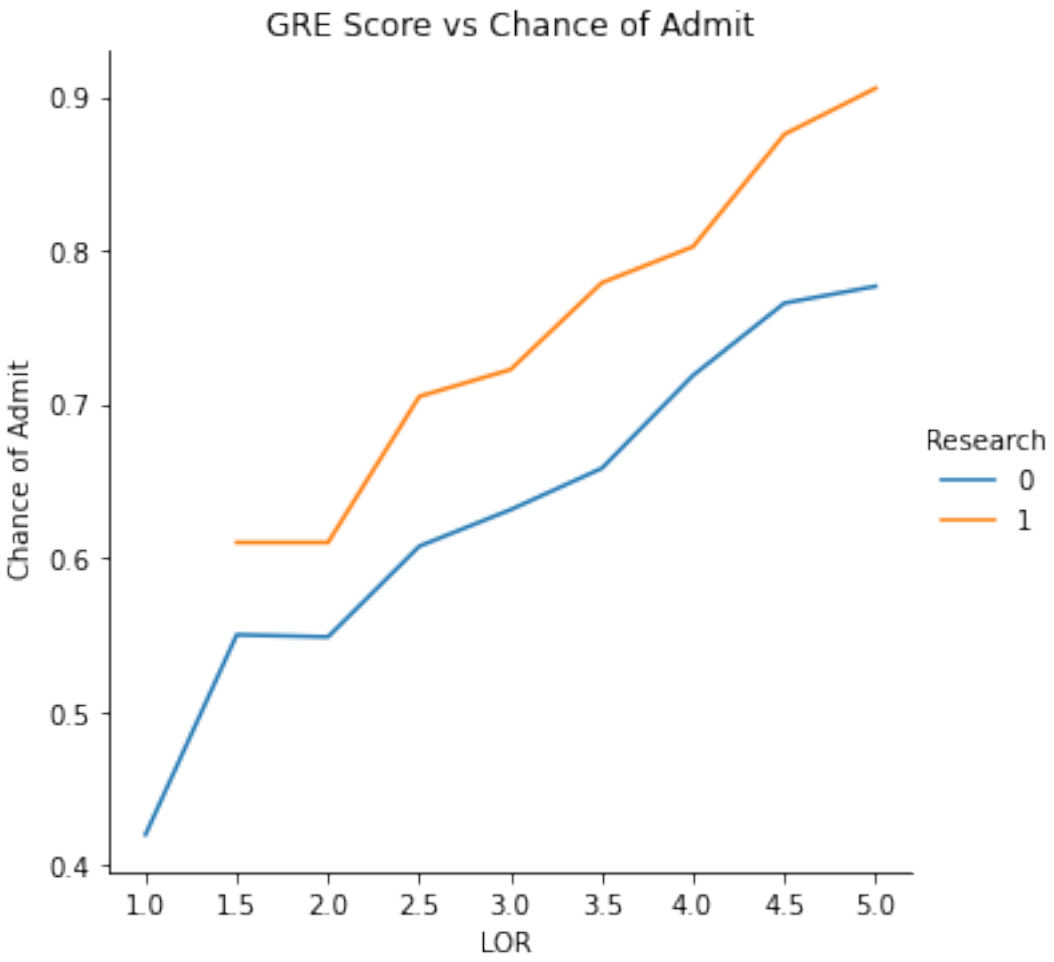

```
sns.relplot(data=df,x="CGPA",y="Chance of Admit ",hue="Research")
plt.title("GRE Score vs Chance of Admit")
plt.show()
```



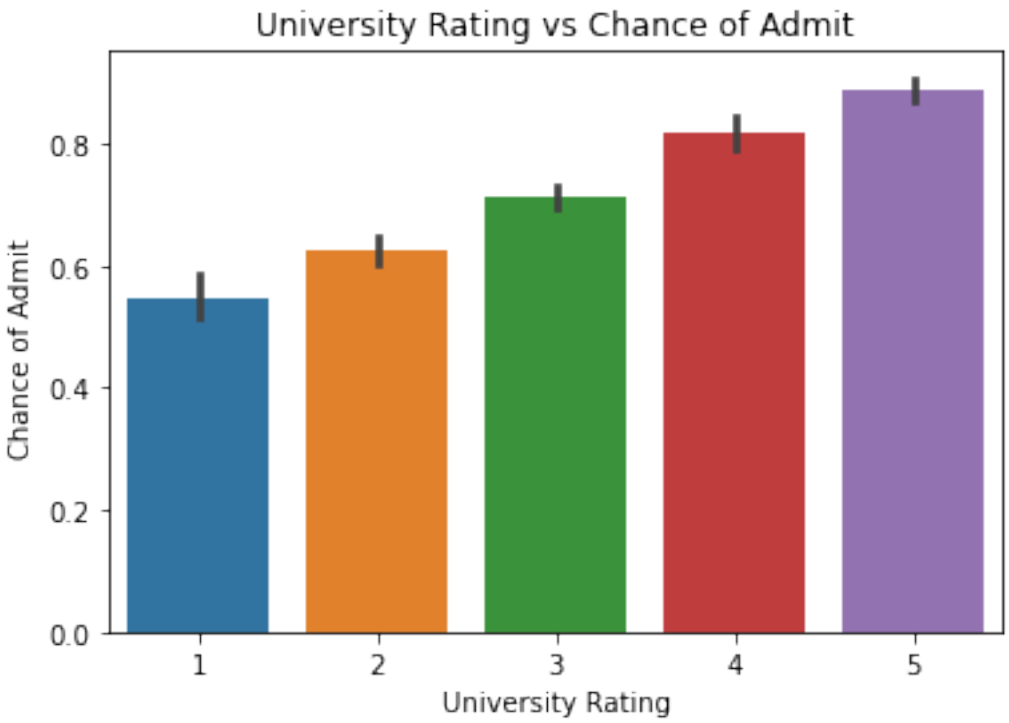

```
sns.relplot(data=df,x="SOP",y="Chance of Admit
",hue="Research",kind="line",ci=None)
plt.title("GRE Score vs Chance of Admit")
plt.show()
```




```
sns.relplot(data=df,x="LOR ",y="Chance of Admit
",hue="Research",kind="line",ci=None)
plt.title("GRE Score vs Chance of Admit")
plt.show()
```

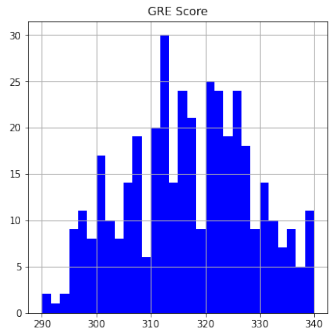


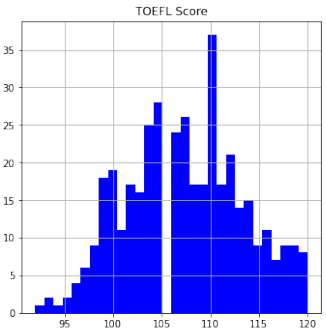

```
sns.barplot(data=df,x="University Rating",y="Chance of Admit ")
plt.title("University Rating vs Chance of Admit")
plt.show()
```

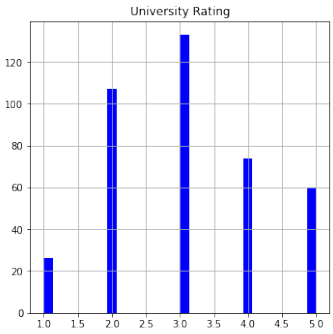


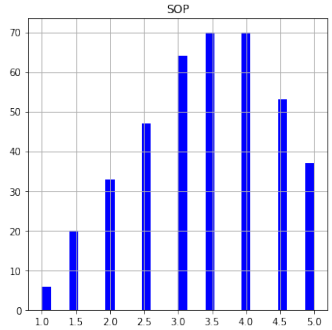

```
df.hist(bins = 30, figsize = (20,20), color = 'blue')
```

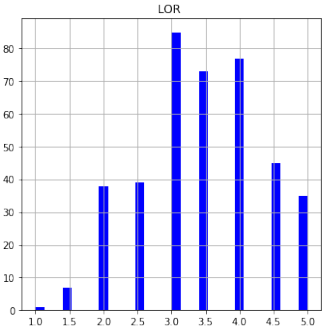
```
array([[<AxesSubplot:title={'center':'GRE Score'}>,
        <AxesSubplot:title={'center':'TOEFL Score'}>,
        <AxesSubplot:title={'center':'University Rating'}>],
       [<AxesSubplot:title={'center':'SOP'}>,
        <AxesSubplot:title={'center':'LOR '}>,
        <AxesSubplot:title={'center':'CGPA'}>],
       [<AxesSubplot:title={'center':'Research'}>,
        <AxesSubplot:title={'center':'Chance of Admit '}>,
        <AxesSubplot:>]], dtype=object)
```

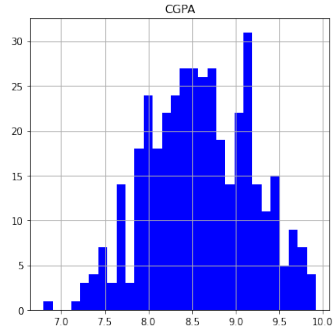


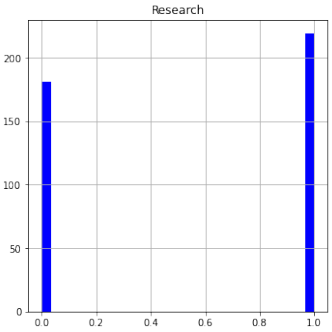


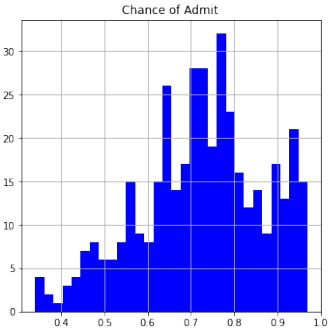












Importing the required libraries for regression model

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
```

Split the dataset into dependent column and independent column

```
x = df[["GRE Score","TOEFL Score","University Rating","SOP","LOR
","CGPA","Research"]]
y = df["Chance of Admit "]

x.head()
```

```
   GRE Score  TOEFL Score  University Rating  SOP  LOR   CGPA
Research
0        337          118                  4  4.5   4.5  9.65
1
1        324          107                  4  4.0   4.5  8.87
1
2        316          104                  3  3.0   3.5  8.00
1
3        322          110                  3  3.5   2.5  8.67
1
4        314          103                  2  2.0   3.0  8.21
0
```

```
y.head()
```

```
0    0.92
1    0.76
2    0.72
3    0.80
4    0.65
Name: Chance of Admit , dtype: float64
```

Spliting the dataset into training and testing data

```
x_train, x_test, y_train, y_test =
train_test_split(x,y,test_size=0.2,random_state=1)
```

MULTIPLE LINEAR REGRESSION

```
multiple_lin_reg = LinearRegression()
multiple_lin_reg.fit(x_train,y_train)

y_pred_mlr = multiple_lin_reg.predict(x_test)

r2_score_mlr = r2_score(y_test,y_pred_mlr)
print("Mutiple Linear Regression's Score =
{:.3f}".format(r2_score_mlr))
```

```
Mutiple Linear Regression's Score = 0.808
```

RANDOM FOREST REGRESSION

```
ran_for_reg = RandomForestRegressor(n_estimators=100,random_state=1)
ran_for_reg.fit(x_train,y_train)

y_pred_rfr = ran_for_reg.predict(x_test)

r2_score_rfr = r2_score(y_test,y_pred_rfr)
```

```
print("Random Forest Regression's Score =
{:.3f}".format(r2_score_rfr))
```

```
Random Forest Regression's Score = 0.778
```

Conclusion

R^2 score is an indicator of accuracy of Regression Models, and the accuracy is measured as close to 1 of this value. Therefore, as seen, Multiple Linear Regression Model is better than Random Forest Regression on this dataset when comparing their R^2 scores.

Save the Model in IBM cloud

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

```
Requirement already satisfied: ibm-watson-machine-learning in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: importlib-metadata in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-
watson-machine-learning) (4.8.2)
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: urllib3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-
watson-machine-learning) (1.26.7)
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/python3.9/site-packages (from ibm-watson-machine-learning)
(0.8.9)
Requirement already satisfied: packaging in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning)
(21.3)
Requirement already satisfied: requests in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning)
(2.26.0)
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: 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-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-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-learning) (0.10.0)
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-machine-learning) (2.11.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in
/opt/conda/envs/Python-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/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-
watson-machine-learning) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/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/python3.9/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: idna<4,>=2.5 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from requests->ibm-watson-machine-
learning) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests-
>ibm-watson-machine-learning) (2.0.4)
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-
3.9/lib/python3.9/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)
Note: you may need to restart the kernel to use updated packages.
```

```
from ibm_watson_machine_learning import APIClient

wml_credentails={
    "apikey":"dBZwh_kI4ymMDFrMHVa3Qt4_hBk-ezkorDqJNV6t7Nl_",
    "url":"https://us-south.ml.cloud.ibm.com"
}

wml_client=APIClient(wml_credentails)
wml_client.spaces.list()
```

```
Note: 'limit' is not provided. Only first 50 records will be displayed
if the number of records exceed 50
------------------------------------
------------------------------------  ------------------------
ID                                    NAME
CREATED
044f5a7f-14d0-4f72-9a3e-81d3088441e3  University Admit Eligibility
Predictor  2022-11-10T15:52:57.889Z
```

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

space_id="044f5a7f-14d0-4f72-9a3e-81d3088441e3"

wml_client.set.default_space(space_id)

'SUCCESS'

wml_client.software_specifications.list(500)

------------------------------  ------------------------------------
----
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
autoai-kb_rt22.2-py3.10         125b6d9a-5b1f-5e8d-972a-b251688ccf40
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
autoai-ts_rt22.2-py3.10           396b2e83-0953-5b86-9a55-7ce1628a406f
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
pytorch-onnx_rt22.2-py3.10        40e73f55-783a-5535-b3fa-0c8b94291431
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
pytorch-onnx_1.2-py3.6          692a6a4d-2c4d-45ff-a1ed-b167ee55469a
base
spark-mllib_2.3-scala_2.11      7963efe5-bbec-417e-92cf-0574e21b4e8d
base
spark-mllib_2.4-py37            7abc992b-b685-532b-a122-a396a3cdbaab
base
caffe_1.0-py3.6                 7bb3dbe2-da6e-4145-918d-b6d84aa93b6b
base
pytorch-onnx_1.7-py3.7          812c6631-42b7-5613-982b-02098e6c909c
base
cuda-py3.6                      82c79ece-4d12-40e6-8787-a7b9e0f62770
base
tensorflow_1.15-py3.6-horovod   8964680e-d5e4-5bb8-919b-8342c6c0dfd8
base
hybrid_0.1                      8c1a58c6-62b5-4dc4-987a-df751c2756b6
base
pytorch-onnx_1.3-py3.7          8d5d8a87-a912-54cf-81ec-3914adaa988d
base
caffe-ibm_1.0-py3.6             8d863266-7927-4d1e-97d7-56a7f4c0a19b
base
spss-modeler_17.1               902d0051-84bd-4af6-ab6b-8f6aa6fdeabb
base
do_12.10                        9100fd72-8159-4eb9-8a0b-a87e12eefa36
base
do_py3.7                        9447fa8b-2051-4d24-9eef-5acb0e3c59f8
base
spark-mllib_3.0-r_3.6           94bb6052-c837-589d-83f1-f4142f219e32
base
cuda-py3.7-opence               94e9652b-7f2d-59d5-ba5a-23a414ea488f
base
```

```
nlp-py3.8                          96e60351-99d4-5a1c-9cc0-473ac1b5a864
base
cuda-py3.7                         9a44990c-1aa1-4c7d-baf8-c4099011741c
base
hybrid_0.2                         9b3f9040-9cee-4ead-8d7a-780600f542f7
base
spark-mllib_3.0-py38               9f7a8fc1-4d3c-5e65-ab90-41fa8de2d418
base
autoai-kb_3.3-py3.7                a545cca3-02df-5c61-9e88-998b09dc79af
base
spark-mllib_3.0-py39               a6082a27-5acc-5163-b02c-6b96916eb5e0
base
runtime-22.1-py3.9-do              a7e7dbf1-1d03-5544-994d-e5ec845ce99a
base
default_py3.8                      ab9e1b80-f2ce-592c-a7d2-4f2344f77194
base
tensorflow_rt22.1-py3.9            acd9c798-6974-5d2f-a657-ce06e986df4d
base
kernel-spark3.2-py3.9              ad7033ee-794e-58cf-812e-a95f4b64b207
base
autoai-obm_2.0 with Spark 3.0      af10f35f-69fa-5d66-9bf5-acb58434263a
base
default_py3.7_opence               c2057dd4-f42c-5f77-a02f-72bdbd3282c9
base
tensorflow_2.1-py3.7               c4032338-2a40-500a-beef-b01ab2667e27
base
do_py3.7_opence                    cc8f8976-b74a-551a-bb66-6377f8d865b4
base
spark-mllib_3.3                    d11f2434-4fc7-58b7-8a62-755da64fdaf8
base
autoai-kb_3.0-py3.6                d139f196-e04b-5d8b-9140-9a10ca1fa91a
base
spark-mllib_3.0-py36               d82546d5-dd78-5fbb-9131-2ec309bc56ed
base
autoai-kb_3.4-py3.8                da9b39c3-758c-5a4f-9cfd-457dd4d8c395
base
kernel-spark3.2-r3.6               db2fe4d6-d641-5d05-9972-73c654c60e0a
base
autoai-kb_rt22.1-py3.9             db6afe93-665f-5910-b117-d879897404d9
base
tensorflow_rt22.1-py3.9-horovod    dda170cc-ca67-5da7-9b7a-cf84c6987fae
base
autoai-ts_1.0-py3.7                deef04f0-0c42-5147-9711-89f9904299db
base
tensorflow_2.1-py3.7-horovod       e384fce5-fdd1-53f8-bc71-11326c9c635f
base
default_py3.7                      e4429883-c883-42b6-87a8-f419d64088cd
base
do_22.1                            e51999ba-6452-5f1f-8287-17228b88b652
base
```

```
autoai-obm_3.2                    eae86aab-da30-5229-a6a6-1d0d4e368983
base
tensorflow_rt22.2-py3.10          f65bd165-f057-55de-b5cb-f97cf2c0f393
base
do_20.1                           f686cdd9-7904-5f9d-a732-01b0d6b10dc5
base
pytorch-onnx_rt22.2-py3.10-edt    f8a05d07-e7cd-57bb-a10b-23f1d4b837ac
base
scikit-learn_0.19-py3.6           f963fa9d-4bb7-5652-9c5d-8d9289ef6ad9
base
tensorflow_2.4-py3.8              fe185c44-9a99-5425-986b-59bd1d2eda46
base
-------------------------------  ------------------------------------
----

MODEL_NAME='MULTIPLE_LINEAR_REGRESSION'
DEPLOYMENT_NAME='MULTIPLE_LINEAR_REGRESSION_DEPLOY'
LINEAR_MODEL=multiple_lin_reg

#setup python version
software_spec_uid=wml_client.software_specifications.get_id_by_name('r
untime-22.1-py3.9')

#setup for 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_u
id

}

#save the model
models_details=wml_client.repository.store_model(
    model=LINEAR_MODEL,
    meta_props=model_props,
    training_data=x_train,
    training_target=y_train

)

models_details

{'entity': {'hybrid_pipeline_software_specs': [],
  'label_column': 'Chance of Admit ',
  'schemas': {'input': [{'fields': [{'name': 'GRE Score', 'type':
'int64'},
      {'name': 'TOEFL Score', 'type': 'int64'},
      {'name': 'University Rating', 'type': 'int64'},
      {'name': 'SOP', 'type': 'float64'},
```

```
     {'name': 'LOR ', 'type': 'float64'},
     {'name': 'CGPA', 'type': 'float64'},
     {'name': 'Research', 'type': 'int64'}],
    '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': '2022-11-12T15:37:03.402Z',
 'id': '1b6400aa-f639-42f7-8d61-b7c9dee7ee47',
 'modified_at': '2022-11-12T15:37:09.028Z',
 'name': 'MULTIPLE_LINEAR_REGRESSION',
 'owner': 'IBMid-666002MQ5W',
 'resource_key': 'd8e8b193-7dbf-4863-bfe2-ce04a7309780',
 'space_id': '044f5a7f-14d0-4f72-9a3e-81d3088441e3'},
'system': {'warnings': []}}
```