Data Exploration GRE Scores Case Study

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
#reading the data
df= pd.read_csv("/content/Admission_Predict.csv")
#how the data looks
df.head()
```

	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

```
print("DATA INFORMATION AND DATA TYPES")
df.info()
```

```
DATA INFORMATION AND DATA TYPES
<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
```

```
print('MISSING DATA (IF ANY)')
df.isnull().sum()
```

```
MISSING DATA (IF ANY)
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
```

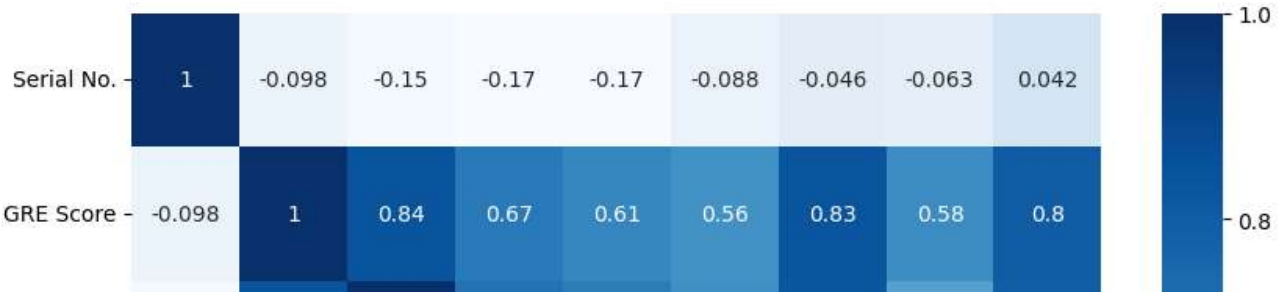
```
df.corr()
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Cl
Serial No.	1.000000	-0.097526	-0.147932	-0.169948	-0.166932	-0.088221	-0.045608	-0.063138	
GRE Score	-0.097526	1.000000	0.835977	0.668976	0.612831	0.557555	0.833060	0.580391	
TOEFL Score	-0.147932	0.835977	1.000000	0.695590	0.657981	0.567721	0.828417	0.489858	
University Rating	-0.169948	0.668976	0.695590	1.000000	0.734523	0.660123	0.746479	0.447783	
SOP	-0.166932	0.612831	0.657981	0.734523	1.000000	0.729593	0.718144	0.444029	
LOR	-0.088221	0.557555	0.567721	0.660123	0.729593	1.000000	0.670211	0.396859	
CGPA	-0.045608	0.833060	0.828417	0.746479	0.718144	0.670211	1.000000	0.521654	
Research	-0.063138	0.580391	0.489858	0.447783	0.444029	0.396859	0.521654	1.000000	

There is a 0.802 correlation between the GRE score and the chance of admission. So there might be a big chance that these variables (data) are highly related. In fact, the correlation is the second-highest, after the CGPA. So, we can determine that CGPA and GRE scores are most important in determining the chances of admission.

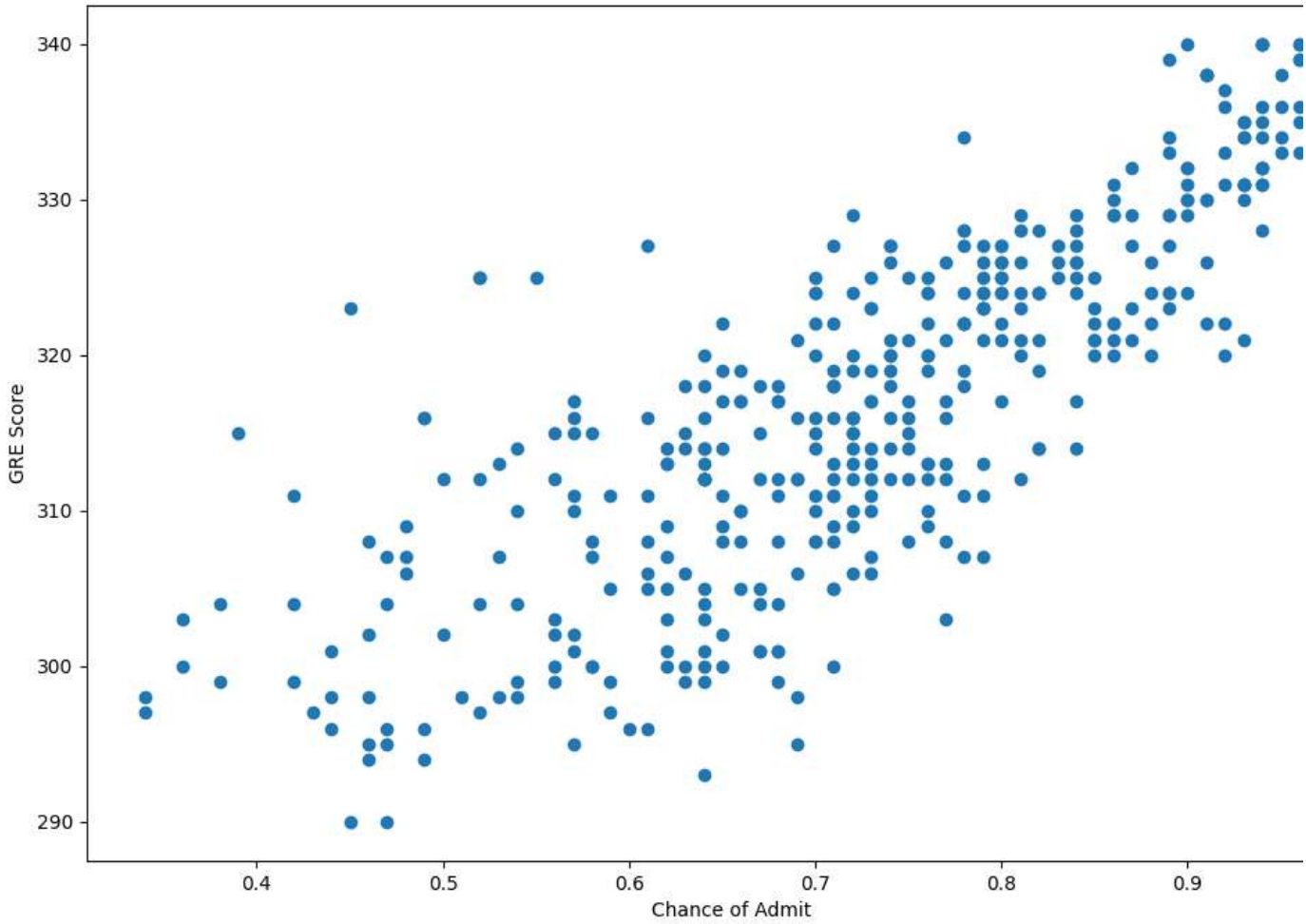
```
plt.figure(figsize = (10,10))
sns.heatmap(df.corr(),annot=True, cmap='Blues')
```

```
<Axes: >
```



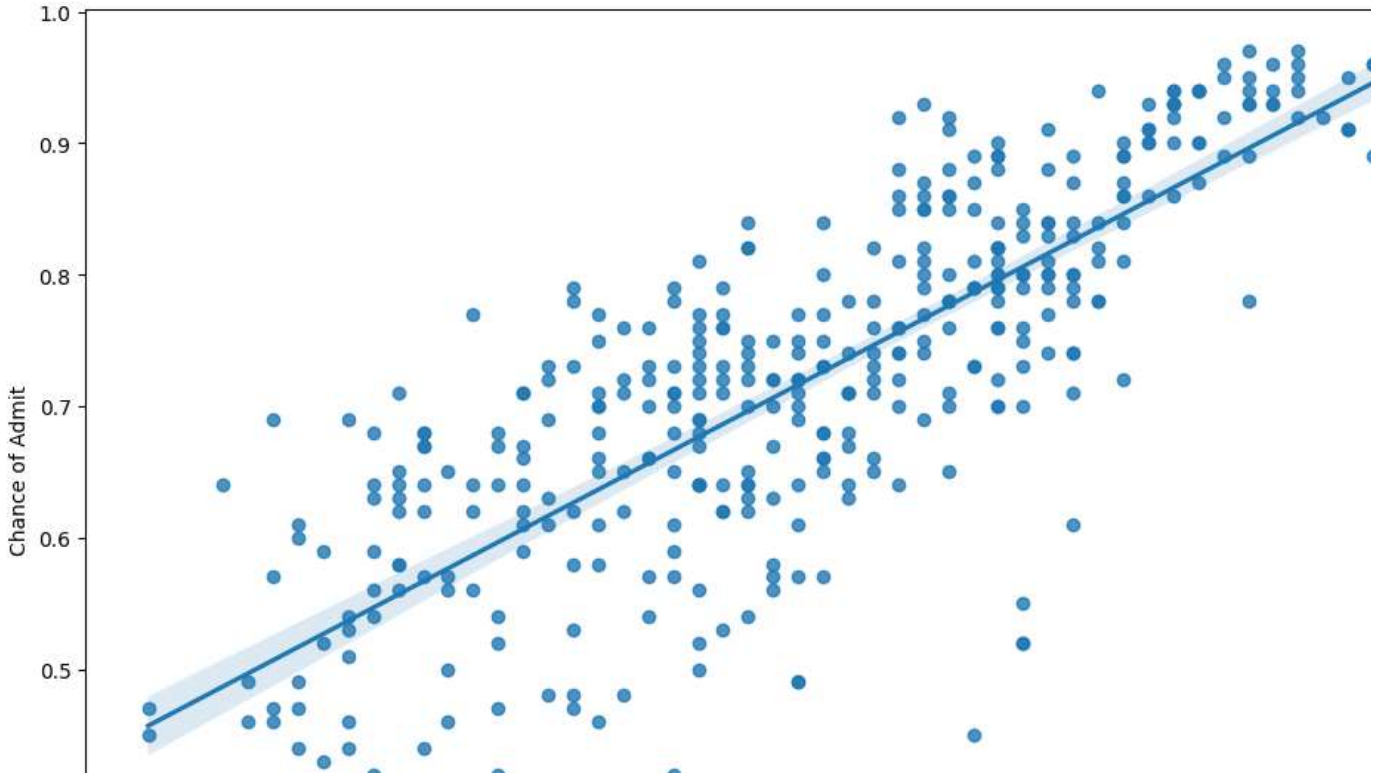

```
plt.subplots(figsize=(12,8))
plt.scatter(df["Chance of Admit "],df["GRE Score"])
plt.xlabel("Chance of Admit")
plt.ylabel("GRE Score")
```

```
Text(0, 0.5, 'GRE Score')
```




```
#There does appear to be a connection between the two variables. Some exploration needs to be done.

plt.subplots(figsize=(12,8))
sns.regplot(x="GRE Score", y="Chance of Admit ", data=df)
```

```
<Axes: xlabel='GRE Score', ylabel='Chance of Admit '>
```

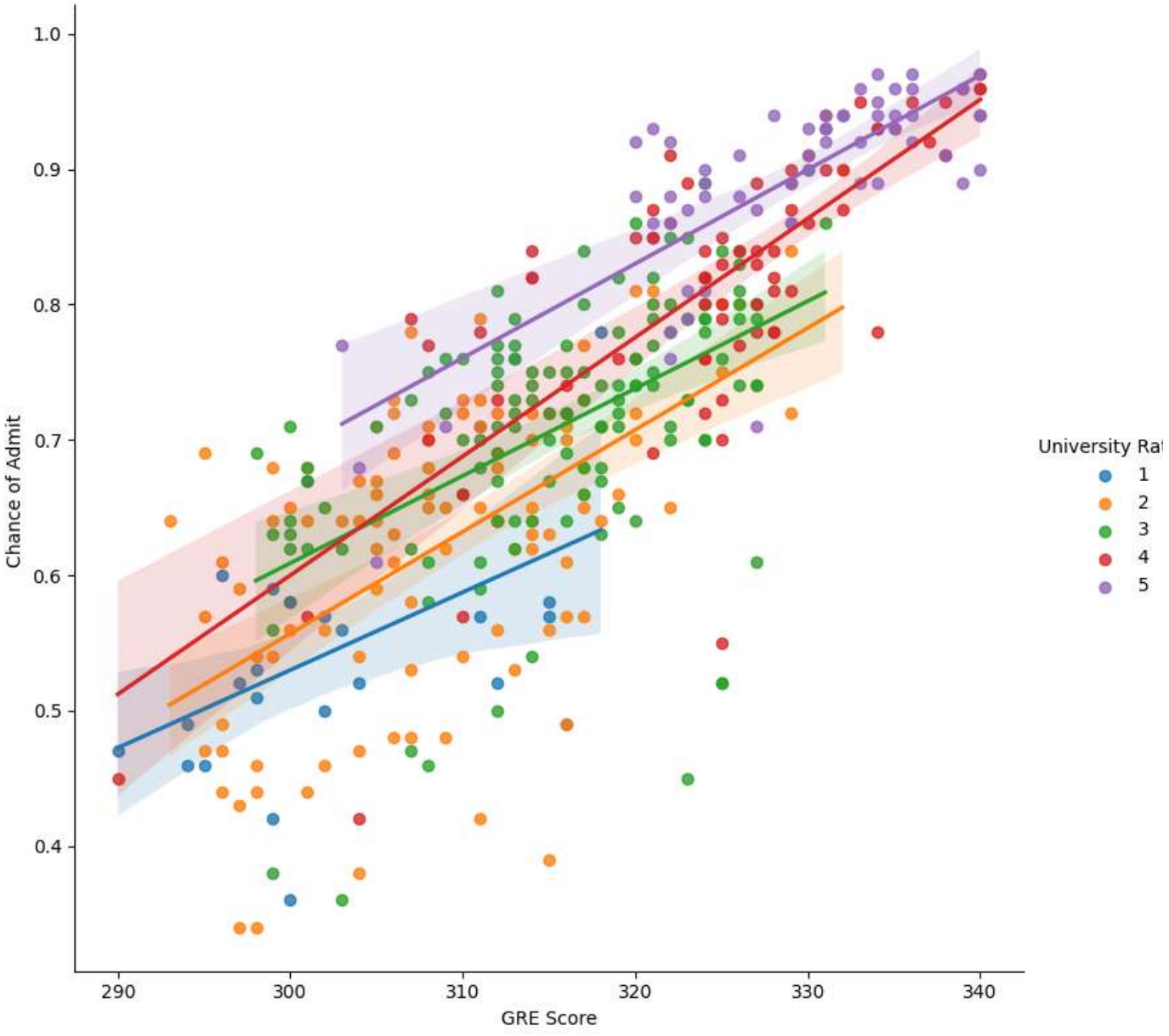



```
# Research experience of a candidate helps in getting admits
sns.lmplot(x="GRE Score", y="Chance of Admit ", data=df, hue="Research",height= 8)
#The data does show that candidates having research experience (orange in the figure), usually have more chance of admit
#Having research experience is very important.
```

```
<seaborn.axisgrid.FacetGrid at 0x7c70eb00d6c0>
```


```
#university ratings

sns.lmplot(x="GRE Score", y="Chance of Admit ", data=df, hue="University Rating",height=8)
```

```
<seaborn.axisgrid.FacetGrid at 0x7c70eb09ffd0>
```



Observations :

Students having higher GRE scores (>320) usually have a high chance of admission into the university with higher ratings (4/5). A lower GRE score has a lower chance of admission, that too for universities of low ratings. Students having a higher chance of

admission, all have good GRE scores and University ratings of 4 or 5. Now we take some data where we take chances of admit to being 0.8 or higher and check how important are GRE scores.

```
admit_high_chance= df[df["Chance of Admit "]>=0.8]
admit_high_chance.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 128 entries, 0 to 399
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Serial No.         128 non-null    int64  
 1   GRE Score          128 non-null    int64  
 2   TOEFL Score        128 non-null    int64  
 3   University Rating  128 non-null    int64  
 4   SOP                128 non-null    float64
 5   LOR                128 non-null    float64
 6   CGPA               128 non-null    float64
 7   Research           128 non-null    int64  
 8   Chance of Admit    128 non-null    float64
dtypes: float64(4), int64(5)
memory usage: 10.0 KB
```

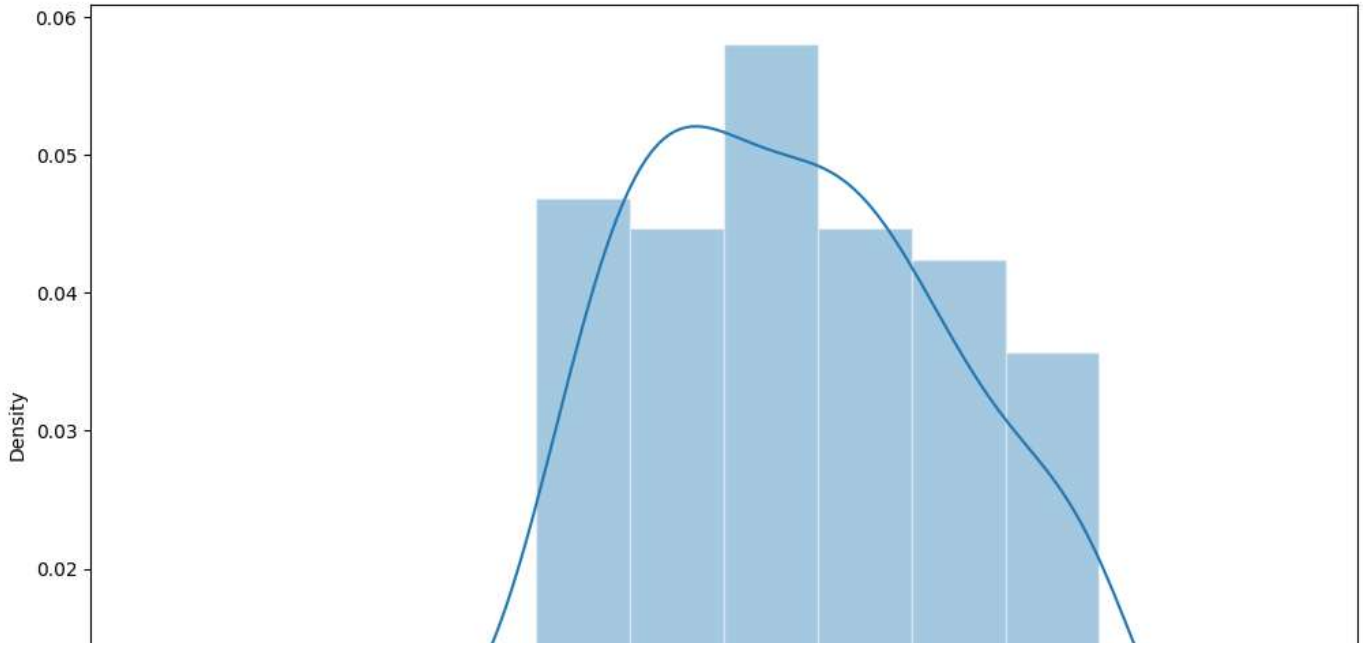
```
admit_high_chance.corr()
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Cl
Serial No.	1.000000	-0.140435	-0.223184	-0.211793	-0.088391	-0.141164	-0.220561	-0.031246	-
GRE Score	-0.140435	1.000000	0.722463	0.358013	0.320138	0.246629	0.754434	0.167532	
TOEFL Score	-0.223184	0.722463	1.000000	0.274811	0.337175	0.302047	0.648308	0.083921	
University Rating	-0.211793	0.358013	0.274811	1.000000	0.584860	0.531448	0.479284	0.190083	
SOP	-0.088391	0.320138	0.337175	0.584860	1.000000	0.601405	0.519791	0.148911	
LOR	-0.141164	0.246629	0.302047	0.531448	0.601405	1.000000	0.441634	0.050772	
CGPA	-0.220561	0.754434	0.648308	0.479284	0.519791	0.441634	1.000000	0.158186	
Research	-0.031246	0.167532	0.083921	0.190083	0.148911	0.050772	0.158186	1.000000	
Chance of Admit	-0.227214	0.716187	0.673774	0.584556	0.565463	0.488480	0.871533	0.226028	

Now let us look at the distribution of Chance of Admit and GRE score.

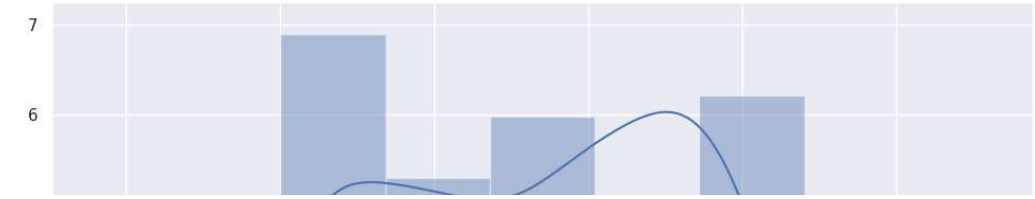
```
plt.subplots(figsize=(12,8))
sns.set_theme(style="darkgrid")
sns.distplot( admit_high_chance["GRE Score"])
```

```
<ipython-input-13-02911b31d2ab>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot( admit_high_chance["GRE Score"])
<Axes: xlabel='GRE Score', ylabel='Density'>
```




```
plt.subplots(figsize=(12,8))
sns.set_theme(style="darkgrid")
sns.distplot( admit_high_chance["Chance of Admit "])
```

```
<ipython-input-14-ab381f61a609>:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot( admit_high_chance["Chance of Admit "])
<Axes: xlabel='Chance of Admit ', ylabel='Density'>
```



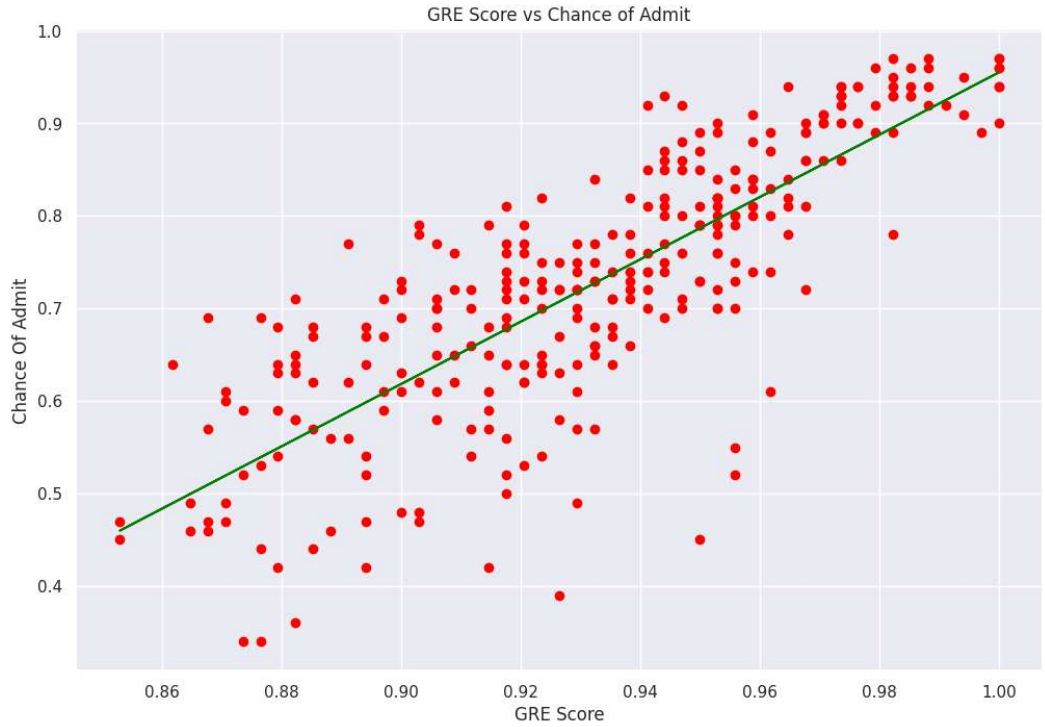
Observations :

For a higher chance of admission, the GRE score is also high. Maximum GRE scores are in the range of 320-340.

```
#Linear Regression between GRE Scores and the chance of admit:
X= df["GRE Score"].values
#bringing GRE score in a range of 0-1

X=X/340
y= df["Chance of Admit "].values
#sk learn train test split data

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
#sk learn linear regression

from sklearn.linear_model import LinearRegression
lr = LinearRegression()
#training the model on training data
lr.fit(X_train.reshape(-1,1), y_train)
y_pred = lr.predict(X_test.reshape(-1,1))
#model score

lr.score(X_test.reshape(-1,1),y_test.reshape(-1,1))
```

```
0.6261897869882879
```

```
plt.subplots(figsize=(12,8))
plt.scatter(X_train, y_train, color = "red")
plt.plot(X_train, lr.predict(X_train.reshape(-1,1)), color = "green")
plt.title("GRE Score vs Chance of Admit")
plt.xlabel("GRE Score")
plt.ylabel("Chance Of Admit")
plt.show()
```



The model is not performing that well, but we do understand that there is a correlation between GRE scores and the chance of admit.

```
#test input

test= 320
val= test/340

val_out=lr.predict(np.array([[val]]))

print("Chance of admission :", val_out[0])
```

```
Chance of admission : 0.7569886709286315
```

```
#Creating a Model on the entire data:
x = df.drop(['Chance of Admit ','Serial No.'],axis=1)
y = df['Chance of Admit ']
X_train, X_test, y_train, y_test = train_test_split(x,y,test_size=0.25, random_state = 7)
#random forest regression

from sklearn.ensemble import RandomForestRegressor

regr = RandomForestRegressor(max_depth=2, random_state=0, n_estimators=5)
regr.fit(X_train,y_train)
regr.score(X_test, y_test)
```

```
0.6901443456671795
```

```
#Let us work with a sample input.

val=regr.predict([[325, 100, 3, 4.1, 3.7, 7.67, 1]])

print("Your chances are (in %):")
print(val[0]*100)
```

```
Your chances are (in %):
54.47694678499888
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but
  warnings.warn(
```

Conclusion: GRE Score is important for admission. Students having good GRE score, seem to have good overall profiles. There are obviously exceptions, which comprise the outliers.