In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df=pd.read_csv("13_placement.csv")
df
```

Out[2]:

	cgpa	placement_exam_marks	placed
0	7.19	26.0	1
1	7.46	38.0	1
2	7.54	40.0	1
3	6.42	8.0	1
4	7.23	17.0	0
...	...	...	...
995	8.87	44.0	1
996	9.12	65.0	1
997	4.89	34.0	0
998	8.62	46.0	1
999	4.90	10.0	1

1000 rows × 3 columns

In [3]:
```
df.head()
```

Out[3]:

	cgpa	placement_exam_marks	placed
0	7.19	26.0	1
1	7.46	38.0	1
2	7.54	40.0	1
3	6.42	8.0	1
4	7.23	17.0	0

DATA CLEANING AND DATA PREPROCESSING

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 3 columns):
```

```
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   cgpa                  1000 non-null   float64
 1   placement_exam_marks  1000 non-null   float64
 2   placed                1000 non-null   int64  
dtypes: float64(2), int64(1)
memory usage: 23.6 KB
```

In [5]:
```
df.describe()
```

Out[5]:

	cgpa	placement_exam_marks	placed
count	1000.000000	1000.000000	1000.000000
mean	6.961240	32.225000	0.489000
std	0.615898	19.130822	0.500129
min	4.890000	0.000000	0.000000
25%	6.550000	17.000000	0.000000
50%	6.960000	28.000000	0.000000
75%	7.370000	44.000000	1.000000
max	9.120000	100.000000	1.000000

In [6]:
```
df.columns
```

Out[6]:
```
Index(['cgpa', 'placement_exam_marks', 'placed'], dtype='object')
```

In [7]:
```
df1=df.dropna(axis=1)
df1
```

Out[7]:

	cgpa	placement_exam_marks	placed
0	7.19	26.0	1
1	7.46	38.0	1
2	7.54	40.0	1
3	6.42	8.0	1
4	7.23	17.0	0
...	...	...	...
995	8.87	44.0	1
996	9.12	65.0	1
997	4.89	34.0	0
998	8.62	46.0	1
999	4.90	10.0	1

1000 rows × 3 columns

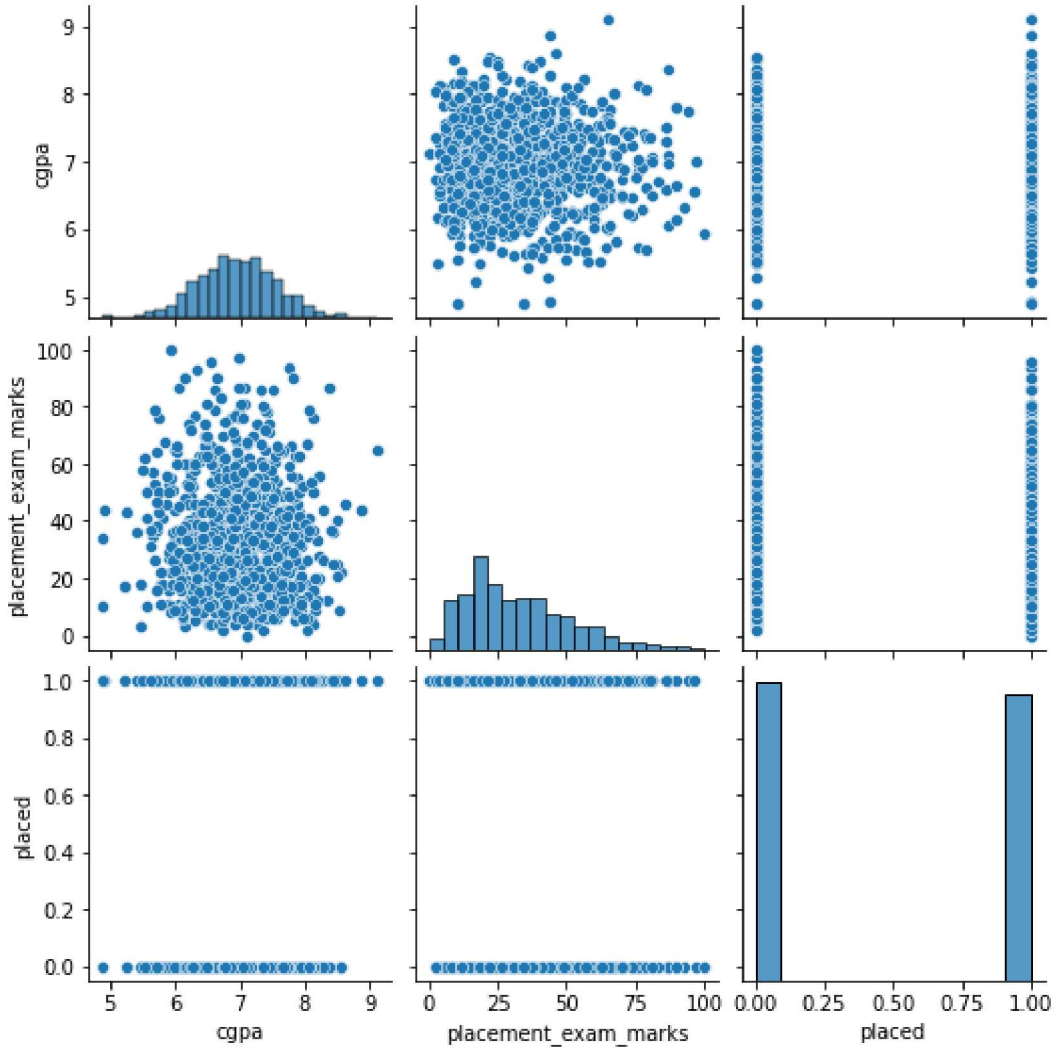
In [8]:
```
df1.columns
```

Out[8]:
```
Index(['cgpa', 'placement_exam_marks', 'placed'], dtype='object')
```

EDA AND VISUALIZATION

In [9]:
```
sns.pairplot(df1)
```

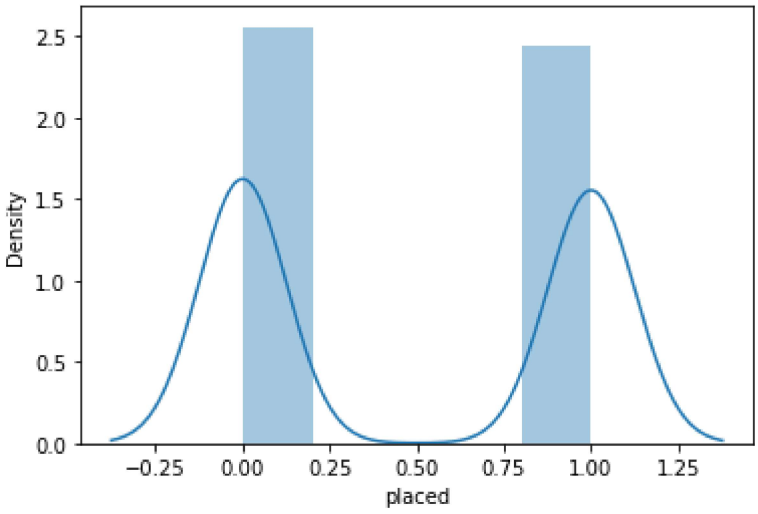
Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x2459137d8b0>
```



In [10]:
```
sns.distplot(df1['placed'])
```

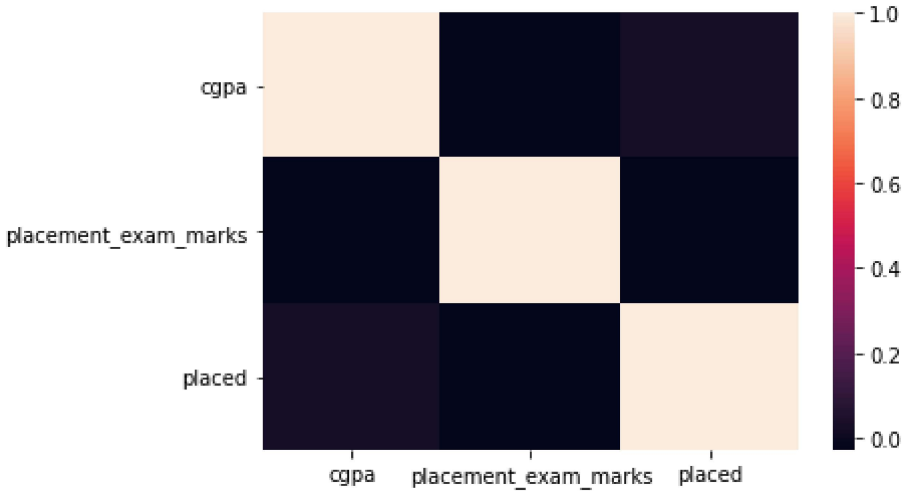
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[10]: <AxesSubplot:xlabel='placed', ylabel='Density'>



In [11]:
```
sns.heatmap(df1.corr())
```

Out[11]: <AxesSubplot:>



TO TRAIN THE MODEL AND MODEL BULDING

In [12]:
```
x=df[['cgpa', 'placement_exam_marks']]
y=df['placed']
```

In [13]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [14]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[14]: LinearRegression()

In [15]:
```
lr.intercept_
```

Out[15]: 0.2978075903914311

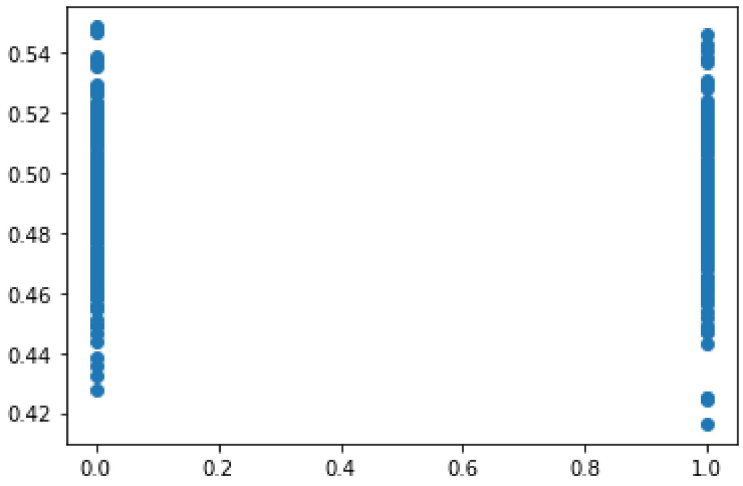
In [16]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[16]:

	Co-efficient
cgpa	0.031243
placement_exam_marks	-0.000809

In [17]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[17]: <matplotlib.collections.PathCollection at 0x24593b0ba30>



ACCURACY

In [18]:
```
lr.score(x_test,y_test)
```

Out[18]: -0.0019770546504029873

In [19]:
```
lr.score(x_train,y_train)
```

Out[19]: 0.0025174732665412813

In [20]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[20]: Ridge(alpha=10)

In [21]:
```
rr.score(x_train,y_train)
```

Out[21]: 0.002515566686323023

In [22]:
```
rr.score(x_test,y_test)
```

Out[22]: -0.0018692666635926614

In [23]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[23]: Lasso(alpha=10)

In [24]:
```
la.score(x_train,y_train)
```

Out[24]:
```
0.0
```

In [25]:
```
la.score(x_test,y_test)
```

Out[25]:
```
-8.166531918929465e-06
```

In [26]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[26]:
```
ElasticNet()
```

In [27]:
```
en.coef_
```

Out[27]:
```
array([ 0., -0.])
```

In [28]:
```
en.intercept_
```

Out[28]:
```
0.48857142857142855
```

In [29]:
```
prediction=en.predict(x_test)
```

In [30]:
```
en.score(x_test,y_test)
```

Out[30]:
```
-8.166531918929465e-06
```

In [31]:
```
from sklearn import metrics
print(metrics.mean_absolute_error(y_test,prediction))
print(metrics.mean_squared_error(y_test,prediction))
print(np.sqrt(metrics.mean_squared_error(y_test,prediction)))
```

```
0.4997714285714286
0.24990204081632655
0.4999020312184444
```