

**A real estate agent want help to predict the house price for regions in Usa.he gave us the dataset to work on to use linear Regression model.Create a model that helps him to estimate**

## Data Collection

```
In [4]: #import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [87]: #import the dataset
data=pd.read_csv(r"C:\Users\user\Desktop\Vicky\13_placement.csv")
```

```
In [88]: #to display top 10 rows
data.head()
```

Out[88]:

	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

```
In [89]: #to display null values
data.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 [90]: #to display summary of statistics  
data.describe()
```

Out[90]:

	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 [91]: #to display columns name  
data.columns
```

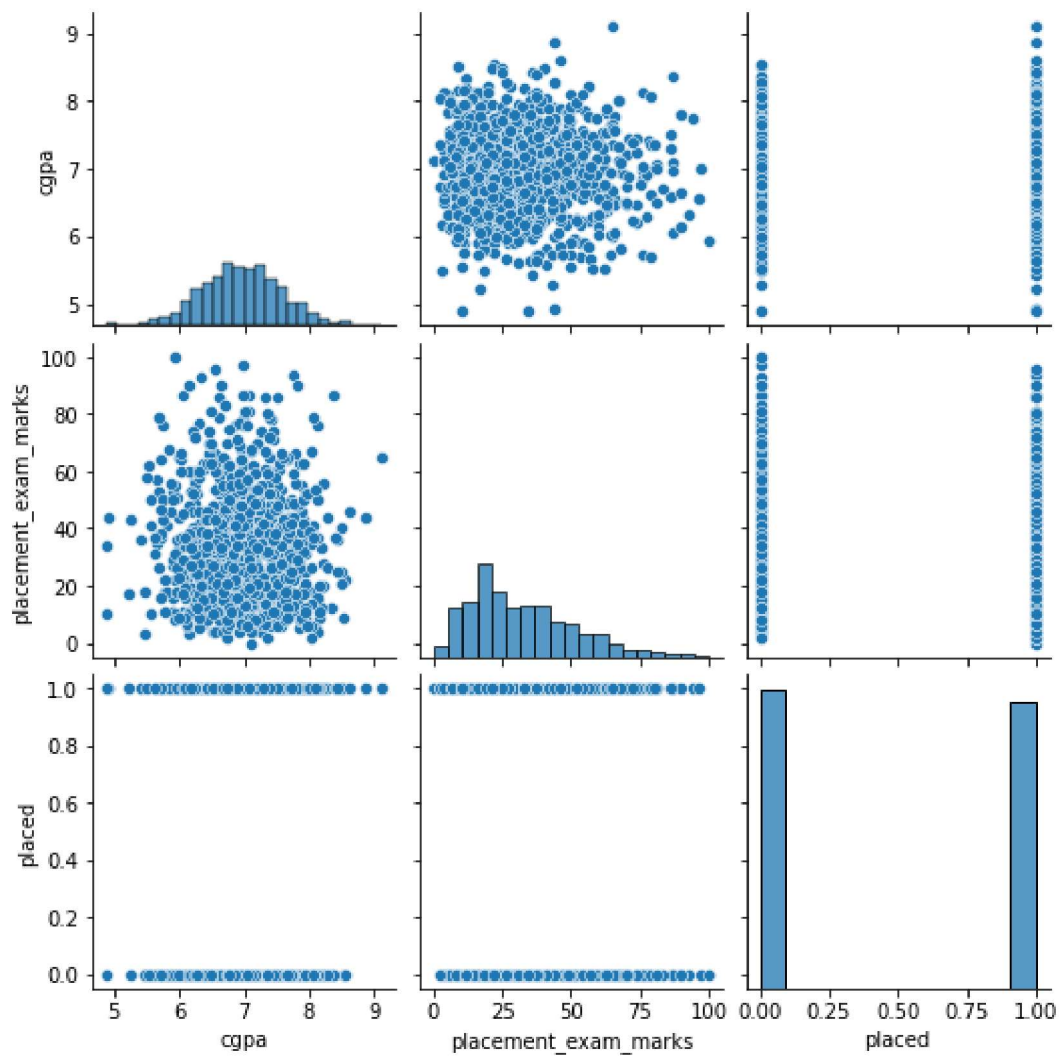
Out[91]: Index(['cgpa', 'placement\_exam\_marks', 'placed'], dtype='object')

```
In [92]: data1=data[['cgpa', 'placement_exam_marks', 'placed']]
```

## EDA and Visualization

```
In [93]: sns.pairplot(data1)
```

```
Out[93]: <seaborn.axisgrid.PairGrid at 0x20fa6ae6400>
```

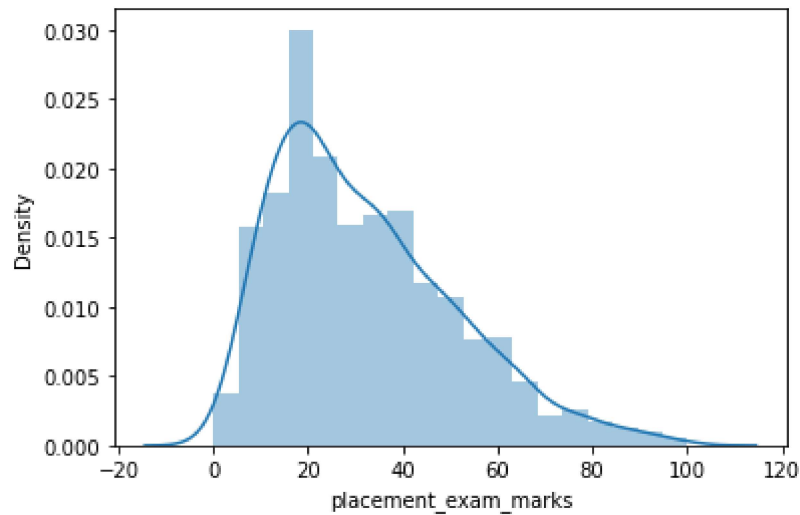


```
In [94]: sns.distplot(data['placement_exam_marks'])
```

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

```
warnings.warn(msg, FutureWarning)
```

```
Out[94]: <AxesSubplot:xlabel='placement_exam_marks', ylabel='Density'>
```

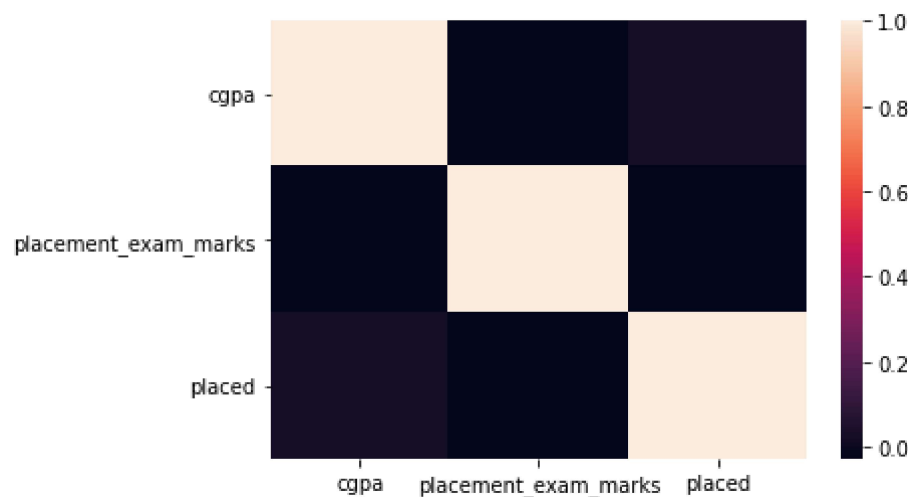


```
In [ ]:
```

```
In [ ]:
```

```
In [95]: sns.heatmap(data1.corr())
```

```
Out[95]: <AxesSubplot:>
```



## To train the model

we are going to train the linear regression model ;We need to split the two variable x and y where x is independent variable (input) and y is dependent of x(output) so we could ignore

```
In [138]: x=data1[['placement_exam_marks', 'placed' ]]
          y=data1['placement_exam_marks']
```

```
In [139]:
```

```
#To split test and train data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.6)
```

```
In [140]: from sklearn.linear_model import LinearRegression
          lr=LinearRegression()
          lr.fit(x_train,y_train)
```

```
Out[140]: LinearRegression()
```

```
In [141]: lr.intercept_
```

```
Out[141]: -7.105427357601002e-15
```

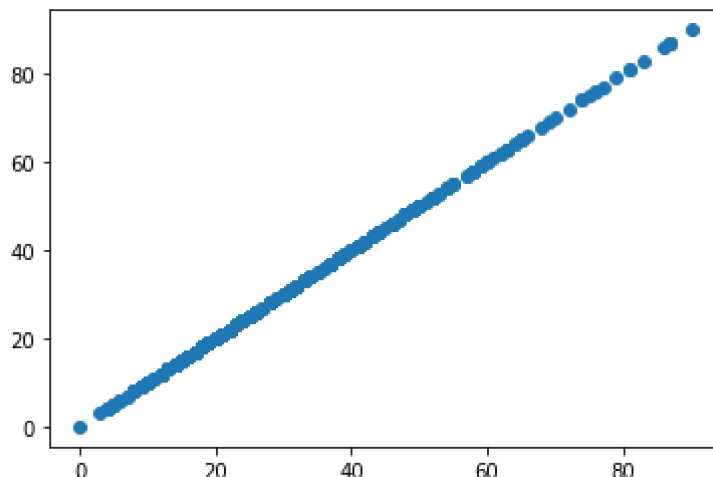
```
In [142]: coeff = pd.DataFrame(lr.coef_,x.columns,columns=["Co-efficient"])
          coeff
```

```
Out[142]:
```

	Co-efficient
placement_exam_marks	1.000000e+00
placed	2.077647e-16

```
In [143]: prediction = lr.predict(x_train)
          plt.scatter(y_train,prediction)
```

```
Out[143]: <matplotlib.collections.PathCollection at 0x20fa84b7b80>
```



In [144]: `lr.score(x_test,y_test)`

Out[144]: 1.0

In [ ]:

In [ ]:

In [ ]: