

d618wxluo

July 28, 2023

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

```
[9]: df=pd.read_csv("/content/17_student_marks.csv")
df
```

```
[9]:
```

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7	\
0	22000	78	87	91	91	88	98	94	
1	22001	79	71	81	72	73	68	59	
2	22002	66	65	70	74	78	86	87	
3	22003	60	58	54	61	54	57	64	
4	22004	99	95	96	93	97	89	92	
5	22005	41	36	35	28	35	36	27	
6	22006	47	50	47	57	62	64	71	
7	22007	84	74	70	68	58	59	56	
8	22008	74	64	58	57	53	51	47	
9	22009	87	81	73	74	71	63	53	
10	22010	40	34	37	33	31	35	39	
11	22011	91	84	78	74	76	80	80	
12	22012	81	83	93	88	89	90	99	
13	22013	52	50	42	38	33	30	28	
14	22014	63	67	65	74	80	86	95	
15	22015	76	82	88	94	85	76	70	
16	22016	83	78	71	71	77	72	66	
17	22017	55	45	43	38	43	35	44	
18	22018	71	67	76	74	64	61	57	
19	22019	62	61	53	49	54	59	68	
20	22020	44	38	36	34	26	34	39	
21	22021	50	56	53	46	41	38	47	
22	22022	57	48	40	45	43	36	26	
23	22023	59	56	52	44	50	40	45	
24	22024	84	92	89	80	90	80	84	
25	22025	74	80	86	87	90	100	95	
26	22026	92	84	74	83	93	83	75	
27	22027	63	70	74	65	64	55	61	

28	22028	78	77	69	76	78	74	67
29	22029	55	58	59	67	71	62	53
30	22030	54	54	48	38	35	45	46
31	22031	84	93	97	89	86	95	100
32	22032	95	100	94	100	98	99	100
33	22033	64	61	63	73	63	68	64
34	22034	76	79	73	77	83	86	95
35	22035	78	71	61	55	54	48	41
36	22036	95	89	91	84	89	94	85
37	22037	99	89	79	87	87	81	82
38	22038	82	83	85	86	89	80	88
39	22039	65	56	64	62	58	51	61
40	22040	100	93	92	86	84	76	82
41	22041	78	72	73	79	81	73	71
42	22042	98	100	100	93	94	92	100
43	22043	58	62	67	77	71	63	64
44	22044	96	92	94	100	99	95	98
45	22045	86	87	85	84	85	91	86
46	22046	48	55	46	40	34	29	37
47	22047	56	52	54	47	40	35	43
48	22048	42	44	46	53	62	59	57
49	22049	64	54	49	59	54	55	57
50	22050	50	44	37	29	37	46	53
51	22051	70	60	70	62	67	67	68
52	22052	63	73	70	63	60	67	61
53	22053	92	100	100	100	100	100	92
54	22054	64	55	54	61	63	57	47
55	22055	60	66	68	58	49	47	39

	Test_8	Test_9	Test_10	Test_11	Test_12
0	100	100	100	100	93
1	69	59	60	61	67
2	96	88	82	90	86
3	62	72	63	72	76
4	98	91	98	95	88
5	26	19	22	27	31
6	75	85	87	85	89
7	56	64	70	67	59
8	45	42	43	34	24
9	45	39	43	46	38
10	38	40	48	44	50
11	73	75	71	79	70
12	99	95	85	75	84
13	22	12	20	19	20
14	96	92	83	75	81
15	60	50	58	49	59
16	75	66	61	61	66

17	37	45	37	45	54
18	64	61	51	51	58
19	74	65	55	60	61
20	44	36	45	35	44
21	39	44	36	43	46
22	19	9	12	22	27
23	46	54	57	52	47
24	74	68	73	81	74
25	87	85	79	85	88
26	82	81	73	70	73
27	58	48	46	46	51
28	69	78	68	65	68
29	61	67	76	75	70
30	47	41	37	30	25
31	100	100	99	100	100
32	90	80	84	75	80
33	58	50	51	56	64
34	89	90	95	100	100
35	32	41	40	48	38
36	91	100	100	100	92
37	74	64	54	51	50
38	95	87	93	90	89
39	68	70	70	63	73
40	74	79	72	79	85
41	77	83	92	97	99
42	100	98	94	97	100
43	73	83	76	86	91
44	92	84	84	84	91
45	82	85	87	84	83
46	34	39	41	31	40
47	44	40	39	47	43
48	53	43	35	37	43
49	59	63	73	78	88
50	57	55	61	64	68
51	67	72	69	64	65
52	59	52	58	56	46
53	87	94	100	94	98
54	37	44	48	54	54
55	29	39	44	39	45

```
[10]: df.head()
```

```
[10]:   Student_ID  Test_1  Test_2  Test_3  Test_4  Test_5  Test_6  Test_7  Test_8  \
0      22000      78      87      91      91      88      98      94      100
1      22001      79      71      81      72      73      68      59      69
2      22002      66      65      70      74      78      86      87      96
3      22003      60      58      54      61      54      57      64      62
```

4	22004	99	95	96	93	97	89	92	98
---	-------	----	----	----	----	----	----	----	----

	Test_9	Test_10	Test_11	Test_12
0	100	100	100	93
1	59	60	61	67
2	88	82	90	86
3	72	63	72	76
4	91	98	95	88

1 DATA CLEANING AND DATA PREPROCESSING

```
[11]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 56 entries, 0 to 55
Data columns (total 13 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Student_ID  56 non-null    int64
1   Test_1      56 non-null    int64
2   Test_2      56 non-null    int64
3   Test_3      56 non-null    int64
4   Test_4      56 non-null    int64
5   Test_5      56 non-null    int64
6   Test_6      56 non-null    int64
7   Test_7      56 non-null    int64
8   Test_8      56 non-null    int64
9   Test_9      56 non-null    int64
10  Test_10     56 non-null    int64
11  Test_11     56 non-null    int64
12  Test_12     56 non-null    int64
dtypes: int64(13)
memory usage: 5.8 KB
```

```
[12]: df.describe()
```

```
[12]:
```

	Student_ID	Test_1	Test_2	Test_3	Test_4 \
count	56.000000	56.000000	56.000000	56.000000	56.000000
mean	22027.500000	70.750000	69.196429	68.089286	67.446429
std	16.309506	17.009356	17.712266	18.838333	19.807179
min	22000.000000	40.000000	34.000000	35.000000	28.000000
25%	22013.750000	57.750000	55.750000	53.000000	54.500000
50%	22027.500000	70.500000	68.500000	70.000000	71.500000
75%	22041.250000	84.000000	83.250000	85.000000	84.000000
max	22055.000000	100.000000	100.000000	100.000000	100.000000

	Test_5	Test_6	Test_7	Test_8	Test_9	Test_10	\
count	56.000000	56.000000	56.000000	56.000000	56.000000	56.000000	
mean	67.303571	66.000000	66.160714	65.303571	64.392857	64.250000	
std	20.746890	21.054043	21.427914	22.728372	23.211814	22.598673	
min	26.000000	29.000000	26.000000	19.000000	9.000000	12.000000	
25%	53.750000	50.250000	47.000000	45.750000	44.000000	45.750000	
50%	69.000000	65.500000	64.000000	67.500000	65.500000	65.500000	
75%	85.250000	83.750000	85.250000	83.250000	84.250000	83.250000	
max	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	

	Test_11	Test_12
count	56.000000	56.000000
mean	64.517857	65.928571
std	22.610529	22.464402
min	19.000000	20.000000
25%	46.750000	46.750000
50%	64.000000	67.500000
75%	84.000000	86.500000
max	100.000000	100.000000

```
[13]: df.columns
```

```
[13]: Index(['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',
          'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11',
          'Test_12'],
          dtype='object')
```

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

```
[14]:
```

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7	\
0	22000	78	87	91	91	88	98	94	
1	22001	79	71	81	72	73	68	59	
2	22002	66	65	70	74	78	86	87	
3	22003	60	58	54	61	54	57	64	
4	22004	99	95	96	93	97	89	92	
5	22005	41	36	35	28	35	36	27	
6	22006	47	50	47	57	62	64	71	
7	22007	84	74	70	68	58	59	56	
8	22008	74	64	58	57	53	51	47	
9	22009	87	81	73	74	71	63	53	
10	22010	40	34	37	33	31	35	39	
11	22011	91	84	78	74	76	80	80	
12	22012	81	83	93	88	89	90	99	
13	22013	52	50	42	38	33	30	28	
14	22014	63	67	65	74	80	86	95	
15	22015	76	82	88	94	85	76	70	

16	22016	83	78	71	71	77	72	66
17	22017	55	45	43	38	43	35	44
18	22018	71	67	76	74	64	61	57
19	22019	62	61	53	49	54	59	68
20	22020	44	38	36	34	26	34	39
21	22021	50	56	53	46	41	38	47
22	22022	57	48	40	45	43	36	26
23	22023	59	56	52	44	50	40	45
24	22024	84	92	89	80	90	80	84
25	22025	74	80	86	87	90	100	95
26	22026	92	84	74	83	93	83	75
27	22027	63	70	74	65	64	55	61
28	22028	78	77	69	76	78	74	67
29	22029	55	58	59	67	71	62	53
30	22030	54	54	48	38	35	45	46
31	22031	84	93	97	89	86	95	100
32	22032	95	100	94	100	98	99	100
33	22033	64	61	63	73	63	68	64
34	22034	76	79	73	77	83	86	95
35	22035	78	71	61	55	54	48	41
36	22036	95	89	91	84	89	94	85
37	22037	99	89	79	87	87	81	82
38	22038	82	83	85	86	89	80	88
39	22039	65	56	64	62	58	51	61
40	22040	100	93	92	86	84	76	82
41	22041	78	72	73	79	81	73	71
42	22042	98	100	100	93	94	92	100
43	22043	58	62	67	77	71	63	64
44	22044	96	92	94	100	99	95	98
45	22045	86	87	85	84	85	91	86
46	22046	48	55	46	40	34	29	37
47	22047	56	52	54	47	40	35	43
48	22048	42	44	46	53	62	59	57
49	22049	64	54	49	59	54	55	57
50	22050	50	44	37	29	37	46	53
51	22051	70	60	70	62	67	67	68
52	22052	63	73	70	63	60	67	61
53	22053	92	100	100	100	100	100	92
54	22054	64	55	54	61	63	57	47
55	22055	60	66	68	58	49	47	39

	Test_8	Test_9	Test_10	Test_11	Test_12
0	100	100	100	100	93
1	69	59	60	61	67
2	96	88	82	90	86
3	62	72	63	72	76
4	98	91	98	95	88

5	26	19	22	27	31
6	75	85	87	85	89
7	56	64	70	67	59
8	45	42	43	34	24
9	45	39	43	46	38
10	38	40	48	44	50
11	73	75	71	79	70
12	99	95	85	75	84
13	22	12	20	19	20
14	96	92	83	75	81
15	60	50	58	49	59
16	75	66	61	61	66
17	37	45	37	45	54
18	64	61	51	51	58
19	74	65	55	60	61
20	44	36	45	35	44
21	39	44	36	43	46
22	19	9	12	22	27
23	46	54	57	52	47
24	74	68	73	81	74
25	87	85	79	85	88
26	82	81	73	70	73
27	58	48	46	46	51
28	69	78	68	65	68
29	61	67	76	75	70
30	47	41	37	30	25
31	100	100	99	100	100
32	90	80	84	75	80
33	58	50	51	56	64
34	89	90	95	100	100
35	32	41	40	48	38
36	91	100	100	100	92
37	74	64	54	51	50
38	95	87	93	90	89
39	68	70	70	63	73
40	74	79	72	79	85
41	77	83	92	97	99
42	100	98	94	97	100
43	73	83	76	86	91
44	92	84	84	84	91
45	82	85	87	84	83
46	34	39	41	31	40
47	44	40	39	47	43
48	53	43	35	37	43
49	59	63	73	78	88
50	57	55	61	64	68
51	67	72	69	64	65

52	59	52	58	56	46
53	87	94	100	94	98
54	37	44	48	54	54
55	29	39	44	39	45

```
[15]: df1.columns
```

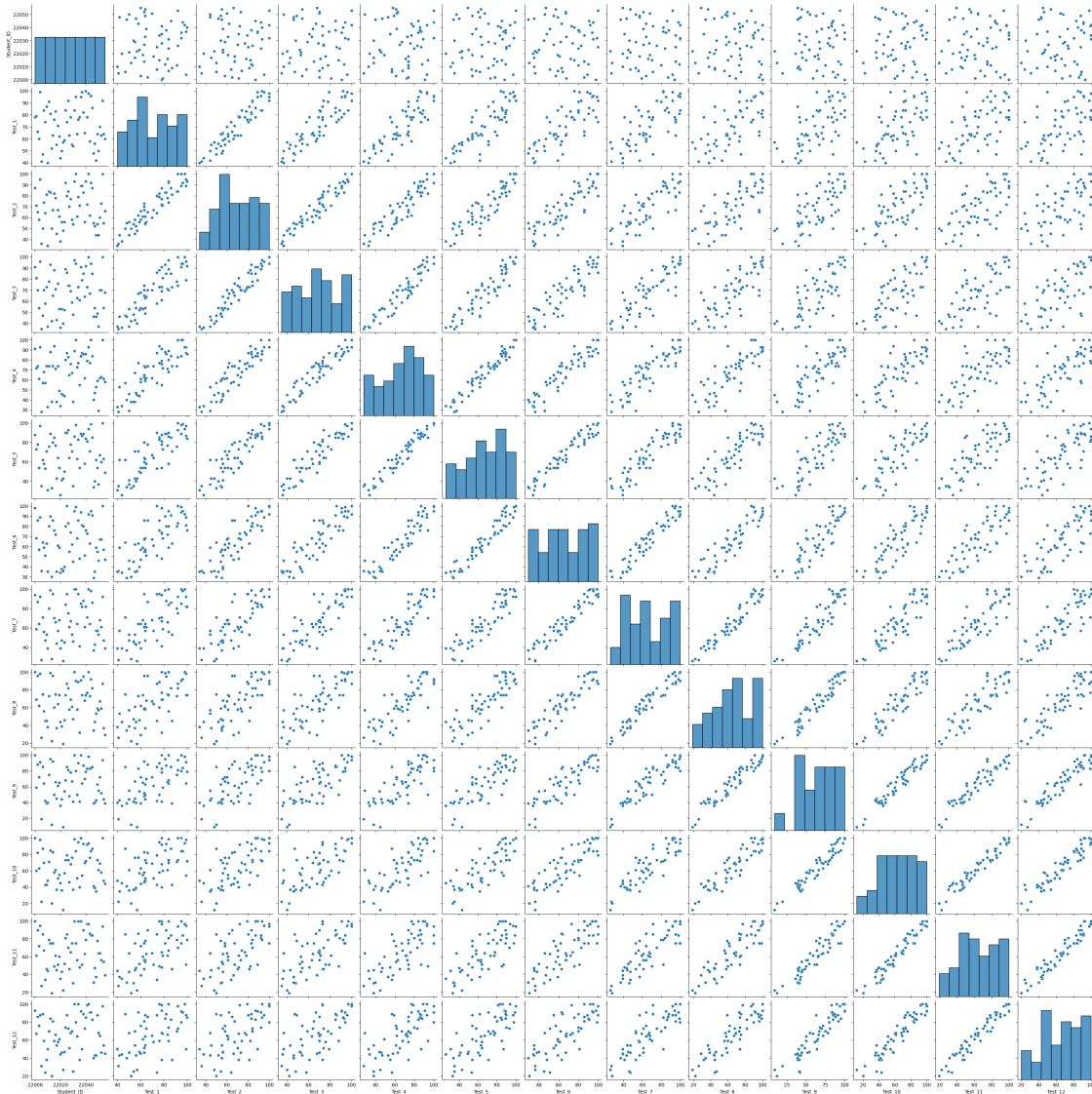
```
[15]: Index(['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',  
          'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11',  
          'Test_12'],  
          dtype='object')
```

```
[16]: df1=df1[['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',  
             'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11',  
             'Test_12']]
```

2 EDA AND VISUALIZATION

```
[17]: sns.pairplot(df1)
```

```
[17]: <seaborn.axisgrid.PairGrid at 0x7b5b08f3db10>
```

```
[18]: sns.distplot(df1['Test_12'])
```

```
<ipython-input-18-c52684cbf714>:1: UserWarning:
```

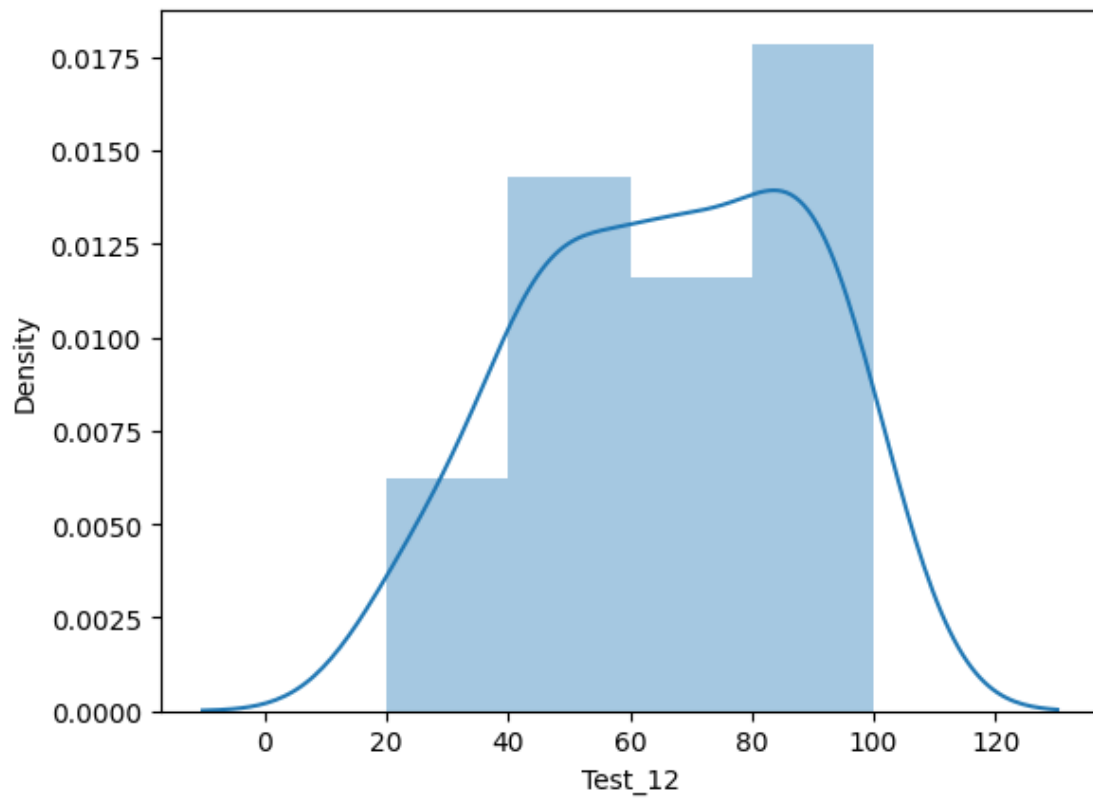
```
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
```

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>

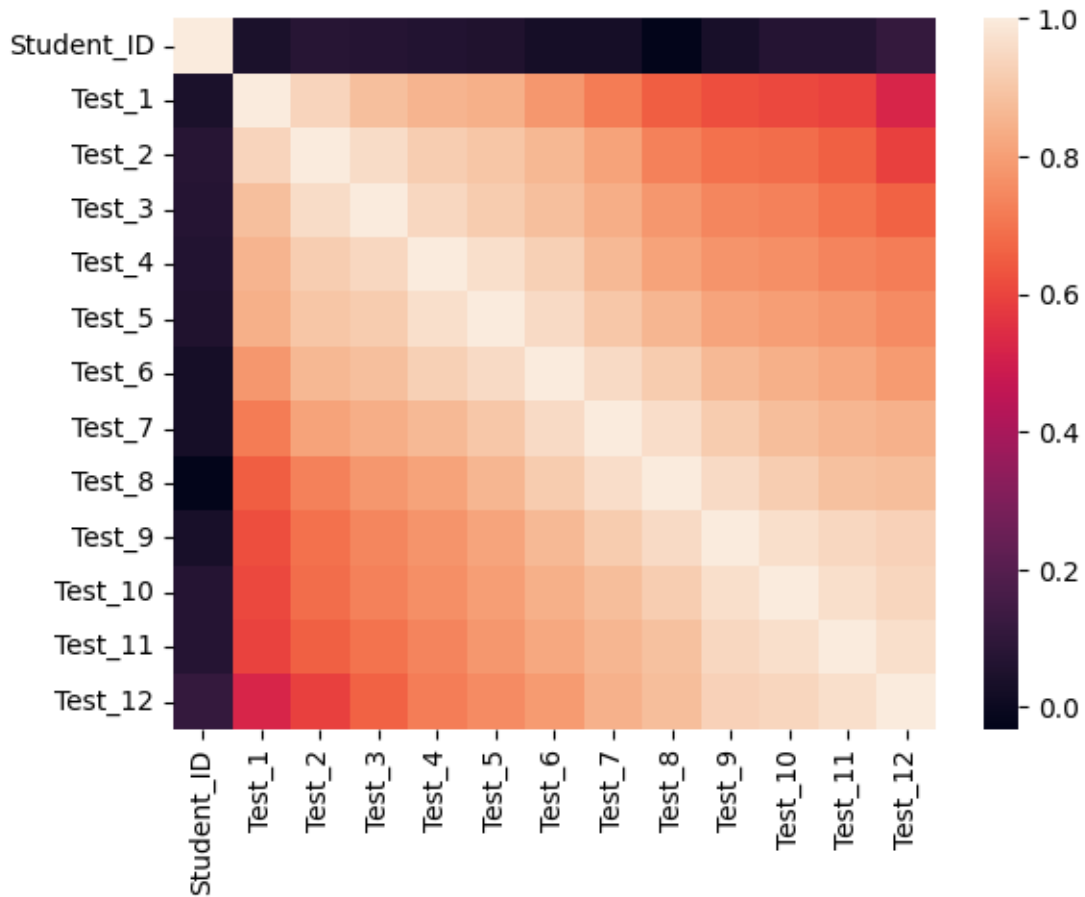
```
sns.distplot(df1['Test_12'])
```

```
[18]: <Axes: xlabel='Test_12', ylabel='Density'>
```



```
[19]: sns.heatmap(df1.corr())
```

```
[19]: <Axes: >
```



3 TO TRAIN THE MODEL AND MODEL BULDING

```
[20]: x=df[['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',
          'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11',
          'Test_12']]
      y=df['Test_12']
```

```
[21]: 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)
```

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

```
[22]: LinearRegression()
```

```
[23]: lr.intercept_
```

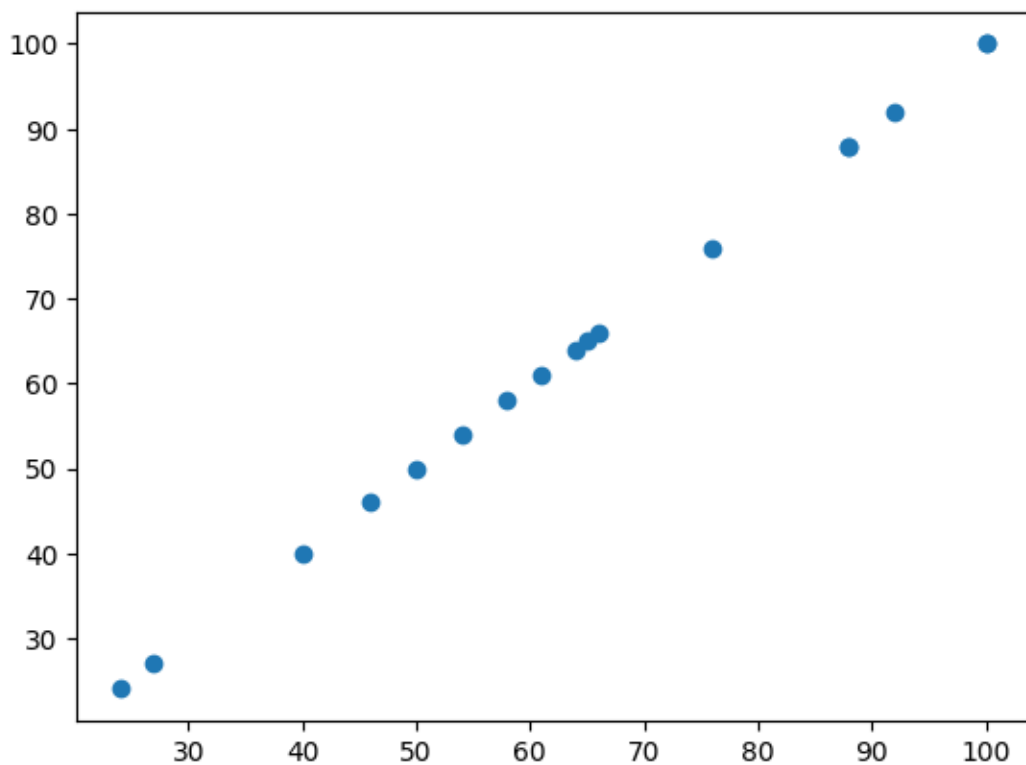
[23]: -1.9895196601282805e-12

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

```
[24]:      Co-efficient  
Student_ID  7.196850e-17  
Test_1      2.775558e-16  
Test_2     -1.665335e-16  
Test_3      9.020562e-16  
Test_4      5.551115e-16  
Test_5      2.914335e-16  
Test_6      7.702172e-16  
Test_7      4.857226e-16  
Test_8      2.706169e-16  
Test_9      7.771561e-16  
Test_10     9.159340e-16  
Test_11     1.942890e-16  
Test_12     1.000000e+00
```

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

[25]: <matplotlib.collections.PathCollection at 0x7b5af90ad780>



4 ACCURACY

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

```
[26]: 1.0
```

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

```
[27]: 1.0
```

```
[28]: from sklearn.linear_model import Ridge,Lasso
```

```
[29]: rr=Ridge(alpha=10)  
      rr.fit(x_train,y_train)
```

```
[29]: Ridge(alpha=10)
```

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

```
[30]: 0.9999712298207973
```

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

```
[31]: 0.9999911737785708
```

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

```
[32]: Lasso(alpha=10)
```

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

```
[33]: 0.9995685108797543
```

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

```
[34]: 0.9995712899401711
```