mk 31/07/23

In [247]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

In [249]: x=pd.read_csv(r"C:\Users\user\Downloads\17_student_marks.csv")
x

Out[249]:

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7	Test_8	Test_9	Test_10	Test_11	1
0	22000	78	87	91	91	88	98	94	100	100	100	100	
1	22001	79	71	81	72	73	68	59	69	59	60	61	
2	22002	66	65	70	74	78	86	87	96	88	82	90	
3	22003	60	58	54	61	54	57	64	62	72	63	72	
4	22004	99	95	96	93	97	89	92	98	91	98	95	
5	22005	41	36	35	28	35	36	27	26	19	22	27	
6	22006	47	50	47	57	62	64	71	75	85	87	85	
7	22007	84	74	70	68	58	59	56	56	64	70	67	
8	22008	74	64	58	57	53	51	47	45	42	43	34	
9	22009	87	81	73	74	71	63	53	45	39	43	46	
10	22010	40	34	37	33	31	35	39	38	40	48	44	
11	22011	91	84	78	74	76	80	80	73	75	71	79	
12	22012	81	83	93	88	89	90	99	99	95	85	75	
13	22013	52	50	42	38	33	30	28	22	12	20	19	
14	22014	63	67	65	74	80	86	95	96	92	83	75	
15	22015	76	82	88	94	85	76	70	60	50	58	49	
16	22016	83	78	71	71	77	72	66	75	66	61	61	
17	22017	55	45	43	38	43	35	44	37	45	37	45	
18	22018	71	67	76	74	64	61	57	64	61	51	51	
19	22019	62	61	53	49	54	59	68	74	65	55	60	
20	22020	44	38	36	34	26	34	39	44	36	45	35	
21	22021	50	56	53	46	41	38	47	39	44	36	43	
22	22022	57	48	40	45	43	36	26	19	9	12	22	
23	22023	59	56	52	44	50	40	45	46	54	57	52	
24	22024	84	92	89	80	90	80	84	74	68	73	81	
25	22025	74	80	86	87	90	100	95	87	85	79	85	
26	22026	92	84	74	83	93	83	75	82	81	73	70	
27	22027	63	70	74	65	64	55	61	58	48	46	46	
28	22028	78	77	69	76	78	74	67	69	78	68	65	
29	22029	55	58	59	67	71	62	53	61	67	76	75	
30	22030	54	54	48	38	35	45	46	47	41	37	30	
31	22031	84	93	97	89	86	95	100	100	100	99	100	
32	22032	95	100	94	100	98	99	100	90	80	84	75	
33	22033	64	61	63	73	63	68	64	58	50	51	56	
34	22034	76	79	73	77	83	86	95	89	90	95	100	
35	22035	78	71	61	55	54	48	41	32	41	40	48	
36	22036	95	89	91	84	89	94	85	91	100	100	100	
37	22037	99	89	79	87	87	81	82	74	64	54	51	

Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7	Test_8	Test_9	Test_10	Test_11	T
22038	82	83	85	86	89	80	88	95	87	93	90	
22039	65	56	64	62	58	51	61	68	70	70	63	
22040	100	93	92	86	84	76	82	74	79	72	79	
22041	78	72	73	79	81	73	71	77	83	92	97	
22042	98	100	100	93	94	92	100	100	98	94	97	
22043	58	62	67	77	71	63	64	73	83	76	86	
22044	96	92	94	100	99	95	98	92	84	84	84	
22045	86	87	85	84	85	91	86	82	85	87	84	
22046	48	55	46	40	34	29	37	34	39	41	31	
22047	56	52	54	47	40	35	43	44	40	39	47	
22048	42	44	46	53	62	59	57	53	43	35	37	
22049	64	54	49	59	54	55	57	59	63	73	78	
22050	50	44	37	29	37	46	53	57	55	61	64	
22051	70	60	70	62	67	67	68	67	72	69	64	
22052	63	73	70	63	60	67	61	59	52	58	56	
22053	92	100	100	100	100	100	92	87	94	100	94	
22054	64	55	54	61	63	57	47	37	44	48	54	
22055	60	66	68	58	49	47	39	29	39	44	39	
	22038 22039 22040 22041 22042 22043 22044 22045 22046 22047 22048 22049 22050 22051 22052 22053 22054	22038 82 22039 65 22040 100 22041 78 22042 98 22043 58 22044 96 22045 86 22046 48 22047 56 22048 42 22049 64 22050 50 22051 70 22052 63 22053 92 22054 64	22038 82 83 22039 65 56 22040 100 93 22041 78 72 22042 98 100 22043 58 62 22044 96 92 22045 86 87 22046 48 55 22047 56 52 22048 42 44 22049 64 54 22050 50 44 22051 70 60 22052 63 73 22053 92 100 22054 64 55	22038 82 83 85 22039 65 56 64 22040 100 93 92 22041 78 72 73 22042 98 100 100 22043 58 62 67 22044 96 92 94 22045 86 87 85 22046 48 55 46 22047 56 52 54 22048 42 44 46 22049 64 54 49 22050 50 44 37 22051 70 60 70 22052 63 73 70 22053 92 100 100 22054 64 55 54	22038 82 83 85 86 22039 65 56 64 62 22040 100 93 92 86 22041 78 72 73 79 22042 98 100 100 93 22043 58 62 67 77 22044 96 92 94 100 22045 86 87 85 84 22046 48 55 46 40 22047 56 52 54 47 22048 42 44 46 53 22049 64 54 49 59 22050 50 44 37 29 22051 70 60 70 62 22052 63 73 70 63 22053 92 100 100 100 22054 64 55 54 61	22038 82 83 85 86 89 22039 65 56 64 62 58 22040 100 93 92 86 84 22041 78 72 73 79 81 22042 98 100 100 93 94 22043 58 62 67 77 71 22044 96 92 94 100 99 22045 86 87 85 84 85 22046 48 55 46 40 34 22047 56 52 54 47 40 22048 42 44 46 53 62 22049 64 54 49 59 54 22050 50 44 37 29 37 22051 70 60 70 62 67 22052 63 73 70 63 60 22053 92 100 100	22038 82 83 85 86 89 80 22039 65 56 64 62 58 51 22040 100 93 92 86 84 76 22041 78 72 73 79 81 73 22042 98 100 100 93 94 92 22043 58 62 67 77 71 63 22044 96 92 94 100 99 95 22045 86 87 85 84 85 91 22046 48 55 46 40 34 29 22047 56 52 54 47 40 35 22048 42 44 46 53 62 59 22049 64 54 49 59 54 55 22050 50 44 37 29 37 46 22051 70 60 70 62	22038 82 83 85 86 89 80 88 22039 65 56 64 62 58 51 61 22040 100 93 92 86 84 76 82 22041 78 72 73 79 81 73 71 22042 98 100 100 93 94 92 100 22043 58 62 67 77 71 63 64 22044 96 92 94 100 99 95 98 22045 86 87 85 84 85 91 86 22046 48 55 46 40 34 29 37 22047 56 52 54 47 40 35 43 22048 42 44 46 53 62 59 57 22050 50 44 37 29 37 46 53 22051 <	22038 82 83 85 86 89 80 88 95 22039 65 56 64 62 58 51 61 68 22040 100 93 92 86 84 76 82 74 22041 78 72 73 79 81 73 71 77 22042 98 100 100 93 94 92 100 100 22043 58 62 67 77 71 63 64 73 22044 96 92 94 100 99 95 98 92 22045 86 87 85 84 85 91 86 82 22046 48 55 46 40 34 29 37 34 22047 56 52 54 47 40 35 43 44 22048 42 44 46 53 62 59 57 53	22038 82 83 85 86 89 80 88 95 87 22039 65 56 64 62 58 51 61 68 70 22040 100 93 92 86 84 76 82 74 79 22041 78 72 73 79 81 73 71 77 83 22042 98 100 100 93 94 92 100 100 98 22043 58 62 67 77 71 63 64 73 83 22044 96 92 94 100 99 95 98 92 84 22045 86 87 85 84 85 91 86 82 85 22046 48 55 46 40 34 29 37 34 39 22047 56 52 54 47 40 35 43 44 40	22038 82 83 85 86 89 80 88 95 87 93 22039 65 56 64 62 58 51 61 68 70 70 22040 100 93 92 86 84 76 82 74 79 72 22041 78 72 73 79 81 73 71 77 83 92 22042 98 100 100 93 94 92 100 100 98 94 22043 58 62 67 77 71 63 64 73 83 76 22044 96 92 94 100 99 95 98 92 84 84 22045 86 87 85 84 85 91 86 82 85 87 22046 48 55 46 40 34 29 37 34 39 41 22047 56	22038 82 83 85 86 89 80 88 95 87 93 90 22039 65 56 64 62 58 51 61 68 70 70 63 22040 100 93 92 86 84 76 82 74 79 72 79 22041 78 72 73 79 81 73 71 77 83 92 97 22042 98 100 100 93 94 92 100 100 98 94 97 22043 58 62 67 77 71 63 64 73 83 76 86 22044 96 92 94 100 99 95 98 92 84 84 84 22045 86 87 85 84 85 91 86 82 85 87 84 22046 48 55 46 40 34

In [250]: x.head(10)

Out[250]:

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7	Test_8	Test_9	Test_10	Test_11	Te
0	22000	78	87	91	91	88	98	94	100	100	100	100	
1	22001	79	71	81	72	73	68	59	69	59	60	61	
2	22002	66	65	70	74	78	86	87	96	88	82	90	
3	22003	60	58	54	61	54	57	64	62	72	63	72	
4	22004	99	95	96	93	97	89	92	98	91	98	95	
5	22005	41	36	35	28	35	36	27	26	19	22	27	
6	22006	47	50	47	57	62	64	71	75	85	87	85	
7	22007	84	74	70	68	58	59	56	56	64	70	67	
8	22008	74	64	58	57	53	51	47	45	42	43	34	
9	22009	87	81	73	74	71	63	53	45	39	43	46	
												_	

```
In [251]: x.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

```
In [252]: x.columns
```

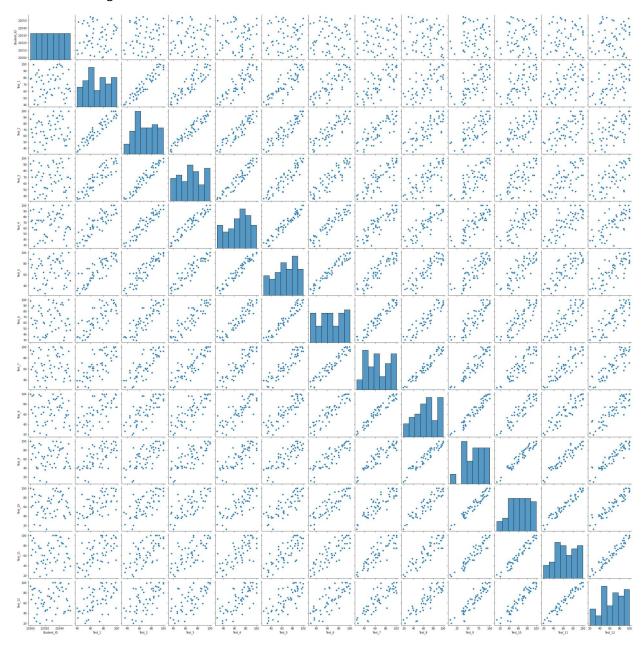
In [253]: x.describe()

Out[253]:

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7
count	56.000000	56.000000	56.000000	56.000000	56.000000	56.000000	56.000000	56.000000
mean	22027.500000	70.750000	69.196429	68.089286	67.446429	67.303571	66.000000	66.160714
std	16.309506	17.009356	17.712266	18.838333	19.807179	20.746890	21.054043	21.427914
min	22000.000000	40.000000	34.000000	35.000000	28.000000	26.000000	29.000000	26.000000
25%	22013.750000	57.750000	55.750000	53.000000	54.500000	53.750000	50.250000	47.000000
50%	22027.500000	70.500000	68.500000	70.000000	71.500000	69.000000	65.500000	64.000000
75%	22041.250000	84.000000	83.250000	85.000000	84.000000	85.250000	83.750000	85.250000
max	22055.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000

In [254]: sns.pairplot(x)

Out[254]: <seaborn.axisgrid.PairGrid at 0x110f6edf1f0>

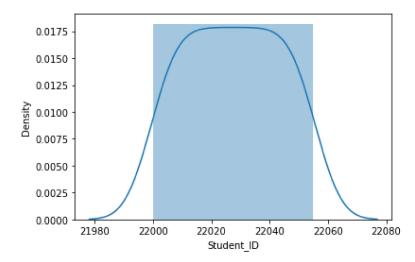


```
In [257]: sns.distplot(x['Student_ID'])
```

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Plea se adapt your code to use either `displot` (a figure-level function with similar flex ibility) or `histplot` (an axes-level function for histograms).

warnings.warn(msg, FutureWarning)

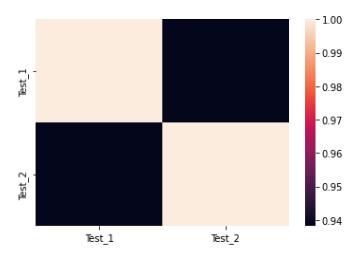
Out[257]: <AxesSubplot:xlabel='Student_ID', ylabel='Density'>



```
In [258]: x1=x[['Test_1','Test_2']]
```

In [259]: sns.heatmap(x1.corr())

Out[259]: <AxesSubplot:>



```
In [262]: a=x1[['Test_1']]
b=x1['Test_2']
```

```
In [263]: from sklearn.model_selection import train_test_split
a_train,a_test, b_train, b_test=train_test_split(a,b,test_size=0.3)
```

```
from sklearn.linear model import LinearRegression
In [264]:
          lr=LinearRegression()
          lr.fit(a train,b train)
Out[264]: LinearRegression()
In [265]:
          print(lr.intercept_)
           1.9724683275231882
In [266]: coeff=pd.DataFrame(lr.coef ,a.columns,columns=['Co-efficient'])
           coeff
Out[266]:
                  Co-efficient
                    0.943151
           Test_1
In [267]: prediction=lr.predict(a test)
          plt.scatter(b_test,prediction)
Out[267]: <matplotlib.collections.PathCollection at 0x110fba53130>
            90
            80
            70
            60
            50
            40
                        50
                               60
                                     70
                                                  90
                                                        100
                  40
In [268]: |lr.score(a_test,b_test)
Out[268]: 0.9236264195366813
In [269]: from sklearn.linear_model import Ridge,Lasso
In [270]: rr=Ridge(alpha=10)
          rr.fit(a_train,b_train)
Out[270]: Ridge(alpha=10)
In [271]: rr.score(a_test,b_test)
Out[271]: 0.9234830317349281
```

```
In [272]: la=Lasso(alpha=10)
          la.fit(a train,b train)
Out[272]: Lasso(alpha=10)
In [273]: la.score(a_test,b_test)
Out[273]: 0.9160705861474879
In [274]: from sklearn.linear model import ElasticNet
          en=ElasticNet()
          en.fit(a train,b train)
Out[274]: ElasticNet()
In [275]: en.coef
Out[275]: array([0.93897178])
In [276]: en.predict(a_test)
Out[276]: array([67.99818858, 71.75407569, 41.70697884, 88.65556767, 96.16734188,
                 53.91361193, 75.5099628 , 80.20482168, 78.32687813, 52.97464016,
                 40.76800706, 73.63201924, 91.472483 , 95.22837011, 49.21875305,
                 43.58492239, 49.21875305])
In [277]: en.intercept_
Out[277]: 2.270164215384739
In [278]: en.score(a test,b test)
Out[278]: 0.9230392412207515
In [279]: from sklearn import metrics
In [280]: |print("Mean Absolute Error",metrics.mean_absolute_error(b_test,prediction))
          Mean Absolute Error 5.060050916038551
In [281]: print("Mean Squared Error", metrics.mean_squared_error(b_test, prediction))
          Mean Squared Error 34.59062523877063
In [282]: print("Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(b_test,prediction))
          Root Mean Squared Error 5.881379535344631
 In [ ]:
```