Freature creation

In [1]:
```
import pandas as pd
df=pd.DataFrame({
    "study hours":[2,4,6,8,10],
    "attendance":[60,70,80,90,95],
    "maths":[40,55,65,75,80],
    "science":[42,58,68,78,88]

})
df
```

Out[1]:

	study hours	attendance	maths	science
0	2	60	40	42
1	4	70	55	58
2	6	80	65	68
3	8	90	75	78
4	10	95	80	88

domain-based features

In [2]:
```
df["total-marks"] =df["maths"] + df["science"]
df
```

Out[2]:

	study hours	attendance	maths	science	total-marks
0	2	60	40	42	82
1	4	70	55	58	113
2	6	80	65	68	133
3	8	90	75	78	153
4	10	95	80	88	168

mathematical feature

In [3]:
```
df["marks_per_hour"] =df["total-marks"]/df["study hours"]
df
```

Out[3]:

	study hours	attendance	maths	science	total-marks	marks_per_hour
0	2	60	40	42	82	41.000000
1	4	70	55	58	113	28.250000
2	6	80	65	68	133	22.166667
3	8	90	75	78	153	19.125000
4	10	95	80	88	168	16.800000

interaction feature

In [4]:
```
df["study_attendance_interaction"] =df["study hours"] + df["attendance"]
df
```

Out[4]:

	study hours	attendance	maths	science	total-marks	marks_per_hour	study_attendance_interaction
0	2	60	40	42	82	41.000000	62
1	4	70	55	58	113	28.250000	74
2	6	80	65	68	133	22.166667	86
3	8	90	75	78	153	19.125000	98
4	10	95	80	88	168	16.800000	105

polynomial feature

In [5]:
```
df["study_hour_squared"] = df["study hours"]
```

In [6]:
```
df
```

Out[6]:

	study hours	attendance	maths	science	total-marks	marks_per_hour	study_attendance_interaction	study_ho
0	2	60	40	42	82	41.000000	62	
1	4	70	55	58	113	28.250000	74	
2	6	80	65	68	133	22.166667	86	
3	8	90	75	78	153	19.125000	98	
4	10	95	80	88	168	16.800000	105	

feature selection

In [7]:
```
x=df.drop("science", axis=1)
y=df["science"]
x
```

Out[7]:

	study hours	attendance	maths	total-marks	marks_per_hour	study_attendance_interaction	study_hour_square
0	2	60	40	82	41.000000	62	
1	4	70	55	113	28.250000	74	
2	6	80	65	133	22.166667	86	
3	8	90	75	153	19.125000	98	
4	10	95	80	168	16.800000	105	

In [8]:
```
y
```

Out[8]:
```
0    42
1    58
2    68
3    78
4    88
Name: science, dtype: int64
```

filter methods (statistical-based)

In [9]:
```
df.corr()["science"].sort_values(ascending=False)
```

Out[9]:
```
science                         1.000000
total-marks                     0.998981
maths                           0.995467
study_attendance_interaction    0.995048
study hours                     0.994309
study_hour_squared              0.994309
attendance                      0.994110
marks_per_hour                 -0.968737
Name: science, dtype: float64
```

selecting important features

In [10]:
```
corr =df.corr()["science"].abs() ## abs antey negative values ni +ve
corr                                    #loki convert chesthadhi
```

Out[10]:
```
study hours                     0.994309
attendance                      0.994110
maths                           0.995467
science                         1.000000
total-marks                     0.998981
marks_per_hour                  0.968737
study_attendance_interaction    0.995048
study_hour_squared              0.994309
Name: science, dtype: float64
```

In [11]:
```
selected_features = corr[corr>0.8].index
selected_features
```

Out[11]:
```
Index(['study hours', 'attendance', 'maths', 'science', 'total-marks',
       'marks_per_hour', 'study_attendance_interaction', 'study_hour_squared'],
      dtype='object')
```

In [ ]:
```

```