Experiment Number : 3

Name: Rushi Daulatkar

Roll no: 53

Aim:

Coefficient of Correlation

In [7]:

```
import numpy as np
```

In [8]:

```
x = [1,2,3,4,5]
y = [2,4,1,3,5]
corr_coeff = np.corrcoef(x,y)[0,1]
print("Correlation coefficiant: ", corr_coeff)
```

```
Correlation coefficiant:  0.49999999999999994
```

Multiple regression

In [14]:

```
# import statsmodels.api as sm
# # X1, X2, X3 are dependent and y independent
# X1 = [1,2,3,4,5]
# X2 = [2,3,4,5,6]
# X3 = [3,4,5,6,7]
# y = [2,4,1,3,5]
# X = np.column_stack((X1, X2, X3))
# X = sm.add_constant(X)
# model = sm.OLS(y, X).fit()
# print(model.summary())
```

> *Executed on [https://sagecell.sagemath.org/ (https://sagecell.sagemath.org/)](https://sagecell.sagemath.org/) for statsmodels*

In [15]:

```
#                             OLS Regression Results
# ==============================================================================
# Dep. Variable:                      y   R-squared:                       0.250
# Model:                            OLS   Adj. R-squared:                  0.000
# Method:                 Least Squares   F-statistic:                     1.000
# Date:                Tue, 26 Mar 2024   Prob (F-statistic):              0.391
# Time:                        09:23:55   Log-Likelihood:                -8.1084
# No. Observations:                   5   AIC:                             20.22
# Df Residuals:                       3   BIC:                             19.44
# Df Model:                           1
# Covariance Type:            nonrobust
# ==============================================================================
#                  coef    std err          t      P>|t|      [0.025      0.975]
# ------------------------------------------------------------------------------
# const          0.3333      0.707      0.471      0.670      -1.917       2.584
# x1            -0.1667      0.866     -0.192      0.860      -2.923       2.589
# x2             0.1667      0.167      1.000      0.391      -0.364       0.697
# x3             0.5000      0.553      0.905      0.432      -1.259       2.259
# ==============================================================================
# Omnibus:                          nan   Durbin-Watson:                   2.533
# Prob(Omnibus):                    nan   Jarque-Bera (JB):                0.361
# Skew:                          -0.408   Prob(JB):                        0.835
# Kurtosis:                       1.967   Cond. No.                     1.26e+17
# ==============================================================================

# Notes:
# [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
# [2] The smallest eigenvalue is 1.77e-32. This might indicate that there are
# strong multicollinearity problems or that the design matrix is singular.
```

Linear regression

In [16]:

```
import scipy.stats as stats
```

In [17]:

```
x_data = np.array([1,2,3,4,5])
y_data = np.array([2,3,5,6,8])
```

In [18]:

```
# perform linear regression
slope, intercept, r_value, p_value, std_err = stats.linregress(x_data, y_data)
```

In [19]:

```
# print regression result
print(f"slope: {slope}")
print(f"intercept: {intercept}")
print(f"R squared: {r_value ** 2}")
```

```
slope: 1.5
intercept: 0.2999999999999998
R squared: 0.9868421052631576
```

Conclusion: