Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

A.Y. 2022-2023

DATA MINING AND WAREHOUSE

AYUSH JAIN

COMPUTER ENGINEERING | TE – B2 | 60004200132

EXPERIMENT – 4

Aim: Implementation of Linear Regression for Single Variate and Multi-variate.

A: Program Single variate using inbuilt functions.

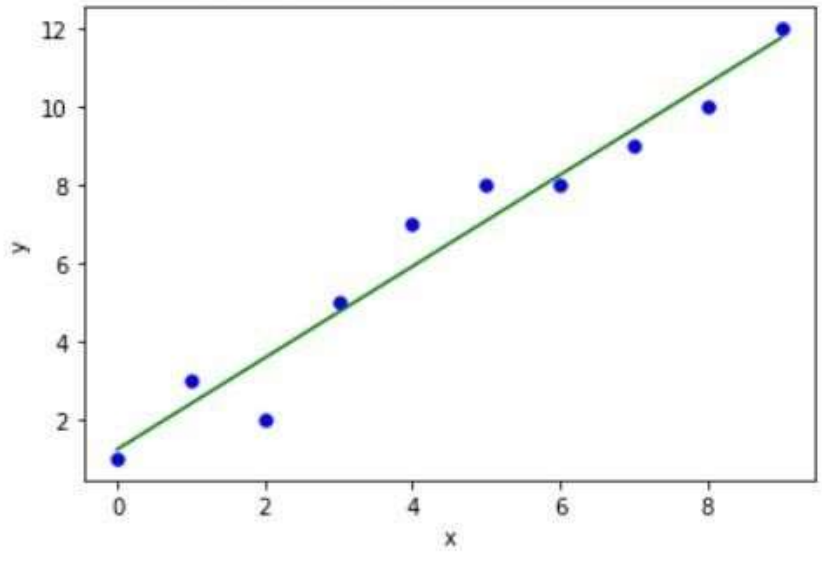
Code:

```
import numpy as np
import matplotlib.pyplot as plt
def estimate_coef(x, y):
n = np.size(x)

# mean of x and y vector m_x = np.mean(x)
m_y = np.mean(y)
# calculating cross-deviation and deviation about x SS_xy = np.sum(y*x) -
n*m_y*m_x
SS_xx = np.sum(x*x) - n*m_x*m_x


# calculating regression coefficients b_1 = SS_xy / SS_xx b_0 = m_y - b_1*m_x


return (b_0, b_1)


def plot_regression_line(x, y, b): # plotting the actual points as scatter plot
plt.scatter(x, y, color = "b", marker = "o", s = 30)


# predicted response vector y_pred = b[0] + b[1]*x

# plotting the regression line plt.plot(x, y_pred, color = "g")
```

```
# putting labels plt.xlabel('x')
plt.ylabel('y')

# function to show plot plt.show()

x   = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
y   = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])

b = estimate_coef(x, y) print("Estimated coefficients:\nb_0 = {} \ \nb_1 =
{}".format(b[0], b[1]))

plot_regression_line(x, y, b)
```

Output: Predict for unseen samples

```
Estimated coefficients:
b_0 = 1.2363636363636363
b_1 = 1.1696969696969697
```



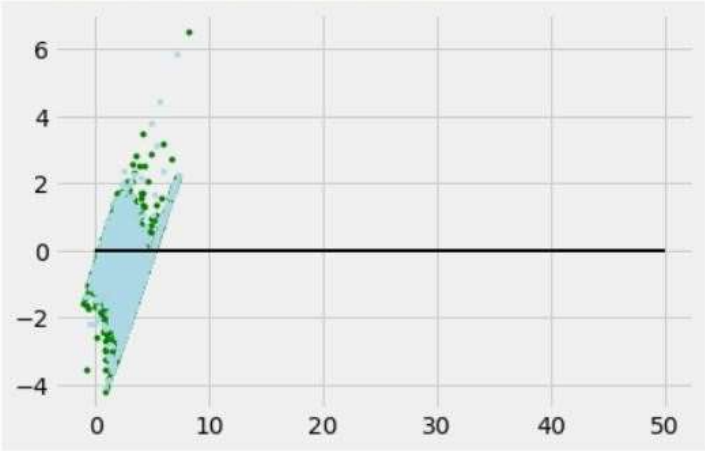
Part B: Program Multi variate using inbuilt functions.

Code:

```
california = datasets.fetch_california_housing(return_X_y=False)
X = california.data y = california.taret

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
random_state=1)


# create linear regression object
reg = linear_model.LinearRegression()


reg.fit(X_train, y_train)


# regression coefficients print('Coefficients: ', reg.coef_)

# variance score: 1 means perfect prediction print('Variance score:
{}'.format(reg.score(X_test, y_test)))

## plotting residual errors in training data plt.scatter(reg.predict(X_train),
reg.predict(X_train) - y_train, color = "green", s = 10, label = 'Training set')

## plotting residual errors in test data plt.scatter(reg.predict(X_test),
reg.predict(X_test) - y_test, color = "lightblue", s = 10, label = 'Testing set')

## plotting line for zero residual error
plt.hlines(y = 0, xmin = 0, xmax = 50, linewidth = 2)

plt.show()
```

Output: Predict for unseen samples

```
Coefficients:  [ 4.36676927e-01  9.58588561e-03 -9.61859974e-02  5.77800722e-01
 -4.33084487e-06 -3.13805195e-03 -4.22347516e-01 -4.34869554e-01]
Variance score: 0.6068373239208641
```



Conclusion: Implemented Linear and multiple simple regression on dataset to predict the values and plotted the regression curve for the same.