**Program - 1**

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_squared\_error, r2\_score

from sklearn.linear\_model import LinearRegression

np.random.seed(0)

x = 2 - 3 \* np.random.normal(0, 1, 20)

y = x - 2 \* (x \*\* 2) + 0.5 \* (x \*\* 3) + np.random.normal(-3, 3, 20)

# transforming the data to include another axis

x = x[:, np.newaxis]

y = y[:, np.newaxis]

model = LinearRegression()

model.fit(x, y)

y\_pred = model.predict(x)

plt.scatter(x, y, s=10)

plt.plot(x, y\_pred, color='r')

plt.show()

rmse = np.sqrt(mean\_squared\_error(y,y\_pred))

r2 = r2\_score(y,y\_pred)

print(rmse)

print(r2)

**Program-2**

import operator

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

from sklearn.preprocessing import PolynomialFeatures

np.random.seed(0)

x = 2 - 3 \* np.random.normal(0, 1, 20)

y = x - 2 \* (x \*\* 2) + 0.5 \* (x \*\* 3) + np.random.normal(-3, 3, 20)

# transforming the data to include another axis

x = x[:, np.newaxis]

y = y[:, np.newaxis]

polynomial\_features= PolynomialFeatures(degree=2)

x\_poly = polynomial\_features.fit\_transform(x)

model = LinearRegression()

model.fit(x\_poly, y)

y\_poly\_pred = model.predict(x\_poly)

rmse = np.sqrt(mean\_squared\_error(y,y\_poly\_pred))

r2 = r2\_score(y,y\_poly\_pred)

print(rmse)

print(r2)

plt.scatter(x, y, s=10)

# sort the values of x before line plot

sort\_axis = operator.itemgetter(0)

sorted\_zip = sorted(zip(x,y\_poly\_pred), key=sort\_axis)

x, y\_poly\_pred = zip(\*sorted\_zip)

plt.plot(x, y\_poly\_pred, color='m')

plt.show()