Implementation-of-Linear-Regression-Using-Gradient-Descent

AIM:

To write a program to predict the profit of a city using the linear regression model with gradient descent.

Equipments Required:

1. Hardware – PCs
2. Anaconda – Python 3.7 Installation / Jupyter notebook

Algorithm

1. Import required libraries in python for Gradient Design.
2. Upload the dataset and check any null value using .isnull() function.
3. Declare the default values for linear regression.
4. Calculate the loss usinng Mean Square Error.
5. Predict the value of y.
6. Plot the graph respect to hours and scores using scatter plot function.

Program:

```
Program to implement the linear regression using gradient descent.
Developed by: Anish Raj P
RegisterNumber:  212222230010

import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler

def linear_regression(X1,y,learning_rate=0.01,num_iters=1000):
    X=np.c_[np.ones(len(X1)),X1]
    theta = np.zeros(X.shape[1]).reshape(-1,1)
    for _ in range(num_iters):
        predictions = (X).dot(theta).reshape(-1,1)
        errors = (predictions-y).reshape(-1,1)
        theta-=learning_rate*(1/len(X1))*X.T.dot(errors)
    return theta

data=pd.read_csv('50_Startups.csv',header=None)
data.head()
X = (data.iloc[1:,:-2].values)
print(X)

X1=X.astype(float)
scaler = StandardScaler()
y=(data.iloc[1:,-1].values).reshape(-1,1)
print(y)

X1_Scaled=scaler.fit_transform(X1)
Y1_Scaled=scaler.fit_transform(y)
print(X1_Scaled)
print(Y1_Scaled)

theta = linear_regression(X1_Scaled,Y1_Scaled)

new_data=np.array([165349.2,136897.8,471784.1]).reshape(-1,1)
new_Scaled = scaler.fit_transform(new_data)
prediction = np.dot(np.append(1,new_Scaled),theta)
prediction = prediction.reshape(-1,1)
pre=scaler.inverse_transform(prediction)
print(f"Predicted Value:{pre}")
```

Output:

X values

```
print(X)
```

```
[['165349.2' '136897.8' '471784.1']
 ['162597.7' '151377.59' '443898.53']
 ['153441.51' '101145.55' '407934.54']
 ['144372.41' '118671.85' '383199.62']
 ['142107.34' '91391.77' '366168.42']
 ['131876.9' '99814.71' '362861.36']
```

y values

```
print(y)
```

```
[['192261.83']
 ['191792.06']
 ['191050.39']
 ['182901.99']
 ['166187.94']
 ['156991.12']
 ['156122.51']
```

X Scaled values

```
print(X1_Scaled)
```

```
[[ 2.01641149e+00  5.60752915e-01  2.15394309e+00]
 [ 1.95586034e+00  1.08280658e+00  1.92360040e+00]
 [ 1.75436374e+00 -7.28257028e-01  1.62652767e+00]
 [ 1.55478369e+00 -9.63646307e-02  1.42221024e+00]
 [ 1.50493720e+00 -1.07991935e+00  1.28152771e+00]
 [ 1.27980001e+00 -7.76239071e-01  1.25421046e+00]
 [ 1.34006641e+00  9.32147208e-01 -6.88149930e-01]
```

y Scaled values

```
print(Y1_Scaled)
```

```
[[ 2.01120333]
 [ 1.99942997]
 [ 1.98084225]
 [ 1.77662724]
 [ 1.35774012]
 [ 1.12724963]
 [ 1.10548055]
 [ 1.09620987]
```

Predicted value

```
Predicted value: [[203866.53076613]]
```

Result:

Thus the program to implement the linear regression using gradient descent is written and verified using python programming.