In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv('cars24-car-price-clean.csv')
df.head()
```

Out[2]:

	selling_price	year	km_driven	mileage	engine	max_power	make	model	transmissio
0	1.20	2012.0	120000	19.70	796.0	46.30	Maruti	Alto Std	
1	5.50	2016.0	20000	18.90	1197.0	82.00	Hyundai	Grand i10 Asta	
2	2.15	2010.0	60000	17.00	1197.0	80.00	Hyundai	i20 Asta	
3	2.26	2012.0	37000	20.92	998.0	67.10	Maruti	Alto K10 2010-2014 VXI	
4	5.70	2015.0	30000	22.77	1498.0	98.59	Ford	Ecosport 2015-2021 1.5 TDCi Titanium BSIV	

In [3]:

```
X = df['max_power'].values
Y = df['selling_price'].values
```

In [6]:

```
X = X.reshape(-1, 1)
Y = Y.reshape(-1, 1)
```

In [7]:

```
X.shape
```

Out[7]:

```
(19820, 1)
```

In [8]:

```
Y.shape
```

Out[8]:

```
(19820, 1)
```

In [89]:

```
mean = np.array([4.0, 5.0])
cov = np.array([[1.0, 0.95], [0.95, 1.2]])
data = np.random.multivariate_normal(mean, cov, 5000)
```

In [90]:

```
data.shape
```

Out[90]:

```
(5000, 2)
```

In [91]:

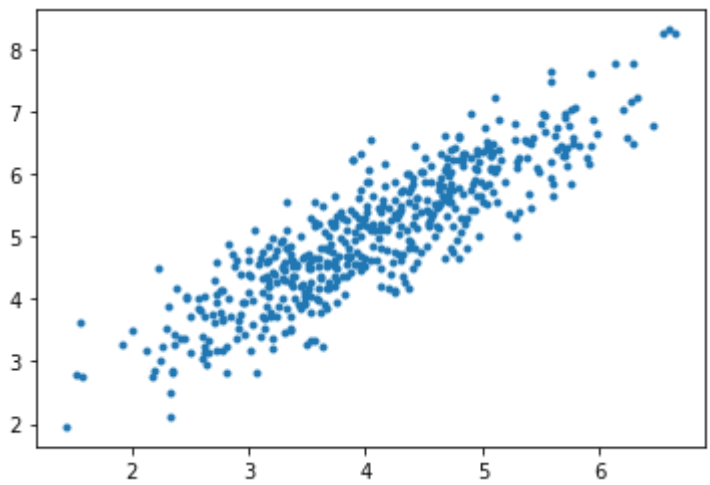
```
data[:5]
```

Out[91]:

```
array([[4.24811459, 5.87254359],
       [4.71645054, 5.74045397],
       [3.24779048, 4.28303651],
       [5.0402471 , 6.42692803],
       [5.03589483, 5.53453865]])
```

In [92]:

```
plt.scatter(data[:500, 0], data[:500, 1], marker = '.')
plt.show()
```



In [93]:

```
X = data[: ,:-1]
y = data[:, -1].reshape(-1,1)
```

In [94]:

```
X.shape,  y.shape
```

Out[94]:

```
((5000, 1), (5000, 1))
```

In [ ]:

```

```

In [95]:

```
def predict(X, weights):
    return np.dot(X, weights)
```

In [96]:

```
def error(X, Y, weights):

    Y_hat = predict(X, weights)
    err = np.mean((Y - Y_hat)**2)

    return err
```

In [97]:

```
def gradient(X, Y, weights):

    n = X.shape[0]

    Y_hat = predict(X, weights)
    grad = np.dot(X.T, Y - Y_hat)

    grad = (grad*-2)/n

    return grad
```

In [98]:

```
def create_mini_batches(X, y, batch_size):

    mini_batches = []
    data = np.hstack((X, y))

    np.random.shuffle(data)
    minibatch_count = data.shape[0] // batch_size

    for i in range(minibatch_count):
        mini_batch = data[i * batch_size: (i + 1)*batch_size, :]
        X_minibatch = mini_batch[:, :-1]
        Y_minibatch = mini_batch[:, -1].reshape((-1, 1))
        mini_batches.append((X_minibatch, Y_minibatch))

    return mini_batches
```

In [ ]:

```
```

In [122]:

```
def gradientDescent(X, y, learning_rate = 0.01, batch_size = 32):
    weight = np.zeros((X.shape[1], 1))
    error_list = []
    epochs = 5

    for itr in range(epochs):

        mini_batches = create_mini_batches(X, y, batch_size)

        for mini_batch in mini_batches:
            X_mini, y_mini = mini_batch
            weight = weight - learning_rate * gradient(X_mini, y_mini, weight)
            error_list.append(error(X_mini, y_mini, weight))

    return weight, error_list
```

In [123]:

```
mu = X.mean()
sig = X.std()

X_new = (X-mu)/sig
```

In [124]:

```
X_new = np.hstack((np.ones((X_new.shape[0],1)),X_new))
X_new[:5]
```

Out[124]:

```
array([[ 1.        ,  0.22008567],
       [ 1.        ,  0.68621217],
       [ 1.        , -0.77551925],
       [ 1.        ,  1.00848116],
       [ 1.        ,  1.00414942]])
```

In [125]:

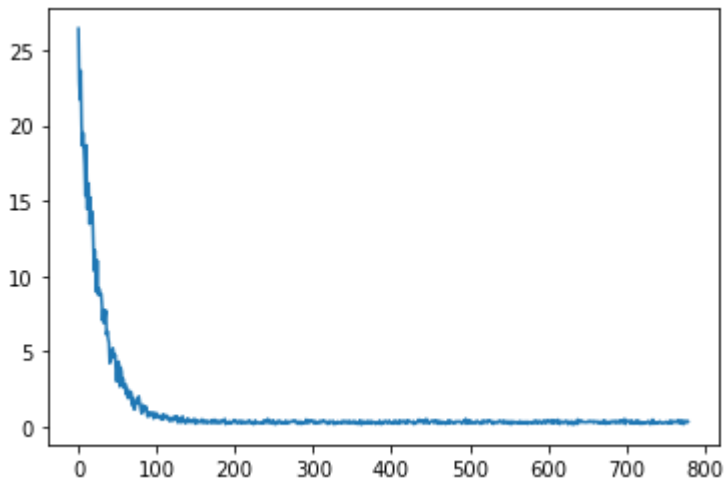
```
weights, error_list = gradientDescent(X_new, y)
```

In [126]:

```
plt.plot(error_list)
```

Out[126]:

```
[<matplotlib.lines.Line2D at 0x7fa0880ff940>]
```



In [88]:

```
weights
```

Out[88]:

```
array([[7.39094507],
       [6.69375882]])
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```