LAB 5 Task

linear regression

In [2]:

```
import numpy as np
```

In [3]:

```
# inputs
inputs = np.array([[73, 67, 43],
                   [91, 88, 64],
                   [87, 134, 58],
                   [102, 43, 37],
                   [69, 96, 70]], dtype='float32')
```

In [4]:

```
# target
targets = np.array([[56],
                    [81],
                    [119],
                    [22],
                    [103]], dtype='float32')
```

In [5]:

```
# initializing theta(weight) with random value
theta = np.random.randn(1,inputs.shape[1])
print("theta:",theta)
```

```
theta: [[-0.46308356 -0.71356647 -1.30446745]]
```

In [6]:

```
# initializing bias
theta0 = np.random.randn(1,1)
print("Theta0(bias):",theta0)
```

```
Theta0(bias): [[1.16649716]]
```

In [7]:

```
def linearRegressionModel(x):
    return x @ theta.T + theta0
```

In [8]:

```
def mse(pred,tar):
    diff=pred-tar
    j=np.sum(diff*diff)
    j=j/pred.shape[0]
    return j
```

In [9]:

```
def grad(x):
    h = linearRegressionModel(x[:,1:])
    diff = h - targets
    diff_of_j=np.sum(diff * x,axis = 0)/x.shape[0]
    return diff_of_j
```

In [10]:

```
predicted = linearRegressionModel(inputs)
```

In [11]:

```
print("predicted:\n",predicted)
```

```
predicted:
 [[-136.5396566 ]
 [-187.25387302]
 [-210.39879189]
 [-125.0166797 ]
 [-190.60137125]]
```

In [12]:

```
print("target:\n",targets)
```

```
target:
 [[ 56.]
 [ 81.]
 [119.]
 [ 22.]
 [103.]]
```

In [13]:

```
loss=mse(predicted,targets)
print("loss: ",loss)
```

```
loss:  65070.178633082796
```

In [14]:

```
# creating new input set for calculating d/d@(J) for theta0 - thetad
x0=np.zeros((inputs.shape[0],1),dtype='float32')
a=np.concatenate((x0,inputs),axis=1)
```

In [15]:

```
# updating all theta values
gradient=grad(a)
theta=theta-gradient[1:]*1e-5
theta0=theta0-gradient[0]*1e-5
```

In [16]:

```
print("theta:",theta)
print("theta0:",theta0)
```

```
theta: [[-0.25832678 -0.4832597  -1.16337886]]
theta0: [[1.16649716]]
```

In [17]:

```
prediction=linearRegressionModel(inputs)
```

In [18]:

```
print("prediction:\n",prediction)
```

```
prediction:
 [[-100.09504889]
 [-139.32434073]
 [-153.54070686]
 [ -89.00801949]
 [-144.48750233]]
```

In [19]:

```
print("target:\n",targets)
```

```
target:
 [[ 56.]
 [ 81.]
 [119.]
 [ 22.]
 [103.]]
```

In [20]:

```
loss=mse(prediction,targets)
print("loss:",loss)
```

```
loss: 44151.9521007608
```

In [21]:

```
for i in range(0,10000):
    x0=np.ones((inputs.shape[0],1),dtype='float32')
    a=np.concatenate((x0,inputs),axis=1)
    gradient=grad(a)
    theta=theta-gradient[1:]*1e-5
    theta0=theta0-gradient[0]*1e-5
    prediction=linearRegressionModel(inputs)
    loss=mse(prediction,targets)
```

In [22]:

```
print("prediction:\n",prediction)
```

```
prediction:
 [[ 57.36182347]
 [ 82.08310784]
 [118.66211444]
 [ 21.05704079]
 [101.95089931]]
```

In [23]:

```
print("target:\n",targets)
```

```
target:
 [[ 56.]
 [ 81.]
 [119.]
 [ 22.]
 [103.]]
```

In [24]:

```
loss=mse(prediction,targets)
print("loss:",loss)
```

```
loss: 1.0263273476377643
```