LAB 5 Task

linear regression

In [1]:

```
import numpy as np
```

In [2]:

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

In [3]:

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

In [7]:

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

```
theta: [[ 0.33385692  0.52783641 -2.38307247]]
```

In [8]:

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

```
Theta0(bias): [[1.2508136]]
```

In [9]:

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

In [10]:

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

In [11]:

```
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 [12]:

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

In [14]:

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

```
predicted:
 [[-41.4847084 ]
 [-74.43524124]
 [-37.19175932]
 [-30.17249672]
 [-91.85583702]]
```

In [15]:

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

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

In [16]:

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

```
loss:  19750.002980959733
```

In [17]:

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

In [18]:

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

In [19]:

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

```
theta: [[ 0.44108956  0.65201451 -2.30553425]]
theta0: [[1.25212588]]
```

In [20]:

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

In [21]:

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

```
prediction:
 [[-22.00133697]
 [-48.78563949]
 [ -6.72412485]
 [-11.0248824 ]
 [-67.10669926]]
```

In [22]:

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

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

In [23]:

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

```
loss: 13752.401669843257
```

In [24]:

```
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 [25]:

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

```
prediction:
 [[ 57.3766383 ]
 [ 82.07293194]
 [118.66171691]
 [ 21.05574675]
 [101.95126627]]
```

In [26]:

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

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

In [27]:

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

```
loss: 1.0304416128090217
```