**Momentum Based Gradient Descent**

def do\_momentum\_gradient\_descent():

w,b,eta,max\_epochs = -2, -2, 1.0,10

prev\_v\_w,prev\_v\_b,gamma=0,0,0.9

for i in range(max\_epochs):

dw,db=0,0

for x,y in zip(X,Y):

dw+=grad\_w(w,b,x,y)

db+=grad\_b(w,b,x,y)

v\_w=gamma\*prev\_v\_w+eta\*dw

v\_b=gamma\*prev\_v\_b+eta\*db

w=w-v\_w

b=b-v\_b

prev\_v\_w=v\_w

prev\_v\_b=v\_b

print(error(w,b))

In the regions having gentle slopes, momentum based gradient descent is able to take large steps because the momentum carries it along and chances of escaping from min valley is more

 Gradient Descent with Momentum considers the past gradients to smooth out the update. It computes an exponentially weighted average of your gradients, and then use that gradient to update your weights instead. It works faster than the standard gradient descent algorithm.

**Nesterov Accelerated Gradient Descent**

def do\_nestrove\_accelerated\_gradient\_descent():

w,b,eta,max\_epochs = -2, -2, 1.0,10

prev\_v\_w,prev\_v\_b,gamma=0,0,0.9

for i in range(max\_epochs):

dw,db=0,0

for x,y in zip(X,Y):

dw+=grad\_w(w,b,x,y)

db+=grad\_b(w,b,x,y)

v\_w=gamma\*prev\_v\_w+eta\*dw

v\_b=gamma\*prev\_v\_b+eta\*db

w=w-v\_w

b=b-v\_b

prev\_v\_w=v\_w

prev\_v\_b=v\_b

print(error(w,b))

In nesterev oscillations are smaller and the chances of escaping the minima valley also smaller and chances are also less to cross our region