Exercise 2

Online Learning

Frist input the data as before:

wdbc = read.csv('C:/Users/Sputnik/Desktop/Statistical Models for Big Data/wdbc.csv', header=FALSE, row.names = 1)

ya = as.character(wdbc[,1])

y = rep(0,length(ya))

y[which(ya=='M')] = 1

X = as.matrix(data[,2:11])

X = scale(X)

X = cbind(rep(1,length(ya)),X)

C)

The function for stochastic gradient descent is:

sgd=function(y,X,BETA0,m=1,iter,alpha)

{

p=1

N=500

mob=matrix(0,iter,p)

mob[1,]=BETA0

loglik=rep(0,iter)

distance=rep(0,iter)

for(ii in 2:iter)

{

# randomize

ind=sample(1:N,1)

yran=y[ind]

Xran=X[ind]

mob[ii,]=mob[ii-1,]-alpha\*gradnorm(yran,Xran,mob[ii-1,])

distance[ii] = dist(mob[ii,]-mob[ii-1,])

loglik[ii] = logliknorm(y,X,mob[ii,])

}

return(list(mob=mob,loglik=loglik,dist=distance))

}

And then:

fit2 = sgd(y,X,BETA0,m,iter,alpha)

plot(fit2$mob[1:iter,],type='l')

abline(h=1,lty=2,col=2)

plot(fit2$loglik[1:iter],type='l')

I set m=1, iter=1000, alpha=0.1. Then tried different value of beta0:

|  |  |  |  |
| --- | --- | --- | --- |
| Beta0 | | Plot of distance | Plot of loglike |
| 0 |  | |  |
| -2 |  | |  |
| -10 |  | |  |

From these plots, we can see when beta0=0, there is some noise; when beta0=-2, it converges but has some noisy in some degree; when beta0=10, it looks nice and converges.

D)

We have a new function for step size by the Robbins-Monro rule:

stepsizeRM=function(C,t,alpha)

{

C\*(t+1)^(-alpha) # Set t0 be one here

}

I set beta0=-2 and iter=200 and play with the values of alpha and C:

When alpha=1

|  |  |  |
| --- | --- | --- |
| C | Plot of distance | Plot of loglike |
| 0.01 |  |  |
| 0.1 |  |  |
| 1 |  |  |
| 10 |  |  |
| 100 |  |  |

From these plots, as C increases from 0.01 to 10, the two plots converge to lower values (55000, 30000, 5000, 0) and converge faster. When C=100, the shapes are weird and I don’t know how to explain them.

When C=1:

|  |  |  |
| --- | --- | --- |
| alpha | Plot of distance | Plot of loglike |
| 0.5 |  |  |
| 0.6 |  |  |
| 0.7 |  |  |
| 0.8 |  |  |
| 0.9 |  |  |
| 1 |  |  |

We can see that changing the values of alpha does not have much impact on distance. When alpha is small (less than 0.6), the plot of loglike descent vibrates and does not converge apparently.

E)

We update the average in each iteration:

sumb[ii,]=(sumb[ii-1,]\*(ii-1)+mob[ii,])/ii # Update the average

where sumb is the matrix used to store the average of beta.

In order to know how the averaging approach changes the performance, I did the same as what I did in section C by trying different values of beta0.

|  |  |  |
| --- | --- | --- |
| Beta0 | Plot of distance | Plot of loglike |
| 0 |  |  |
| -2 |  |  |
| -10 |  |  |

After comparing the plots from section C, I found that the averaging algorithm did not improve the plots of log-likelihood, but it did improve the performance of distance. Especially when beta0=0, the plot finally converges and does not vibrate so much for all values of beta0.