Simple Linear Regression

Source code

clc;

close all;

%training data

x=[2 2.5 3 3.5 4 4.5 5 5.5 6];

y=[5.1 6.1 6.9 7.8 9.2 9.9 11.5 12 12.8];

a=0.01; %learning rate

m=9;

i=1:9;

t0=0;

t1=1;

hyp=t0+t1\*x(i);

cf=zeros(1500,1);

for num=1:1500

hyp=t0+t1\*x(i);

cf(num)=(0.5\*m).\*sum((hyp(i)-y(i)).^2); %cost function

%partial diff

jt0=sum(hyp(i)-y(i));

jt1=sum((hyp(i)-y(i)).\*x(i));

t0=t0-a\*jt0;

t1=t1-a\*jt1;

plot(0:49,cf(1:50))

end

t0

t1

xlabel('Number of iterations')

ylabel('Cost J')

when a=0.01; t0=1.0600,t1=1.9933

A close up of a mans face

Description generated with high confidence

When a=0.1;t0=NaN,t1=NaN

A screenshot of a cell phone

Description generated with high confidence

When a=1; t0=NaN,t1=NaN

A screenshot of a cell phone

Description generated with high confidence

Observations:

-cost function does not converge when the value of learning rate is changed.

Multiple Linear Regression

Source code:

%x=[2 70;3 30;4 80;4 20;3 50;7 10;5 50;3 90;2 20]; %notscaled

y=[79.4 41.5 97.5 36.1 63.2 39.5 69.8 103.5 29.5]';

x=[2 0.75;3 0.25;4 0.825;4 0.125;3 0.5;7 0;5 0.5;3 1;2 0.125]; %scaled

m=9;

x=[ones(9,1) x];

z=x';

a=0.1;

t=ones(size(x(1,:)))';

j=zeros(1500,1);

for num=1:1500;

hyp=x\*t;

j(num) = (0.5/m) .\* sum((hyp - y).\*(hyp-y));

g=(1/m).\*z\*((hyp)-y);

t=t-a\*g;

plot(0:49,j(1:50))

end

t

xlabel('number of iterations')

ylabel('cost j')

when there is no scaling; t0=NaN; t1=NaN; t2=NaN

A screenshot of a cell phone

Description generated with high confidence

When scaling is done, t0=9.315;t1=4.22;t2=82.613

A close up of a map

Description generated with high confidence