Learning From Data

Homework II Report

İpek Erdoğan

150130102

Question 1)

The added/changed code statement	Explanation	State
prev_grad_2=[0,0]	This is for clear initialization of prev_grad_2 variable in each gradient calculation	A
for i in range(len(X)):	This for loop is used to provide addition operation in the gradient calculation	A
Xw=-(np.dot(X[i],w))	This is basically multiplication of x and w vectors where X[i]=[1,x] and w=[b,w] and the result is wx+b. There is a – before the product as we will use this multiplication as a power of e to provide a sigmoid below.	A
prev_grad=((1/(1+math.exp(Xw)))-y[i]) * (1/(1+math.exp(Xw))) * (1-(1/(1+math.exp(Xw))))	This is the first 3 elements of the gradient calculation. This equation's result is scalar. That's why I preferred to calculate it separate.	A
prev_grad_2= [prev_grad*X[i][0]+prev_grad_2[0], prev_grad*X[i][1]+prev_grad_2[1]]	Now, we multiply the scalar result which we had above with the X vector and iteratively add values to the previous values.	A
grad=[2*prev_grad_2[0],2*prev_grad_2[1]]	Finally, we multiply the pre-result with 2 to get the final result.	A
w = [w[0]-alpha*grad[0],w[1]-alpha*grad[1]]	To do the update operation as a vector operation, we needed to correct this line as here.	C

*The bold lines are the ones in the for loop.

*In "State" column, A is for Added, C is for Changed.

Question 2)

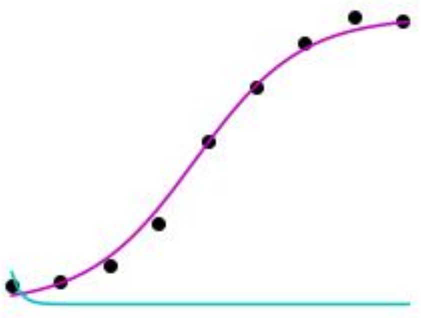
The added/changed code statement	Explanation	State
prev_grad_2=[0,0]	This is for clear initialization of prev_grad_2 variable in each gradient calculation	A
for i in range(len(X)):	This for loop is used to provide addition operation in the gradient calculation	A
Xw=-(np.dot(X[i],w))	This is basically multiplication of x and w vectors where X[i]=[1,x] and w=[b,w] and the result is wx+b. There is a – before the product as we will use this multiplication as a power of e to provide a sigmoid below.	A
prev_grad=((1/(1+math.exp(Xw)))-y[i]) * (1/(1+math.exp(Xw))) * (1-(1/(1+math.exp(Xw))))	This is the first 3 elements of the gradient calculation. This equation’s result is scalar. That’s why I preferred to calculate it separate.	A
prev_grad_2= [prev_grad*X[i][0]+prev_grad_2[0], prev_grad*X[i][1]+prev_grad_2[1] +2*(lamb*w[1])]	Now, we multiply the scalar result which we had above with the X vector and iteratively add values to the previous values. There is a difference between Q1 and Q2 here. As you can see, we add a 2*(lamb*w[1]) value here to the second element of the prev_grad_2. This is for convexifying our loss functions to prevent its stuck in flat places.	A
grad=[2*prev_grad_2[0],2*prev_grad_2[1]]	Finally, we multiply the pre-result with 2 to get the final result.	A
w = [w[0]-alpha*grad[0],w[1]-alpha*grad[1]]	To do the update operation as a vector operation, we needed to correct this line as here.	C

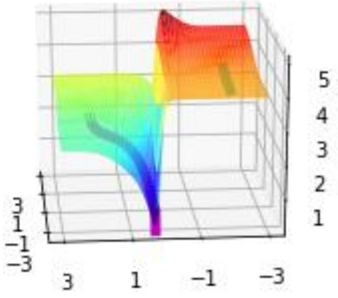
*The bold lines are the ones in the for loop.

*In “State” column, A is for Added, C is for Changed.

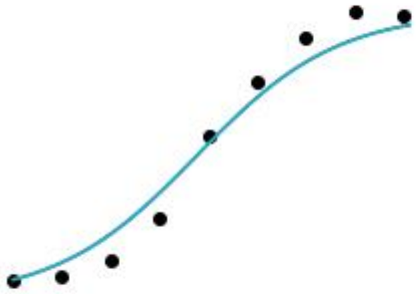
Screenshots

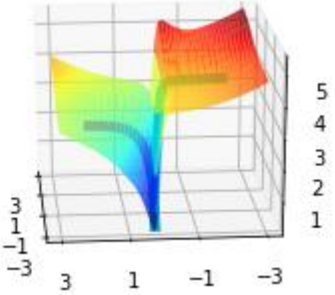
Q1:





Q2:




5
4
3
2
1
3
1
−1
−3
3
1
−1
−3