1.

1. ϴ𝑡+1 is defined to be the vector parameter where the algorithm makes its 𝑡𝑡ℎ error. There for, because t starts at 1, ϴ1 = 0

1. the update of the 𝑡𝑡ℎ error ϴ𝑡+1 = ϴ𝑡 + 𝑦𝑘𝑥𝑘 by the definition of the algorithm. We can add another parameter vector ϴ∗ to each side and that will keep the equality the same. We are thereby able to write the equation as ϴ𝑡+1 ∙ ϴ∗ = (ϴ𝑡 + 𝑦𝑘𝑥𝑘) ∙ ϴ∗
2. From (7) we have ϴ𝑡+1 ∙ ϴ∗ = ϴ𝑡 ∙ ϴ∗ + 𝑦𝑘𝑥𝑘 ∙ ϴ∗. We are told that and some scalar γ > 0 such that for all i = 1, ..., n, 𝑦𝑘𝑥𝑘 ∙ ϴ∗ ≥ γ By substituting γ for 𝑦𝑘𝑥𝑘 ∙ ϴ∗ we can see that:

ϴ𝑡 ∙ ϴ∗ + 𝑦𝑘𝑥𝑘 ∙ ϴ∗ ≥ ϴ𝑡 ∙ ϴ∗ + γ (8)

1. We can solve this proof by induction:

Assume that for t, ϴ𝑡 ∙ ϴ∗ = (𝑡 − 1)γ

Then ϴ𝑡+1 ∙ ϴ∗ = ϴ𝑡 ∙ ϴ∗+ γ = (𝑡 − 1)𝛾 + 𝛾 = t𝛾

1. Because ||||ϴ𝑡+1|| × ||ϴ∗|| ≥ ϴ𝑡+1 ∙ ϴ∗ 𝑎𝑛𝑑 from the explanation from D we saw , ϴ𝑡+1 ∙ ϴ∗ = ϴ𝑡 ∙ ϴ∗+ γ = (𝑡 − 1)𝛾 + 𝛾 = t𝛾

So we have ||ϴ𝑡+1|| ≥ 𝑡γ

1. 𝑦𝑡2 = 1 𝑏𝑦 𝑎𝑠𝑠𝑢𝑝𝑡𝑖𝑜𝑛𝑠 𝑜𝑓 𝑡ℎ𝑒 𝑡ℎ𝑒𝑜𝑟𝑜𝑚 𝑦𝑡2||𝑥𝑡||2 = ||𝑥𝑡||2≤ 𝑅2

Also, 2𝑦𝑘𝑥𝑘 ∙ ϴ𝑡 ≤ 0 because the parameter vector

ϴ𝑡 𝑔𝑎𝑣𝑒 𝑎𝑛 𝑒𝑟𝑟𝑜𝑟 𝑜𝑛 𝑡ℎ𝑒 𝑡𝑡ℎ 𝑒𝑥𝑎𝑚𝑝𝑙𝑒

So ||ϴ𝑡+1||2 = ||ϴ𝑡||2 + 𝑦𝑡2||𝑥𝑡||2 + 2𝑦𝑘𝑥𝑘 ∙ ϴ𝑡 ≤ ||ϴ𝑡||2 + 𝑅2

1. Combining equations (10) and (14) gives us the following equation:

t2𝛾2 ≤ ||ϴ𝑡+1||2 ≤ 𝑡𝑅2

Which can be simplified to be t ≤ 𝑅2/𝛾2

2.

1. In 15(a), the slack variable ξ(i) can be substituted with the loss hinge function s.t

1) yi(θ · xi + θ0) ≥ 1 – ξi 2) ξi ≥ 0, for i = 1, ..., n

by definition of the lhinge function we see that

1. yi(θ · xi + θ0) ≥ 1 – lhinge(yi(θ · xi + θ0))
2. lhinge(yi(θ · xi + θ0)) ≥ 0, for i = 1, ..., n

2.

### STUDENT: Start of code ### def hinge\_loss\_smooth(t):

if (t<=0):

return ((1/2)-t)

elif(t>0 and t<1):

return (1/2)\*(pow((1-t),2)) else: return (0)

def hinge\_loss(t):

if (t<1): return (1-t) else : return (0)

x = []

sh = [0]\*500 h = [0]\*500 for i in range(500): t = (i-250)/50 x.append(t) sh[i]=(hinge\_loss\_smooth(t))

h[i]=(hinge\_loss(t))

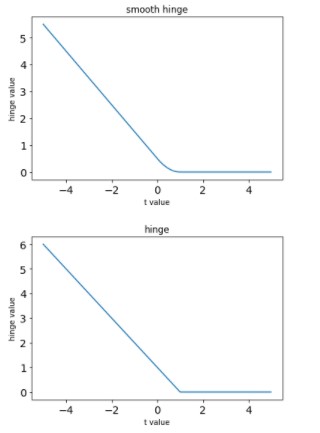
plt.plot(x,sh) plt.title('smooth hinge') plt.ylabel('hinge value') plt.xlabel('t value')

plt.show()

plt.plot(x,h) plt.title('hinge') plt.ylabel('hinge value') plt.xlabel('t value')

plt.show()

### End of code ###



−1 𝑖𝑓 𝑡 ≤ 0

1. Lsmooth-hinge(𝑡)𝑑𝑡 = {1-t if 0 < t < 1}

0 if t ≥ 1

1. t = (yi(xi ∙ 𝜃 + 𝜃0))

−yi(xi) 𝑖𝑓 𝑡 ≤ 0

Lsmooth-hinge(yi(xi ∙ 𝜃 + 𝜃0))d𝜃 = {1- (yi(xi ∙ 𝜃 + 𝜃0)) if 0 < t < 1 }

0 if t ≥ 1

−yi 𝑖𝑓 𝑡 ≤ 0

Lsmooth-hinge(yi(xi ∙ 𝜃 + 𝜃0))d𝜃0 = {1- (yi(xi ∙ 𝜃 + 𝜃0)) if 0 < t < 1 }

0 if t ≥ 1

5.

## STUDENT: Start of code ###

score\_matrix = (feature\_matrix.dot(theta) + theta0) \* labels

feature\_matrix1=np.array([np.zeros(4500)]\*2500) feature\_matrix2=np.array([np.zeros(4500)]\*2500)

for i in range(len(score\_matrix)): if score\_matrix[i]<=0:

feature\_matrix1[i] = feature\_matrix[i] elif score\_matrix[i]<1:

feature\_matrix2[i] = feature\_matrix[i]

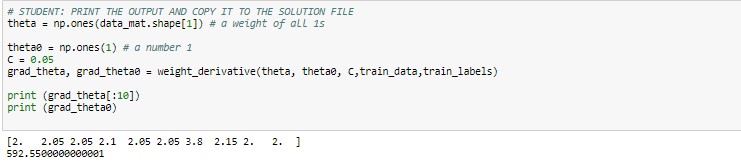
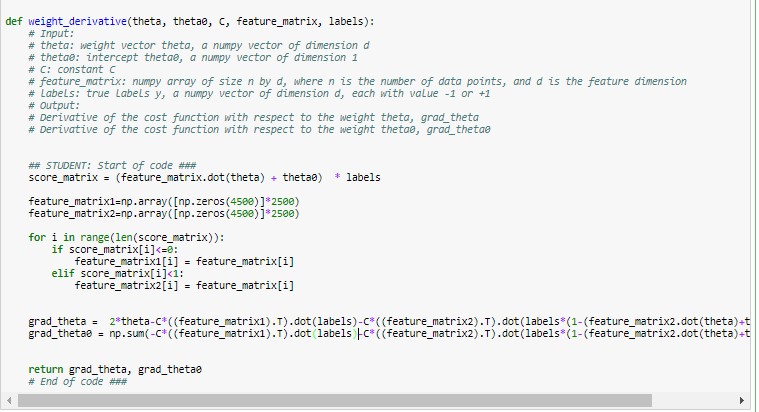
grad\_theta = 2\*theta-C\*((feature\_matrix1).T).dot(labels)-

C\*((feature\_matrix2).T).dot(labels\*(1-(feature\_matrix2.dot(theta)+theta0)\*labels)) grad\_theta0 = np.sum(-C\*((feature\_matrix1).T).dot(labels)-

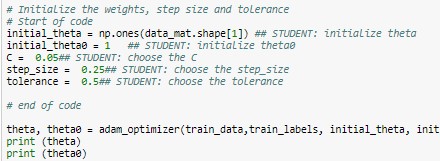
C\*((feature\_matrix2).T).dot(labels\*(1-(feature\_matrix2.dot(theta)+theta0)\*labels)))

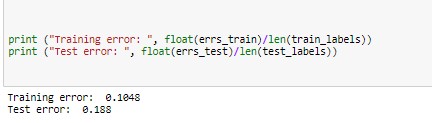
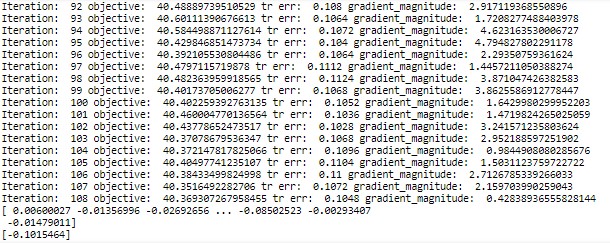
return grad\_theta, grad\_theta0

# End of code ###



6.





7.

