6.036: Final Exam, Fall 2021

Do not tear exam booklet apart!

- This is a closed book exam. Two pages (8 1/2 in. by 11 in.) of notes, front and back, are permitted. Computers, phones, and other electronics are not permitted.
- You have 3 hours.
- The problems are not necessarily in any order of difficulty.
- Write all your answers in the places provided. If you run out of room for an answer, indicate that you are continuing your answer, use the provided blank page at the end, and mark clearly what question is being continued.
- If a question seems vague or under-specified to you, make an assumption, write it down, and solve the problem given your assumption.

Name: _____

Kerberos (MIT username):_____

Question:	1	2	3	4	5	6	7	8	Total
Points:	14	12	12	12	14	12	12	12	100
Score:									

Name: _____

App Store

1. (14 points) Mac O'Larnin is considering selling an app on Frugal Play. You have a friend with inside info at Frugal, and they're able to share data on how previous apps have performed on the store.

   Mac decides that he will learn a neural network with no hidden layer (i.e., consisting only of the output layer). He needs help in figuring out the precise formulation for machine learning.

   (a) For each of the following app characteristics, suggest the best way of encoding it as feature(s) to be input to the neural network. Choose from among the following: multiple unary features (one-hot encoding), multiple binary features (thermometer encoding), an integer or real-valued feature. Also **give the exact function that maps each input to its corresponding feature(s)**.

   Genre (Game, Productivity, Education, Information, Social):

   Suitable for people ages (2–4, 5–10, 11–15, 16 and over):

   Was it banned in any previous quarter (True, False):

   Price of the app (positive number):

   Does it have in-game advertising (True, False):

(b) Mac wants to predict the sales volume (how many times someone will purchase the app each month) for his new app. The sales volume can be negative if many people returned the app for a refund in a given month. What should Mac choose for the number of units in the output layer, the activation function(s) in the output layer (linear, ReLU, sigmoid, softmax), and the loss function (negative log likelihood, quadratic)?

Number of units:

Activation function(s):

Loss function:

(c) Mac also has data on several other properties of interest, including

- whether an app was featured on the front page
- whether it got a favorable review on the Coolest Apps Evar web site
- whether Orange Computer offered to pay to port the app to their site

He would like to train a new neural network to predict these three properties.

For this new prediction task, what should Mac choose for the number of units in the output layer, the activation function(s) in the output layer (linear, ReLU, sigmoid, softmax), and the loss function (negative log likelihood, quadratic)?

Number of units:

Activation function(s):

Loss function:

(d) Mac's first attempt at machine learning to predict the sales volume (setup of (b)) uses all customer data from 2020. He randomly partitions the data into train (80%) and validation (20%), and uses the same number of units, activation function(s), and loss function as in (b). To prevent overfitting, he uses ridge regularization of the weights W, minimizing the optimization objective

$$J(W;\lambda) = \sum_{i=1}^{n} \mathcal{L}(h(x^{(i)};W), y^{(i)}) + \lambda \|W\|^2,$$

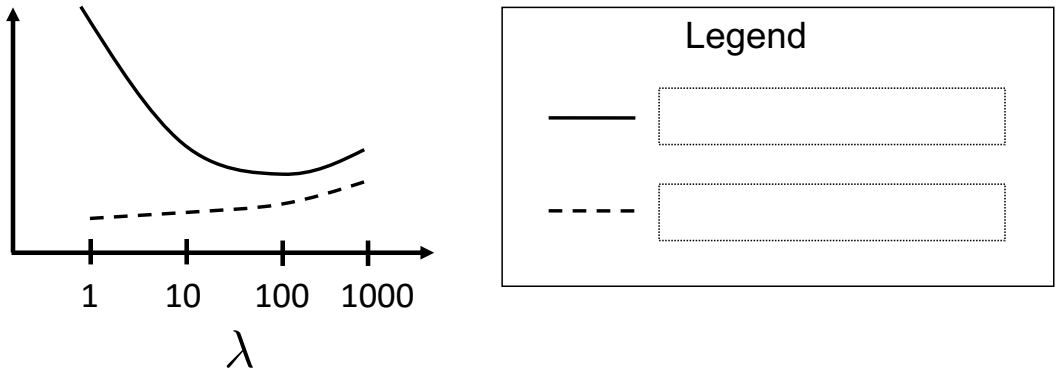
where $\|W\|^2$ is the sum over the square of all output units' weights.

Mac discovers that it's possible to find a value of W such that $J(W;\lambda) = 0$ even when λ is very large, nearing ∞. Mac suspects that he might have an error in the code that he wrote to derive the labels (i.e., the monthly sales volumes). Let's see why. First, what can Mac conclude about W from this finding? Second, what does this imply about the labels?

Conclusion about W:

Conclusion about the labels:

(e) Mac found and fixed the error. Now, to choose the regularization constant λ, Mac tried values of 1, 10, 100, and 1000, creating the below plot. Unfortunately, he forgot to label the legend! Help Mach by filling in the legend using two of the following: 'Training error', 'Validation error', 'Training time'.



(f) Continuing the scenario of (e), which value of λ (out of 1, 10, 100, and 1000) should Mac choose to obtain the neural network that he will deploy on the app store, and why?

$\lambda =$

Explanation:

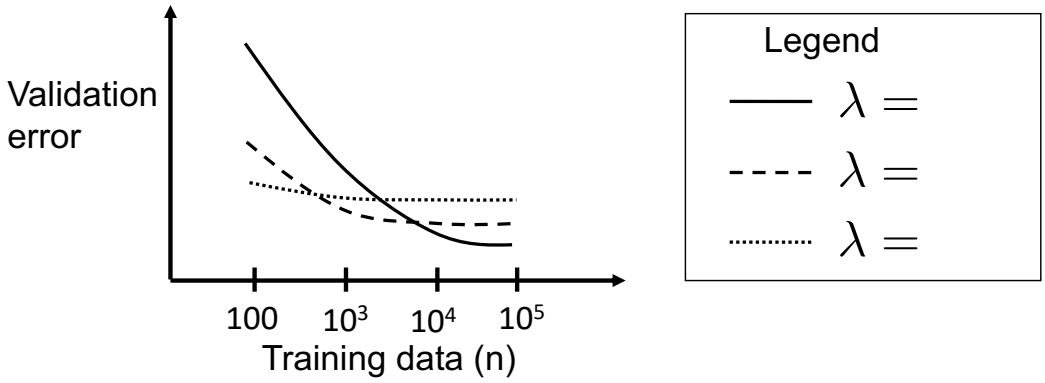
(g) When Mac wakes up the next day, he decides to re-run learning for the λ selected in (f), now with a different partition of the data into train and validation sets (since he had previously forgotten to set the random seed). He finds that he gets a very different validation error! To obtain a more stable estimate, Mac decides to split the data into 5 disjoint chunks of 20% of the data. For each chunk, he evaluates on it after training on the union of the other 4 chunks. He gets the following results for the average error within each chunk: 0.15, 0.3, 0.1, 0.2, 0.25. What can Mac conclude is an estimate of the test error of the neural network, and why?

Test error =

Explanation:

(h) The initial results look promising. Mac now wants to add in data from additional, earlier, years. (He is confident his customers have been behaving similarly over many years, so the earlier data is relevant.)

Before curating the older data, Mac decides to use the training data that he has to get a sense of whether more data would help. He creates a learning curve where on the horizontal axis he varies the amount of training data used and on the vertical axis he shows the validation error, using a fixed validation set across all settings considered. He experiments with $\lambda = 1, 10, 100$, but again forgot to include a legend. Fill in the below legend by labeling the curves with the value of λ that each corresponds to:



(i) Based on these plots does it seem likely that even more data will improve validation error (possibly for a different value of λ)? Explain why or why not.

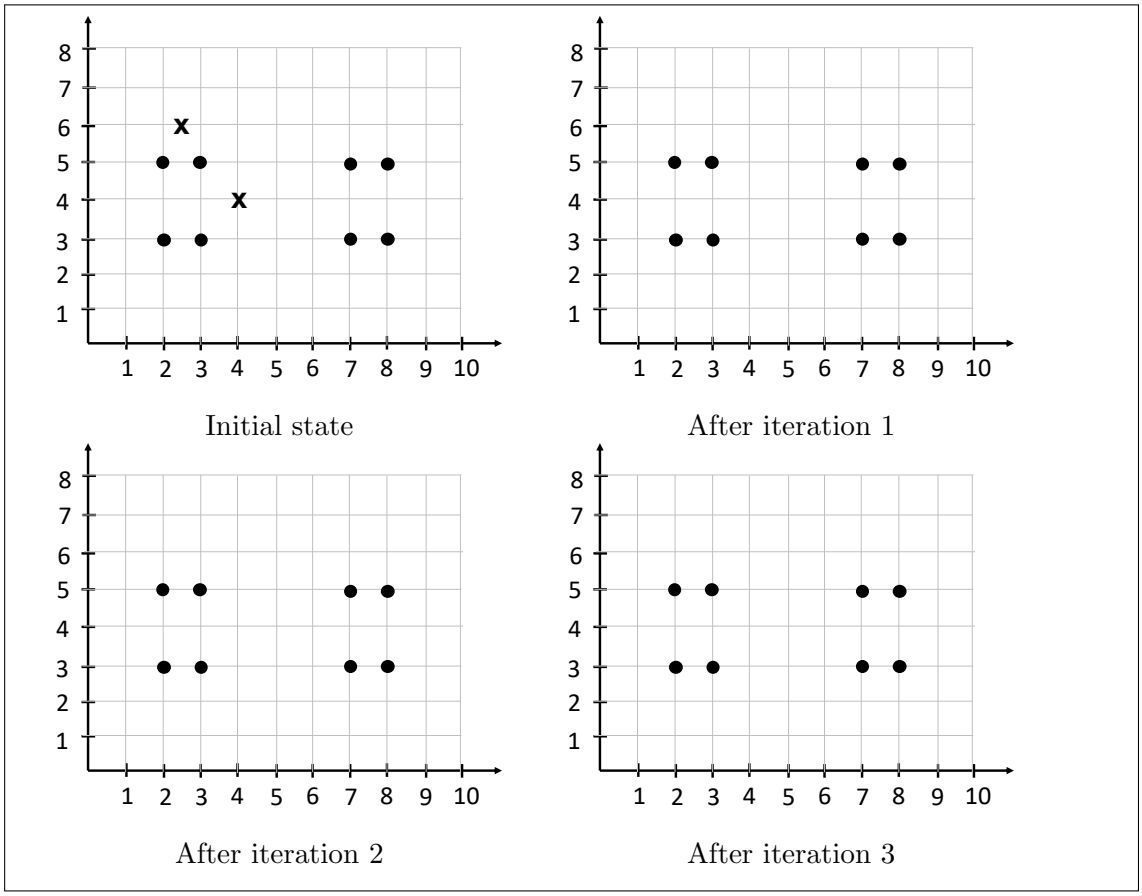
(j) Mac experiments with even more training data and additional values of λ, but finds that he cannot decrease the validation error further. Are there changes to the neural network architecture that Mac could make to try to improve prediction performance? Explain.

Name: _____

Clustering

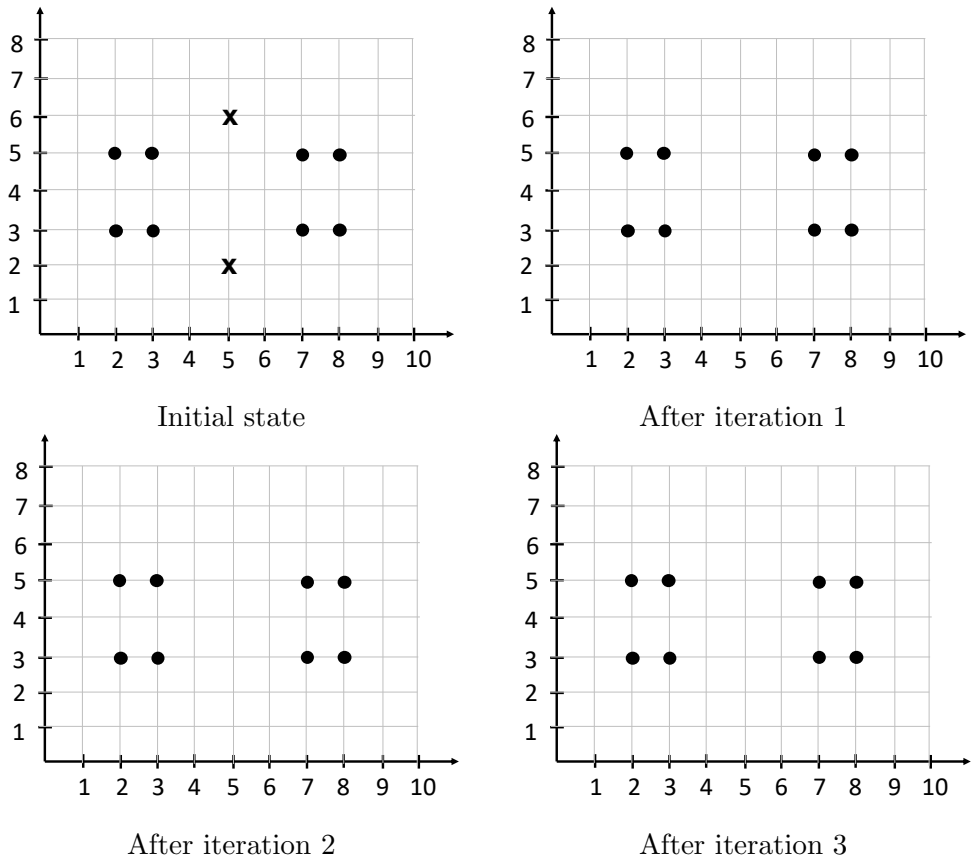
2. (12 points) Assume that the number of clusters $k = 2$ for all of the following questions.

   (a) Walk through each step of the k-means algorithm, beginning with the initialization shown in the plot in the top left of the box below. Dots show the observed data. In each plot (go left to right, top to down), mark with two '**x**' symbols where the cluster centers are in that iteration of k-means. These are already shown in the initial state. Once the k-means algorithm has converged, you can leave all subsequent plots unmarked.



   (b) What is the numerical value of the k-means objective for the clustering found in (a), after the algorithm has finished running?

(c) Just as in (a), walk through each step of the k-means algorithm, beginning with the initialization shown in the plot in the top left. In each plot (go left to right, top to down), mark with two '**x**' symbols where the cluster centers are in that iteration of k-means. Once the k-means algorithm has converged, you can leave all subsequent figures unmarked.

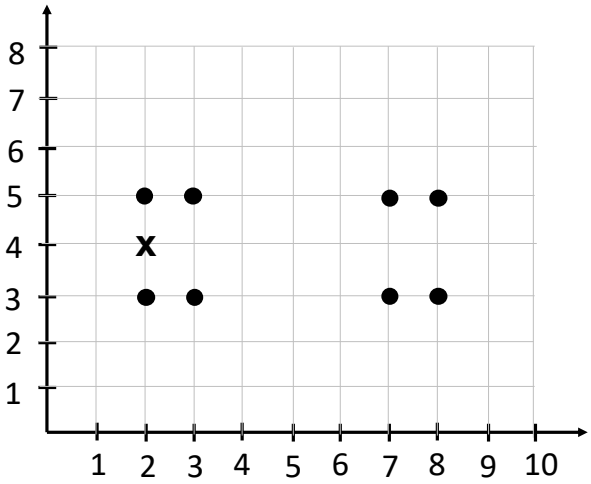


(d) What is the numerical value of the k-means objective for the clustering found in (c), after the algorithm has finished running?

(e) According to the k-means objective of the learned clusters, which initialization was better?

◯ Initialization (a) ◯ Initialization (c)

(f) Consider the data in black dots shown in the plot below. We drew one cluster center with an **x** symbol at (2,4). Draw the second cluster center to satisfy the following property. When we initialize the clusters centers at the two **x**'s and run the k-means algorithm to convergence, the final state will be such that one cluster will have all the data points assigned to it, and the other cluster will have no data points assigned to it.



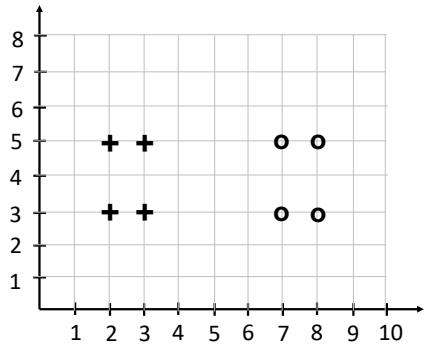
(g) Christy thinks she came up with a compelling new initialization method for the k-means algorithm. Looking at her code below, explain why it is unlikely to give good results.

```
def kmeans_init(X, n_clusters):
    centers = []
    for i in range(n_clusters):
        centers.append(X[:, X.shape[1]-1-i])
    return np.asarray(centers).T
```

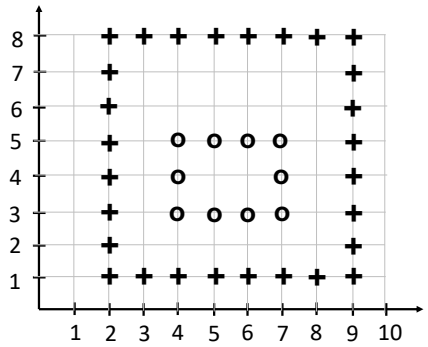
Name: _____

(h) Each of the following five data sets has two ground truth clusters, whose points are denoted as '+' and '**o**'. For which of these would the clustering with the smallest k-means objective value **not** recover the ground truth? Assume $k = 2$. (Select all that apply.)

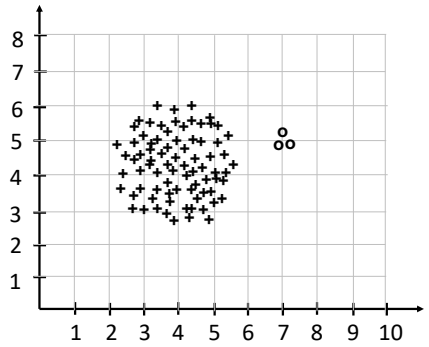
◯ (I)
◯ (II)
◯ (III)
◯ (IV)
◯ (V)



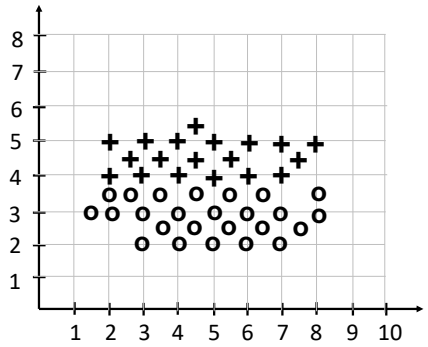
(I)



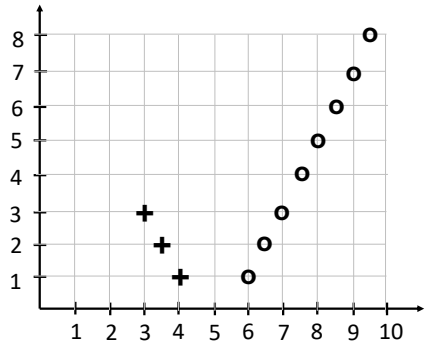
(II)



(III)



(IV)

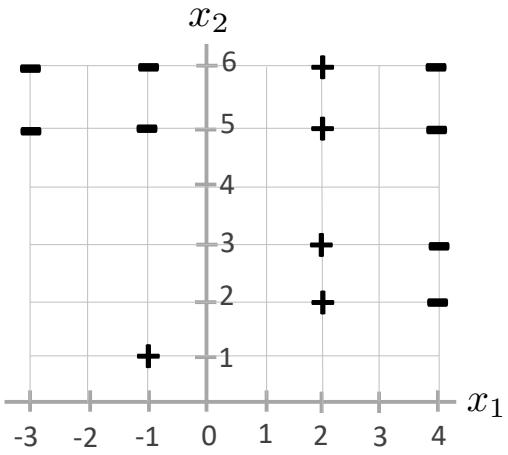


(V)

Name: _____

Decision trees

3. (12 points) We seek to learn a classifier on the data set shown on the left with 13 data points labeled $+1$ or -1. For your convenience, we include some helpful calculations in the table to the right.

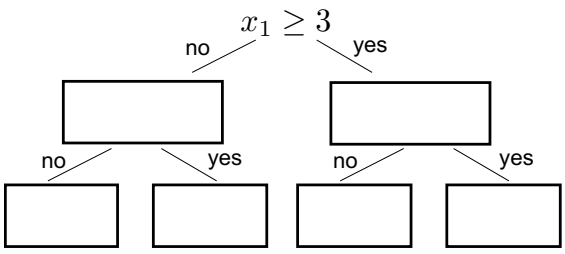


For reference:

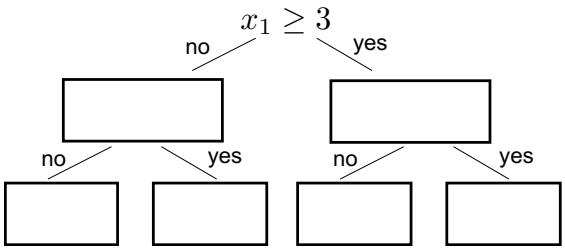
$-\frac{2}{7}\log_2\left(\frac{2}{7}\right)-\frac{5}{7}\log_2\left(\frac{5}{7}\right)\approx 0.86$
$\frac{7}{9}\times 0.86\approx 0.67$
$-\frac{4}{5}\log_2\left(\frac{4}{5}\right)-\frac{1}{5}\log_2\left(\frac{1}{5}\right)\approx 0.72$
$\frac{5}{9}\times 0.72\approx 0.40$
$-\frac{4}{6}\log_2\left(\frac{4}{6}\right)-\frac{2}{6}\log_2\left(\frac{2}{6}\right)\approx 0.92$
$\frac{6}{9}\times 0.92\approx 0.61$

(a) We first learn a linear logistic classifier with offset on this data set, with no regularization. Will it obtain zero training error? Write "yes" or "no" and explain your answer.

(b) We now learn a depth-2 decision tree, with min_samples_split=2. We give you a partially completed tree below, where the first split is $x_1 \geq 3$. Complete the rest of the tree by filling in the boxes with the splits on the second level and the classifications (either $+1$ or -1) at the leafs. Use the entropy criterion to choose the splits, and leave empty any boxes that are unused. As a reminder, min_samples_split is the minimum number of data points required to split an internal node.



(c) Now suppose that we set min_samples_split=10. Again, complete the rest of the tree by filling in the boxes. Leave empty any boxes that are unused.



(d) What is the purpose of increasing min_samples_split?

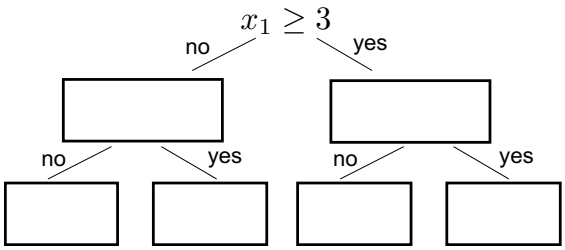
(e) What is the training error of the trees learned in parts (b) and (c)?

Tree (b):

Tree (c):

(f) With min_samples_split=2, if we were to continue building the tree without any restriction to its depth, what would be the training error of the resulting tree?

(g) Suppose we give as new features x_i^3, using these in addition to the original features x_i. Draw the new depth-2 tree that would be learned. Assume the features are organized x_1, x_2, x_1^3, x_2^3 and if two features are equally good for the split according to the entropy criterion, then we choose the first one in this order. As in part (b), assume min_samples_split=2.

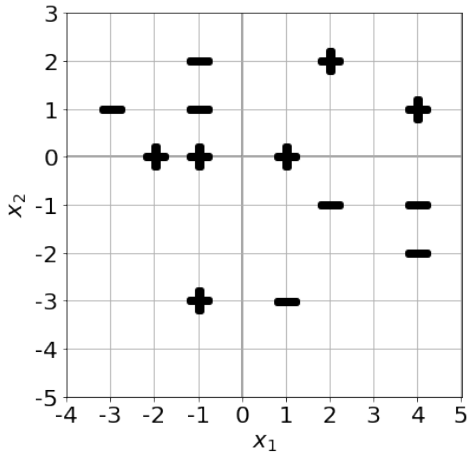


Name: _____

Nearest neighbor classifiers

4. (12 points) This question asks about learning nearest neighbor (NN) classifiers. Assume that we are using Euclidean distance squared as the distance metric, i.e. $d(x, x') = ||x - x'||^2$.

   (a) Draw on the below figure the decision boundary for a 1-NN classifier on this data set. In each region, denote whether the classification of any point (*any* point, not just the training data) in that region would be $+1$ or -1. (Note, all data points are assumed to be on integer coordinates.)



   (b) Which training data points, if any, could you remove and keep the decision boundary identical? Answer using their (x_1, x_2) coordinates.

   (c) You perform leave-one-out cross-validation of the 1-NN and 3-NN classifiers on this data set, i.e. you use use cross-validation with a chunk size of 1 data point. Assume ties go to the $+1$ region. What cross-validation errors do you obtain?

   1-NN:

   3-NN:

Name: _____

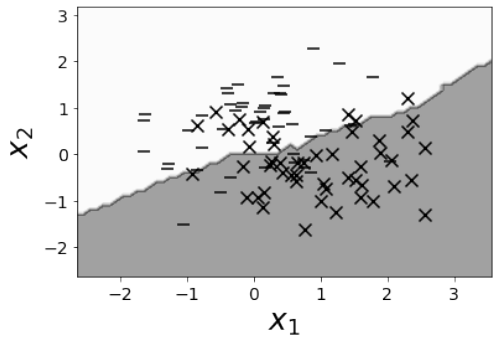
(d) Suppose we now use the following feature transformation, $\phi(x_1, x_2) = x_1 x_2$, and seek to learn a nearest neighbor classifier in the transformed space. This is equivalent to using a different distance metric, $d(x, x') = \|\phi(x) - \phi(x')\|^2$. What is the average leave-one-out cross-validation error of a 3-NN classifier using this new distance metric? Which points would be misclassified (specified using their (x_1, x_2) coordinates)?

3-NN:

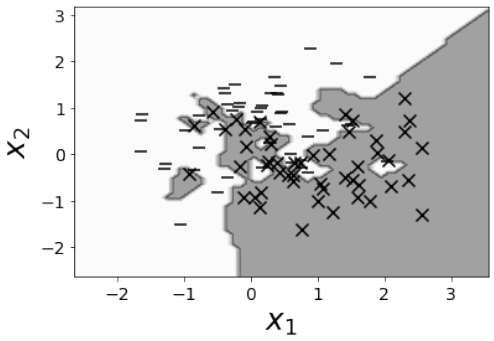
Misclassified points:

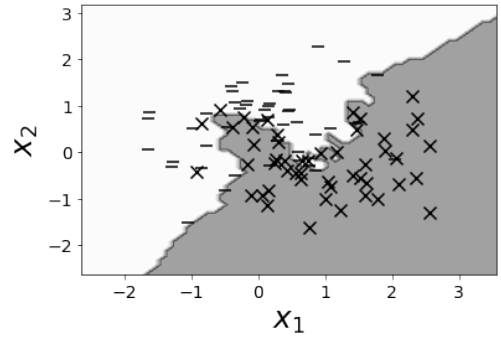
(e) The plots below show the decision boundaries as predicted by a k-NN classifier for four different values of k: 1, 5, 20, 40. Map each plot to the corresponding value of k.

(I)

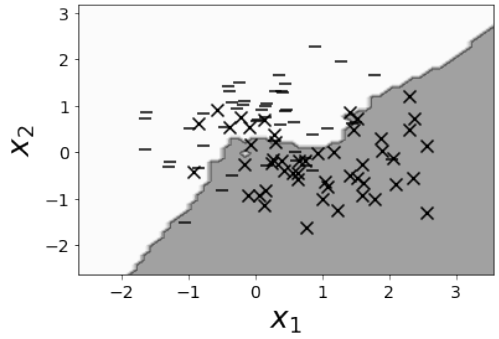


(II)





(III)



(IV)

$k = 1$:	◯ (I)	◯ (II)	◯ (III)	◯ (IV)
$k = 5$:	◯ (I)	◯ (II)	◯ (III)	◯ (IV)
$k = 20$:	◯ (I)	◯ (II)	◯ (III)	◯ (IV)
$k = 40$:	◯ (I)	◯ (II)	◯ (III)	◯ (IV)

Name: _____

Championship Material

5. (14 points) Ser Ena is a professional athlete who plays an individual sport. Ser is either **fully fit**, **partially fit** or **injured**. Regardless of her state, Ser can choose to **play** a tournament, take time to **train** or decide to take a complete **break** to rest. Ser's coaching team formulates an MDP to keep track of the states, actions, rewards and transitions. It is assumed that the discount factor is 1 (unless stated otherwise) and Ser is **fully fit** right before the next big tournament.The team comes up with the following MDP for Ser.

   (a) First the **Rewards** for the state, action pairs:

   When **fully fit**:

   - if Ser decides to **play**, there is a reward of +**100**;
   - if Ser decides to **train**, there is a reward of -**10**;
   - if Ser decides to take a **break**, there is **no reward**.

   When **partially fit**:

   - if Ser decides to **play**, there is a reward of +**20**;
   - if Ser decides to **train**, there is a reward of -**10**;
   - if Ser decides to take a **break**, there is a reward of -**20**.

   When **injured**:

   - if Ser decides to **play**, there is a reward of -**60**;
   - if Ser decides to **train**, there is a reward of -**30**;
   - if Ser decides to take a **break**, there is a reward of **0**.

   (b) Next, the **transition probabilities**:

   When **fully fit**:

   - if Ser decides to **play**, there is a **80%** chance of remaining fully fit, **20%** chance of getting injured.
   - if Ser decides to **train**, there is a **90%** chance of remaining fully fit, **10%** chance of getting injured.
   - if Ser decides to take a **break**, there is a **50%** chance of remaining full fit, **50%** chance of being partially fit.

   When **partially fit**:

   - if Ser decides to **play**, there is a **50%** chance of remaining partially fit; and a **50%** chance of getting injured.
   - if Ser decides to **train**, there is a **40%** chance of remaining partially fit, **60%** chance of getting fully fit;.
   - if Ser decides to take a **break**, Ser will remain partially fit.

   When **injured**:

   - if Ser decides to **play**, Ser will remain injured.
   - if Ser decides to **train**, Ser will remain injured.
   - if Ser decides to take a **break**, there is a **50%** chance Ser will remain injured and a **50%** chance of being partially fit.

Name: _____

(a) What is the horizon 1 optimal policy for Ser? Assume discount is 1.

(b) For horizon 2, what is the best action to take when Ser is **partially fit** and what is the horizon 2 expected reward for taking that best action when **partially fit** ? Assume discount of 1. Please show how you arrived at the answer.

$\pi_2^*(\text{partially fit}) =$

$Q_2(\text{partially fit}, \text{a}^*) =$

(c) Does the answer to (b) change if the discount factor was 0.5? Explain why or why not.

(d) What is the infinite horizon optimal policy for Ser? Assume discount is 1.

(e) Is there any policy which maximizes the expected reward in the infinite horizon under which Ser should play if injured? Explain.

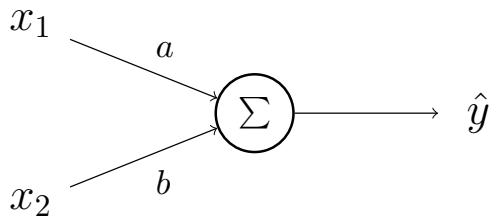
(f) Djo Ko is another athlete who plays the same sport. Djo Ko has the exact same MDP as Ser Ena's, except Djo's team has **forgotten the reward for *playing* when in the *fully fit*** state. Djo's team also remember that the horizon 2 best action to take in the **partially fit** state is exactly the same as that for Ser Ena (determined in part b). Given this information, what are the range of possible values for R(fully fit, play) for Djo Ko? Assume discount of 1.

Name: _____

Harmony in Descent

6. (12 points) Years ago, MIT student Itu Nes learned about neural networks and how to train them, from taking 6.036. Now Itu is an engineer at Orange Computer, a hot tech company employing machine learning to revolutionize music. Looking back at her notes, Itu realizes that she once wrote down exactly what she now needs to do in her job, but unfortunately some key details are lost. Can you help her figure things out?

   Specifically, Itu wants to train this simple single-node neural network:



   The network accepts two inputs x_1 and x_2, and outputs a prediction $\hat{y}$ based on weights a and b. Itu's dataset has points (x, y) where $x = (x_1, x_2)$, and y are the true labels. Itu employs the squared error loss function

$$L(\hat{y}, y) = (y - \hat{y})^2 \ .$$

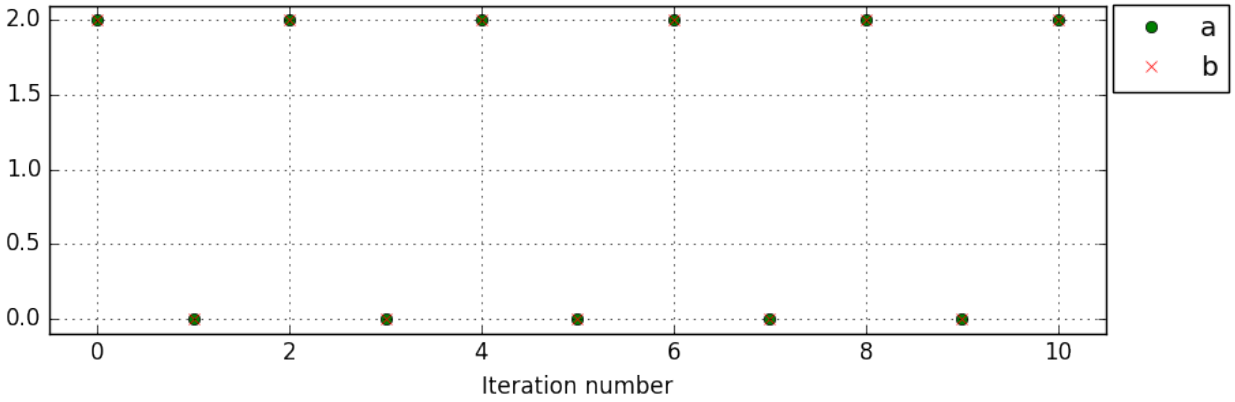
   In her notes, Itu wrote about using gradient descent to obtain the optimal weights for the network, by minimizing this loss. Moreover, for each run of the gradient descent, she used a single data point to train the weights. Afterwards, Itu learns that the true labels are $y = x_1 + x_2$.

   (a) Suppose a_0 and b_0 are the initial values of the weights, and a_k and b_k are the weights at iteration k. Give equations for the updated weights a_{k+1}, b_{k+1} in terms of current iteration's weights a_k, b_k, the step size parameter η, and the inputs x_1, x_2.

$a_{k+1} =$

$b_{k+1} =$

(b) Itu sees that when she fixed $x_1 = 1$, $x_2 = 1$ and ran 10 iterations of gradient descent starting with $a_0 = 2$, $b_0 = 2$, she recorded that the two weights oscillated back and forth, as captured in this plot pasted into her notebook:



Note that in this plot, the a and b points lay on top of each other. Unfortunately, Itu forgot to write down her code, nor did she write down what value of η may have been used to generate this plot. Help her figure out: was this plot a mistake (and explain why), or if not, what value of η could have generated it?

This plot:

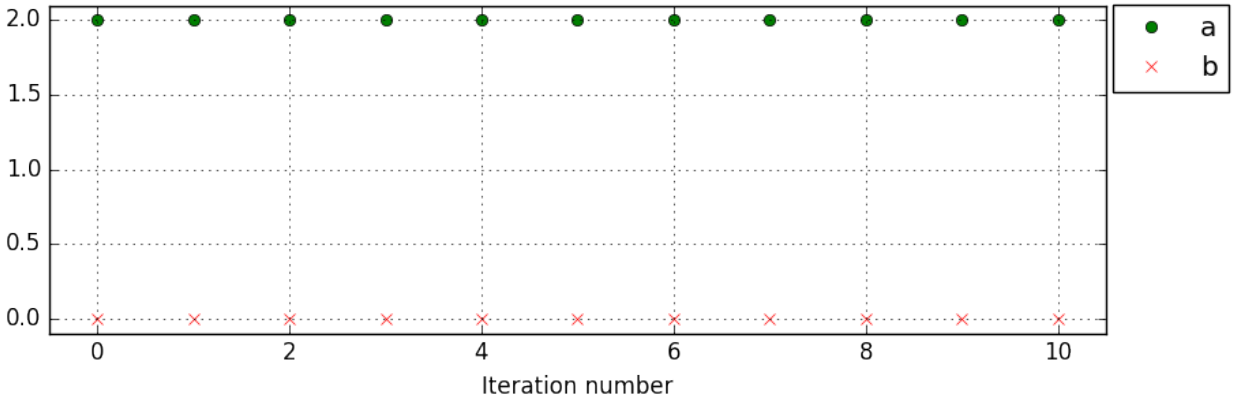
◯ Cannot have been produced by any η

because :

◯ Resulted from choosing a specific η, in particular

$\eta =$

(c) Itu sees that when she fixed $x_1 = 1$, $x_2 = 1$ and ran 10 iterations of gradient descent starting with $a_0 = 2$, $b_0 = 0$, she recorded that the two weights remained unchanged, as captured in this plot pasted into her notebook:



Again: was this plot a mistake (and explain why), or if not, what value of η could have generated it?

This plot:

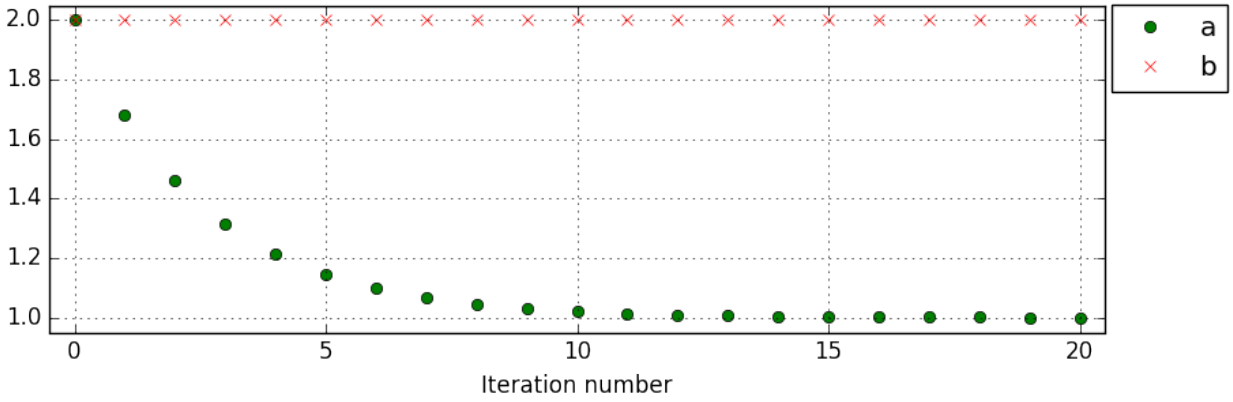
◯ Cannot have been produced by any η

because :

◯ Resulted from choosing a specific η, in particular

$\eta =$

(d) Itu sees that when she fixed x_1 and x_2 and ran 10 iterations of gradient descent with $\eta = 0.01$ starting with $a_0 = b_0 = 2$, she recorded that b stayed unchanged, but a decayed to 1, as captured in this plot pasted into her notebook:



Again: was this plot a mistake (and explain why), or if not, what values of x_1, x_2 could have generated it?

This plot:

◯ Cannot have been produced by any x_1, x_2

because :

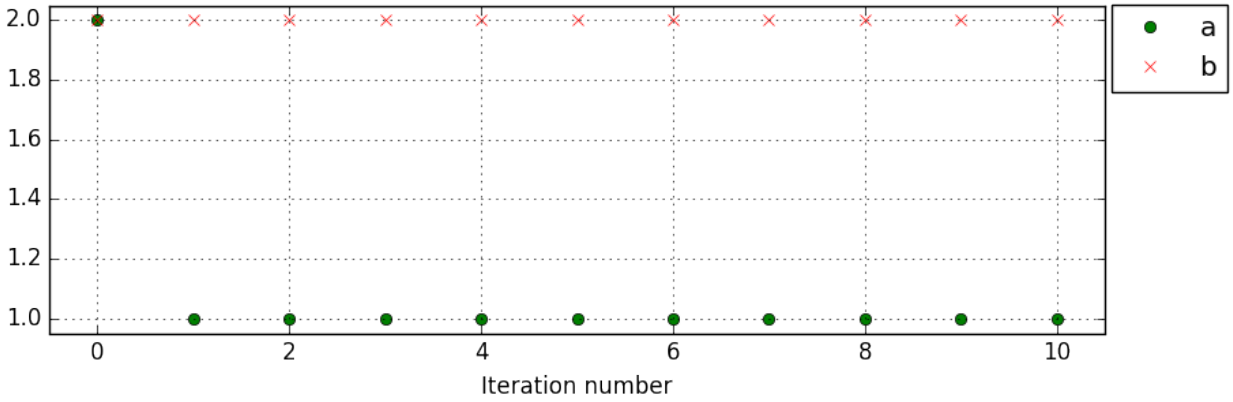
◯ Resulted from choosing specific inputs, in particular

$x_1 =$

$x_2 =$

Name: _____

(e) Itu sees that when she fixed $x_1 = 4$ and $x_2 = 0$ and ran 10 iterations of gradient descent starting with $a_0 = b_0 = 2$, she recorded that b stayed unchanged, but a jumped immediately on the first iteration to 1, as captured in this plot pasted into her notebook:



Again: was this plot a mistake (and explain why), or if not, what value of η could have generated it?

This plot:

◯ Cannot have been produced by any η

because :

◯ Resulted from choosing specific η, specifically

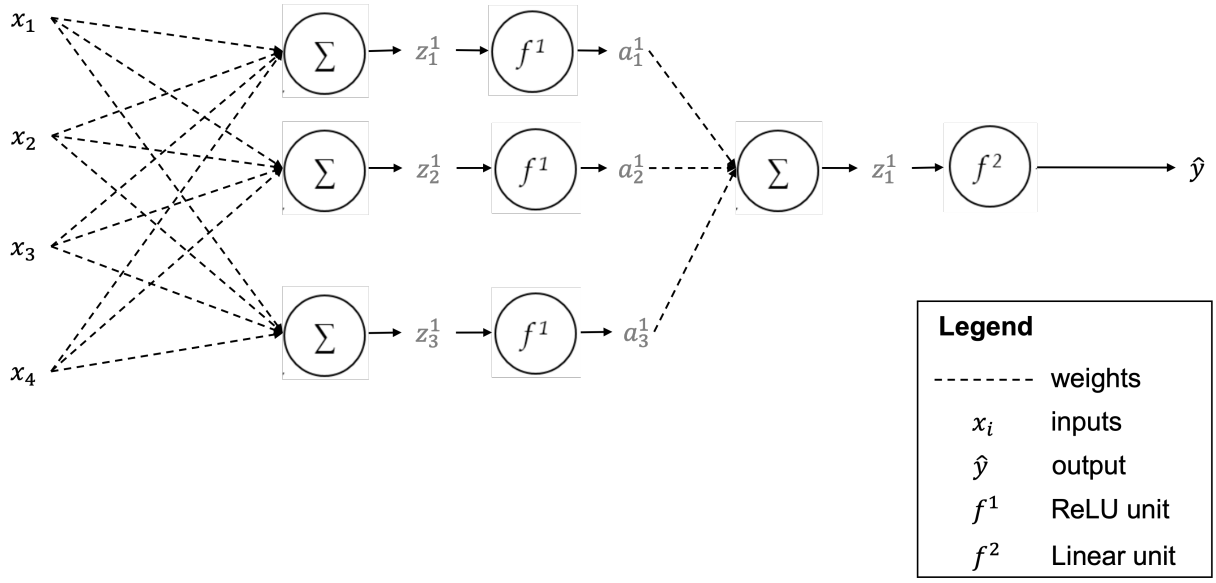
$\eta =$

Name: _____

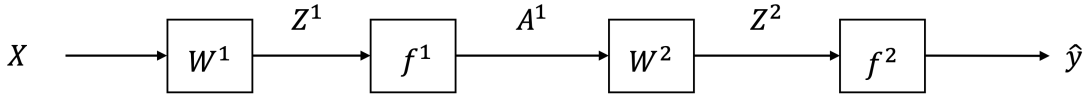
Forward and Backwards

7. (12 points) A feed-forward neural network is shown below. As noted in the Legend, and following 6.036 conventions,

   - The dashed lines represent the weights that the neural network learns
   - There are no bias/constant terms among the weights
   - We are using squared-loss, i.e. $L(\hat{y}, y) = (y - \hat{y})^2$
   - Inputs are represented by $X = [x_1, x_2, x_3, x_4]^T$
   - The *true* output value is denoted by y, and the estimation output by the network is $\hat{y}$
   - f^1 are ReLU units
   - f^2 is a linear unit



   (a) The neural network shown above can be also be represented by the network shown below, which uses matrix notation. Specifically, the input X is a 4×1 column vector, $\hat{y}$ is a 1×1 scalar. W^2 is a 3×1 vector. We also know that, $Z^1 = (W^1)^T X$ and $Z^2 = (W^2)^T A^1$.



   i. What are the dimensions of the matrix W^1?

ii. What are the dimensions of Z^2?

(b) For both parts below, you are told that there is only **one data point** which is: $X = [1, 1, 1, 1]^T$ and $y = [1]$.

i. If W^1 and W^2 are both matrices/vectors of all ones, what is the resulting Loss?

ii. If W^1 is a matrix of all -1's (all negative ones) and W^2 is a vector of all 1's (positive ones), what is the resulting Loss?

(c) i. Determine the expression for $\frac{\partial L}{\partial W^1}$. You may leave your expression in terms of $X, y, \hat{y}, W^2$ and $\frac{\partial A^1}{\partial Z^1}$.

ii. What are the dimensions of $\frac{\partial L}{\partial W^1}$?

(d) We now use back-propagation to update the weights during each iteration. For all questions below, assume that we only have one data point (X, y) available to use, and the stepsize parameter is 0.01. You are asked to determine how many components of W^1 will get updated (i.e. have their value changed) in each scenario below.

i. Assume $X = [1, 1, 1, 1]^T, y = [1]$. Further assume that we start with W^1 as a matrix of -1's (negative ones) while W^2 is a vector of 1's (positive ones). How many components of W^1 will get updated (i.e. have their value changed) after one iteration of backprop? Explain your answer.

ii. Assume $X = [0, 0, 0, 0]^T, y = [0]$. Further assume that we start off with W^1 and W^2 as matrices/vectors of all ones. How many components of W^1 will get updated (i.e. have their value changed) after one iteration of back-propagation? Explain your answer.

iii. Assume $X = [1, 1, 1, 1]^T, y = [1]$. Further assume that we start off with W^1 and W^2 as matrices/vectors of all ones. How many components of W^1 will get updated (i.e. have their value changed) after one iteration of backprop? Explain your answer.

iv. Assume $X = [1, 1, 1, 1]^T, y = [1]$. Further assume that we start off with W^1 as a matrix of all ones. $W^2 = [0, 1, 0]^T$. How many components of W^1 will get updated (i.e. have their value changed) after one iteration of backprop? Explain your answer.

Name: _____

A Tiny CNN for Tetris

8. (12 points) MIT grad student Rec Urrent would like to submit an entry to win this year's Grand ML Tetris Competition, which gives awards to the smallest neural networks which can identify tetris pieces with the highest accuracy. Rec seeks to make a convolutional neural network that can accurately classify single-channel 3×3 images of 2D tetris pieces as being either a line-shaped piece, or a corner-shaped piece, using just one 2×2 filter. Let's help Rec win this competition.

   (a) What are the spatial dimensions of the output image if a 2×2 filter is convolved with a 3×3 image for paddings of 0, 1, and 2, and strides of 1 and 2? Fill in the dimensions below:

Padding	0	1	2
Stride 1	_____	_____	_____
Stride 2	_____	_____	_____

   (b) Rec writes a bit of python code to implement their tiny CNN classifier for images of 2D tetris pieces, following examples they have seen in 6.036. They include in the comments the dimensions of the numpy arrays, where known.

```
def tinycnn(x, fcoef, w, b, final_act):
    """
    x: (numpy array, dimensions [1,3,3]) input image
    fcoef: (numpy array, dimensions [1,1,2,2]) conv filter coefficients
    w: (numpy array, dimensions [?, ?]) weights for classifier network
    b: (numpy array, dimensions [?]) bias for classifier network
    final_act: (function) final output activation
    """
    z = conv2d(x, fcoef, padding=0, stride=1)  # [1, ?, ?]
    a = ReLU(z)
    a_sum = z1.sum(dim=-1).sum(dim=-1)         # [1] sum spatial dim
    z2 = w.T @ a_sum + b
    return final_act(z2)
```

   For performing binary classification, what activation function should Rec use for `final_act` and which loss function should Rec use?

   Activation function:

   Loss function:

Name: _____

(c) If Rec wants to allow for more than two classes, which activation function should they use for `final_act` and which loss function?

Activation function:

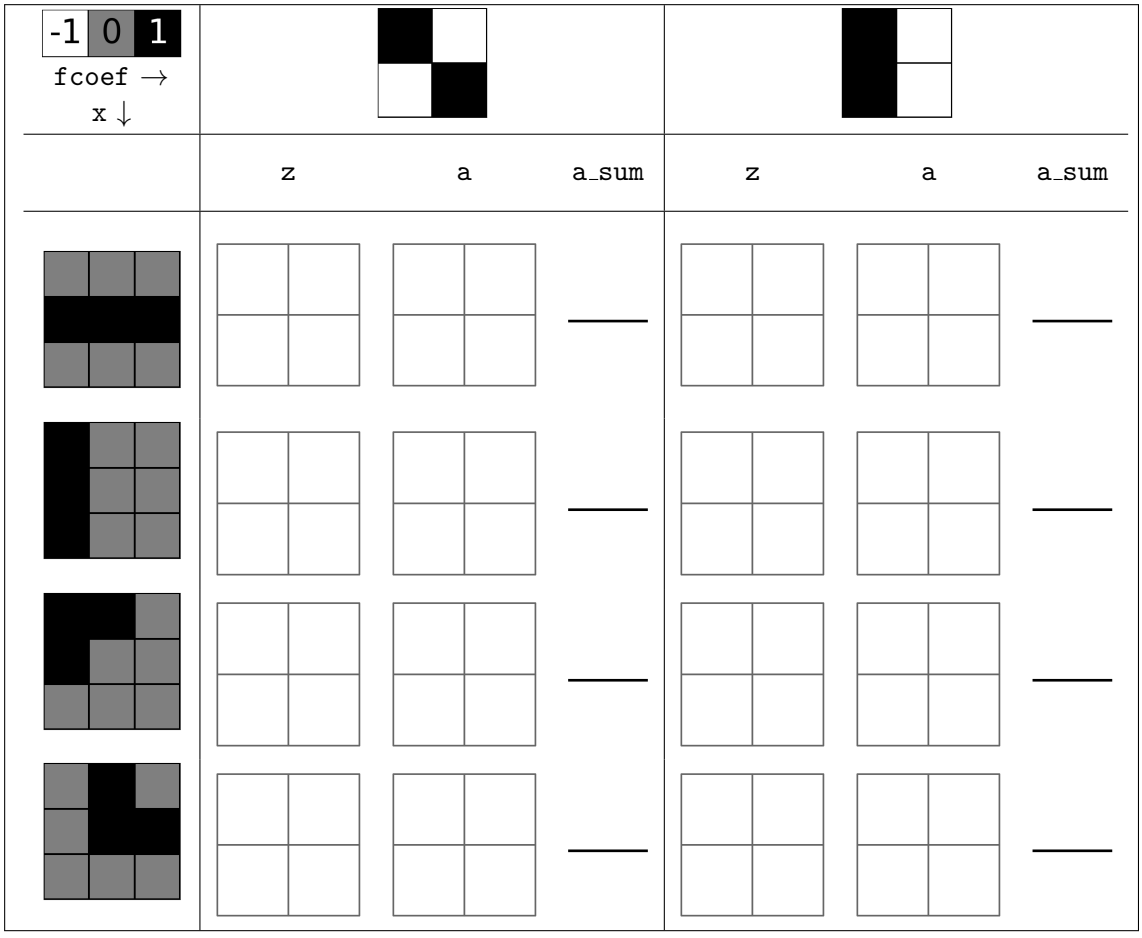
Loss function:

(d) What are dimensions of `w` and `b` for i) binary classification vs. ii) k-class classification?

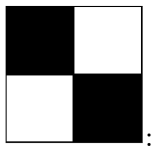
For binary classification `w` is: and `b` is:

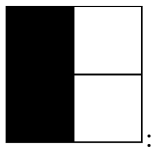
For k-class classification `w` is: and `b` is:

(e) To debug their code, Rec runs `tinycnn` on four different 3×3 input images `x`, and two different 2×2 filters `fcoef`. They add print statements to see `z`, `a`, and `a_sum` for the 8 cases. Fill in the table below with numbers giving what Rec obtains. `fcoef` and `x` are depicted with 1 = black, 0 = grey, and -1 = white.



(f) What does each filter do? Which filter is best for distinguishing line-shaped tetris pieces vs. corner-shaped pieces? Why?

Filter  :

Filter  :

(g) Rec labels corner-shaped tetris pieces as "1" and line-shaped tetris pieces as "0". Using this labeling, what values of `w` and `b` of the output layer give perfect classification and outputs that are close to 1 for corners and close to 0 for lines? (Assume the examples in (e) are representative of the entire dataset.) You may find the plots provided on the last page of this exam helpful.

(h) If Rec instead labeled line-shaped pieces as "1" and corner-shaped pieces as "0" then what values of `w` and `b` of the output layer give perfect classification and outputs that are close to 0 for corners and close to 1 for lines?

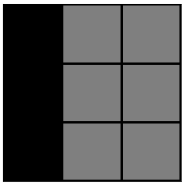
(i) Write an expression for the derivative of the binary classification loss with respect to `z2`, the input of `final_act`. You may express your answer using g for the output of `final_act` and y for the example label.

(j) Using your answers from above, write an expression for gradient of the loss with respect to `w` and `b` of the output layer. You may express your answers in terms of `a_sum`.

$\frac{\partial \mathcal{L}}{\partial w} =$

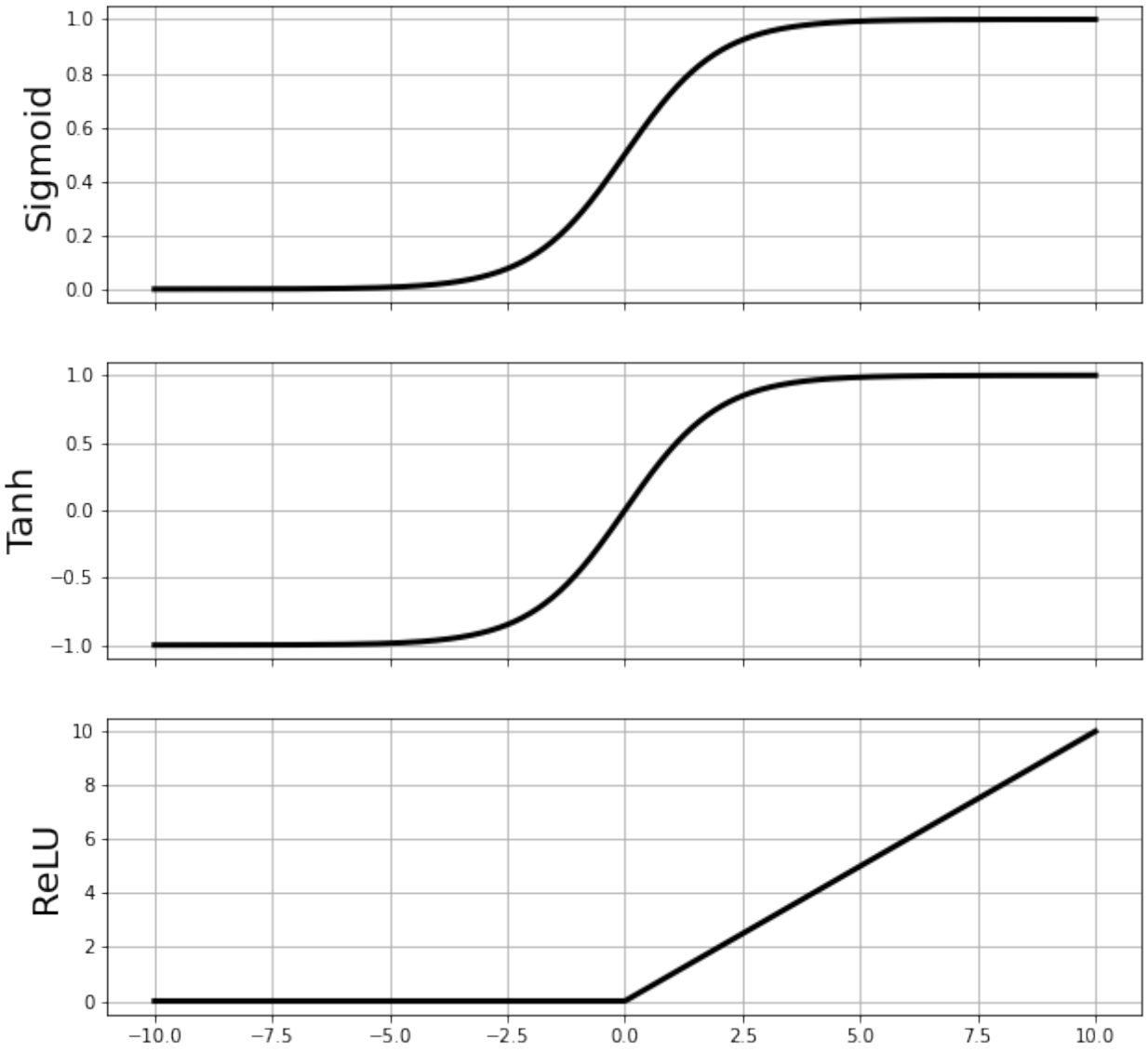
$\frac{\partial \mathcal{L}}{\partial b} =$

(k) Assume we apply a filter with weights `[[f1, f2], [f3, f4]]` to this 3×3 image:



with stride 1 and padding 0 and perform back propagation. Which filter weights may have non-zero gradients? Why? Under what conditions will those gradients be non-zero?

Name: _____


Sigmoid
1.0
0.8
0.6
0.4
0.2
0.0
Tanh
1.0
0.5
0.0
−0.5
−1.0
ReLU
10
8
6
4
2
0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

Name: _____

Work space

Name: _____

Work space