edX Course: 6.86x

Contents

Midterm Exam (1 week)

Midterm Exam 1

edXvertical: Exam Rules
edXtext: Exam Rules

1. You have opened a timed exam with a **48 hours** time limit. Please use the timer to see the time remaining. If you had opened this exam too close to the exam **closing time, August 7 23:59UTC** , you will not have the full 48 hours, and the exam will close at the closing time.

2. This is an **open book exam** and you are allowed to refer back to all course material, use software at your disposal. However, you must abide by the honor code, and **must not ask for answers directly from any aide or any online resources** .

3. As part of the honor code, you **must not share the exam content** with anyone in any way, i.e. **no posting of exam content anywhere on the internet** . Violators will be removed from the course.

4. You will be given **no feedback** during the exam. This means that unlike in the problem sets, you will not be shown whether any of your answers are correct or not. This is to test your understanding, to prevent cheating, and to encourage you to try your very best before submitting. Solutions will be available after the exam closes.

5. You will be given **3 attempts** for each (multipart) problem. Since you will be given no feedback, the extra attempts will be useful only in case you hit the "submit" button in a haste and wish to reconsider. **With no exception, your last submission will be the one that counts** . **DO NOT FORGET TO SUBMIT your answers to each question** . The "end your exam" button will **not** submit answers for you.

6. The exam will only be **graded 2 days after the due date** , and the **Progress Page will show fake scores while the exam is open** .

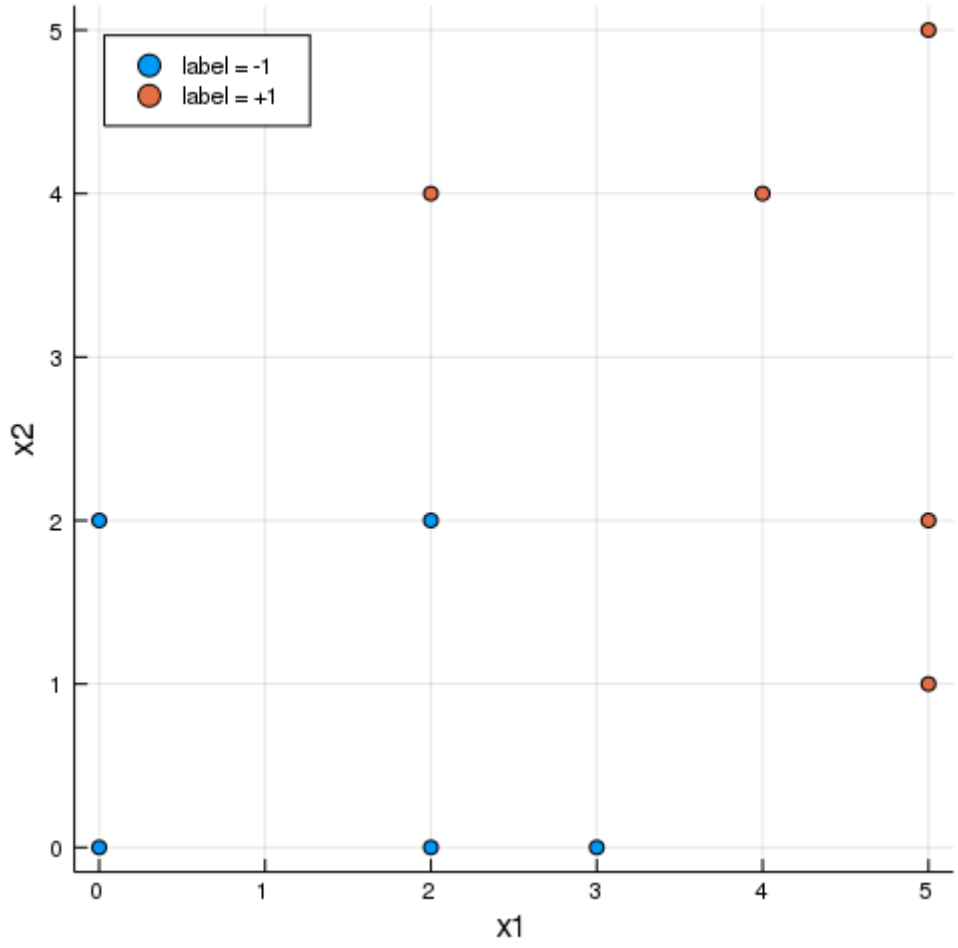
7. **Error and bug reports:** While the exam is open, you are **not allowed to post on the discussion forum on anything related to the exam, except to report bugs/platform difficulties** . If you think you have found a bug, please **state on the forum only what needs to be checked on the forum** . You can still post questions relating to course material, but **the post must not comment on the exam** , and in particular **must not shed any light on the contents or concepts in the exam** . **Violators will receive a failing grade or a grade reduction in this exam** .

8. **Clarification:** If you need clarification on a problem, please first **check the discussion forum** , where staff may have posted notes. After that, if you still need clarification that will **strictly not lead to hints of the solution** , you can email staff at 686exam@mit.edu. If we see that the issue is indeed not addressed already on the forum, we will respond within 28 hours and post a note on the forum; otherwise–if the issue has been addressed on the forum, we will **not** respond and assume your responsibility to check the forum for answers.

edXvertical: Problem 1
edXtext: Problem 1

Problem 1. Linear Classification

Consider a labeled training set shown in figure below:



edXproblem: 1. (1)

We initialize the parameters to all zero values and run the **linear perceptron algorithm** through these points in a particular order until convergence. The number of mistakes made on each point are shown in the table below. (These points correspond to the data point in the plot above)

Label	-1	-1	-1	-1	-1	+1	+1	+1	+1	+1
Coordinates	(0,0)	(2,0)	(3,0)	(0,2)	(2,2)	(5,1)	(5,2)	(2,4)	(4,4)	(5,5)
Perceptron mistakes	1	9	10	5	9	11	0	3	1	1

Note: You should be able to arrive at the answer without programming.

What is the resulting offset parameter θ_0?

Enter the numerical value for θ_0: $\theta_0 =$ **Assessment content removed**

What is the resulting parameter θ?

(Enter θ as a vector, e.g. type `[0,1]` if $\theta = \begin{bmatrix}0 & 1\end{bmatrix}^T$.)

from mitxgraders import * grader = MatrixGrader(answers='[4,4]', max$_a rray_a im = 1, tolerance =' 2)

$\theta =$ **Assessment content removed**

edXinclude: ../xml/standardnotation.xml

Correction note: July 30 17:00UTC In an earlier version, the note "You should be able to arrive at the answer without programming are not present."

edXproblem: 1. (2)

Setup as above: We initialize the parameters to all zero values and run the **linear perceptron algorithm** through these points in a particular order until convergence. The number of mistakes made on each point are shown in the table below. (These points correspond to the data point in the plot above.)

Label	-1	-1	-1	-1	-1	+1	+1	+1	+1	+1
Coordinates	(0,0)	(2,0)	(3,0)	(0,2)	(2,2)	(5,1)	(5,2)	(2,4)	(4,4)	(5,5)
Perceptron mistakes	1	9	10	5	9	11	0	3	1	1

The mistakes that the algorithm makes often depend on the order in which the points were considered. Could the point $(5, 2)$ labeled $+1$ have been the first one considered?

Assessment content removed

Correction Note July 29 15:00UTC: An earlier version of the exam does not include the clarification titled "Setup as above".

edXproblem: 1. (3)

Suppose that we now find the linear separator that **maximizes** the margin instead of running the perceptron algorithm.

What are the parameters θ_0 and θ corresponding to the **maximum margin separator**?

(Enter θ_0 accurate to at least 3 decimal places.)

$\theta_0 =$ **Assessment content removed**

(Enter θ as a vector, enclosed in square brackets, and components separated by commas, e.g. type `[0,1]` for $\begin{bmatrix}0 & 1\end{bmatrix}^T$.)

from mitxgraders import * grader = MatrixGrader(answers='[1,1]', max_array_dim = 1, tolerance = 0.01)

$\theta =$ **Assessment content removed**

edXinclude: ../xml/standardnotation.xml

Although it is not required by the problem, we feel it might be helpful to also discuss how to find "maximum margin separater" generally. In fact, this is known as the hard margin SVM, i.e., to find the linear hyperplane with the maximum margin, such that all points are classified correctly. In particular, if we omit the bias term θ_0, the goal is to maximize $1/\|\theta\|$ (or minimize θ^2, equivalently), given $y^{(i)}(\theta \cdot x^{(i)}) \geq 1$ for all i. This optmization problem can be then written as the following quadratic programming formulation:

$$\begin{aligned} \min_{\theta} \ & \theta^2 \\ \text{s.t. } & y^{(i)}(\theta \cdot x^{(i)}) \geq 1, \forall\ i \end{aligned}$$

Note this is different from the commonly seen soft margin SVM, where you can allow some classification errors (that's why it's called "soft" instead of "hard"). In comparison, the soft margin SVM corresponds to the following optmization problem:

$$\begin{aligned} \min_{\theta, \xi \geq 0} \ & \theta^2 + C\sum_i \xi_i \\ \text{s.t. } & y^{(i)}(\theta \cdot x^{(i)}) \geq 1 - \xi_i, \forall\ i \end{aligned}$$

If you use SVM program directly from machine learning toolkits such as scikit-learn, it is by default the soft-margin SVM (in scikit-learn the default C equals to 1); and some regularizer term is also likely to be there by default on the objective. If you want to numerically use the

soft SVM program to approximately produce a hard SVM result, you need to be very careful on the parameters ($C \leftarrow \infty$, $\lambda \leftarrow 0$). Then you would get something very close within the numerical tolerance.

edXproblem: 1. (4)

What is the value of the margin attained?

(Enter an exact answer or decimal accurate to at least 2 decimal places.)

from mitxgraders import *

grader = SingleListGrader(answers=(['1/sqrt(2)'], ['2/sqrt(2)'],), subgrader=NumericalGrader(tolerance=0.01),)

Assessment content removed

Grading note: Both reasonable answers, i.e. $1/\sqrt{2}$ and $2/\sqrt{2}$, are accepted.

edXproblem: 1. (5)

Using the parameters θ_0 and θ corresponding to the **maximum margin separator**, what is the sum of Hinge losses evaluated on each example?

Sum of hinge losses: **Assessment content removed**

Correction Note (July 31 15:00 UTC): An earlier version does not include "Using the parameters θ_0 and θ corresponding to the **maximum margin separator**" in the problem statement.

edXproblem: 1. (6)

Suppose we modify the maximum margin solution a bit and divide both θ and θ_0 by 2. What is the sum of Hinge losses evaluated on each example for this new separator?

Sum of hinge losses: **Assessment content removed**

edXvertical: Problem 2
edXtext: Problem 2

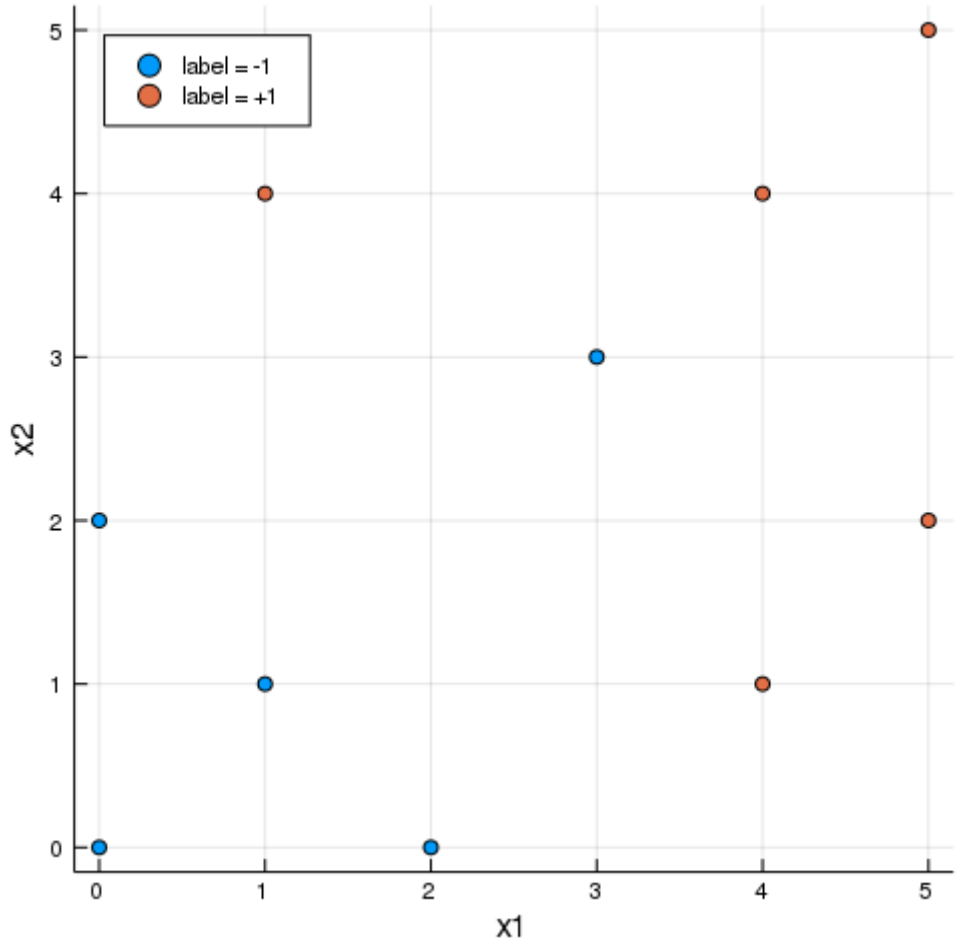
Problem 2. Kernel Methods

In this problem, we want to do classification over a different training dataset, as shown in plot below:

edXproblem: 2. (1)

If we again use the linear perceptron algorithm to train the classifier, what will happen?

Note: In the choices below ,"converge" means given a certain input, the algorithm will terminate with a fixed output within finite steps (assume T is very large: the output of the



algorithm will not change as we increase T). Otherwise we say the algorithm diverges (even for an extremely large T, the output of the algorithm will change as we increase T further).

Assessment content removed

edXproblem: 2. (2)

We decide to run the kernel perceptron algorithm over this dataset using the quadratic kernel. The number of mistakes made on each point is displayed in the table below. (These points correspond to those in the plot above.)

Label	-1	-1	-1	-1	-1	+1	+1	+1	+1	+1
Coordinates	(0,0)	(2,0)	(1,1)	(0,2)	(3,3)	(4,1)	(5,2)	(1,4)	(4,4)	(5,5)
Perceptron mistakes	1	65	11	31	72	30	0	21	4	15

Define the feature map of our quadratic kernel to be:

$$\phi(x) = [x_1^2, \sqrt{2}x_1x_2, x_2^2]^T.$$

Assume all parameters are set to zero before running the algorithm.

Based on the table, what is the output of θ and θ_0?

(Enter θ_0 accurate to at least 2 decimal places.)

$\theta_0 =$ **Assessment content removed**

(Enter θ as a vector, enclosed in square brackets, and components separated by commas, e.g. type `[0,1]` for $\begin{bmatrix}0 & 1\end{bmatrix}^T$. Note that this sample vector input may not be of the same dimension of the answer. Enter each component accurate to at least 2 decimal places.)

from mitxgraders import * grader = MatrixGrader(answers='[21.00, -22.63, 22.00]', max$_array_dim$ = 1, $tolerance$ = 0.01)

$\theta =$ **Assessment content removed**

edXinclude: ../xml/standardnotation.xml

edXproblem: 2. (3)

Based on the calculation of θ and θ_0, does the decision boundary $\theta^T\phi(x) + \theta_0 = 0$ correctly classify all the points in the training dataset?

Assessment content removed

edXproblem: 2. (4)

Recall for $x = \begin{bmatrix}x_1 & x_2\end{bmatrix}^T$

$$\phi(x) = [x_1^2, \sqrt{2}x_1x_2, x_2^2]^T.$$

Define the kernel function

$$K(x, x') = \phi(x)^T\phi(x').$$

Write $K(x, x')$ as a function of the dot product $x \cdot x'$. To answer, let $z = x \cdot x'$, and enter $K(x, x')$ in terms of z.

$K(x, x') =$ **Assessment content removed**

edXinclude: ../xml/standardnotation.xml

edXvertical: Problem 3
edXtext: Problem 3

Stochastic gradient descent (SGD) is a simple but widely applicable optimization technique. For example, we can use it to train a Support Vector Machine. The objective function in this case is given by:

$$J(\theta) = \left[\frac{1}{n}\sum_{i=1}^{n} \text{Loss}_h(y^{(i)}\theta \cdot x^{(i)})\right] + \frac{\lambda}{2}\|\theta\|^2$$

where $\text{Loss}_h(z) = \max\{0, 1-z\}$ is the hinge loss function, $(x^{(i)}, y^{(i)})$ with for $i = 1, \ldots n$ are the training examples, with $y^{(i)} \in \{1, -1\}$ being the label for the vector $x^{(i)}$.

For simplicity, we ignore the offset parameter θ_0 in all problems on this page.

edXproblem: 3. (1)

The stochastic gradient update rule involves the gradient $\nabla_\theta \text{Loss}_h(y^{(i)}\theta \cdot x^{(i)})$ of $\text{Loss}_h(y^{(i)}\theta \cdot x^{(i)})$ with respect to θ.

*Hint:*Recall that for a k-dimensional vector $\theta = \begin{bmatrix}\theta_1 & \theta_2 & \cdots & \theta_k\end{bmatrix}^T$, the gradient of $f(\theta)$ w.r.t. θ is $\nabla_\theta f(\theta) = \begin{bmatrix}\frac{\partial f}{\partial \theta_1} & \frac{\partial f}{\partial \theta_2} & \cdots & \frac{\partial f}{\partial \theta_k}\end{bmatrix}^T$.)

Find $\nabla_\theta \text{Loss}_h(y\theta \cdot x)$ in terms of x.

(Enter `lambda` for λ, `y` for y and `x` for the vector x. Use `*` for multiplication between scalars and vectors, or for dot products between vectors. Use `0` for the zero vector.)

For $y\theta \cdot x \leq 1$:

$\nabla_\theta \text{Loss}_h(y\theta \cdot x) =$ **Assessment content removed**

For $y\theta \cdot x > 1$:

$\nabla_\theta \text{Loss}_h(y\theta \cdot x) =$ **Assessment content removed**

Let θ be the current parameters. What is the stochastic gradient update rule, where $\eta > 0$ is the learning rate? (Choose all that apply.)

$\theta \rightarrow$

Assessment content removed

Correction note: July 29 15:00UTC In the earlier version:

- The conditions for the first answer boxes were written $y\theta \cdot x < 1$ and $y\theta \cdot x \geq 1$ instead of the current $y\theta \cdot x \leq 1$ and $y\theta \cdot x > 1$.
- The summation in the choices had an error: $\sum_{i=1}^{n}$ was written wrongly as $\sum_{i=1^n}$.

Grader is correct: The grader behaves as intended in this problem. If you get an input error, please check your answers carefully. You will also need to complete all parts of the question before the submit button will be un-grayed.

Correction note: July 30 03:00UTC In the earlier version, the question statement did not include the input instruction " in term of x" nor "Use `0` for the zero vector."

Correction note: July 30 16:00UTC In the earlier version, the confirmation note that the grader is working was not included.

Grading Note: The original problem statement, before the correction, which divides the cases into $y\theta \cdot x < 1$ and $y\theta \cdot x \geq 1$ matches with the definition of hinge loss in lecture. The corrected version assigned the boundary case $y\theta \cdot x = 1$ differently. However, the main property of the hinge loss function is that the loss increases as its argument becomes more negative, and the boundary case if not important. Hence, even with this mismatch, we have decided to proceed with the grading as intended originally. If you had switch the orders of the inputs, the grader would have thrown an error, and the "Grader is correct" note was a reminder to check your answers in this case.

edXinclude: ../xml/standardnotation.xml

edXproblem: 3. (2)

Suppose the current parameter θ is as in the figure below:

Here, θ is in the direction of the arrow, the solid line represents the classifer defined by θ, and the dotted lines represent the positive and negative margin boundaries.

For large η (i.e. η close to 1) $0.5 < \eta\lambda < 1$, which of the following figure corresponds to a single SGD update made in response to the point labeled âĂŹ+âĂŹ above?

Assessment content removed

edXproblem: 3. (3)

Again for large η (i.e. η close to 1) and $0.5 < \eta\lambda < 1$, but now we perform a single SGD update made in response to a different point labeled âĂŹ+âĂŹ, shown below:

which of the following figure corresponds to a single SGD update made in response to the point labeled âĂŹ+âĂŹ above?

Assessment content removed

edXvertical: Problem 4
edXtext: Problem 4

For simplicity, suppose our rating matrix is a $2x2$ matrix and we are looking for a rank-1 solution UV^T so that user and movie features U and V are both $2x1$ matrices. The observed rating matrix has only a single entry:

$$Y = \begin{bmatrix} ? & 1 \\ ? & ? \end{bmatrix} \tag{1}$$

In order to learn user/movie features, we minimize

$$J(U,V) = \left(\frac{1}{2} \sum_{(a,i) \in D} (Y_{ai} - [UV^T]_{ai})^2 \right) + \lambda(U_1^2 + V_1^2) \tag{2}$$

where U_1 and V_1 are the first components of the vectors U and V respectively (if $U = [u_1, u_2]$, then $U_1 = u_1$), the set D is just the observed entries of the matrix Y , in this case just $(1, 2)$.

Note that the regularization we use applies only to the first coordinate of user/movie features . We will see how things get a bit tricky with this type of partial regularization.

Correction Note (July 30 21:00UTC): An earlier version does not include the clarifcation "where U_1 and V_1 are the first compoments of the vectors U and V respectively."

Correction Note (Aug 4 03:00UTC): Added an example of what U_1 means: if $U = [u_1, u_2]$, then $U_1 = u_1$).

edXproblem: 4. (1)

If we initialize $U = \begin{bmatrix} u & 1 \end{bmatrix}^T$, for some $u > 0$, what is the solution to the vector $V = \begin{bmatrix} v_1 & v_2 \end{bmatrix}^T$ as a function of λ and u?

(Enter V as a vector, enclosed in square brackets, and components separated by commas, e.g. type `[u,lambda+1]` if $V = \begin{bmatrix} u & \lambda + 1 \end{bmatrix}^T$.)

from mitxgraders import * grader = MatrixGrader(answers='[0,1/u]', variables=['u','lambda'], $\max_{a}rray_{d}im = 1, sample_{f}rom = 'u' : RealInterval([1, 5]),' lambda' : RealInterval([1, 5]), tolerance ='$ 2)

$V =$ **Assessment content removed**

edXinclude: ../xml/standardnotation.xml

edXproblem: 4. (2)

What is the resulting value of $J(U, V)$ as a function of λ and u?

(Type `lambda` for λ).

Assessment content removed

edXinclude: ../xml/standardnotation.xml

edXproblem: 4. (3)

If we continue to iteratively solve for U and V, what would U and V converge to?

Assessment content removed

Correction Note (July 29): In an earlier version, V was missing in all choices.

edXproblem: 4. (4)

Not all rating matrices Y can be reproduced by UV^T when we restrict the dimensions of U and V to be 2×1.

For each matrix below, answer "Yes" or "No" according to whether it can be reproduced by such U and V of size 2×1.

$Y = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$

Assessment content removed

$Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

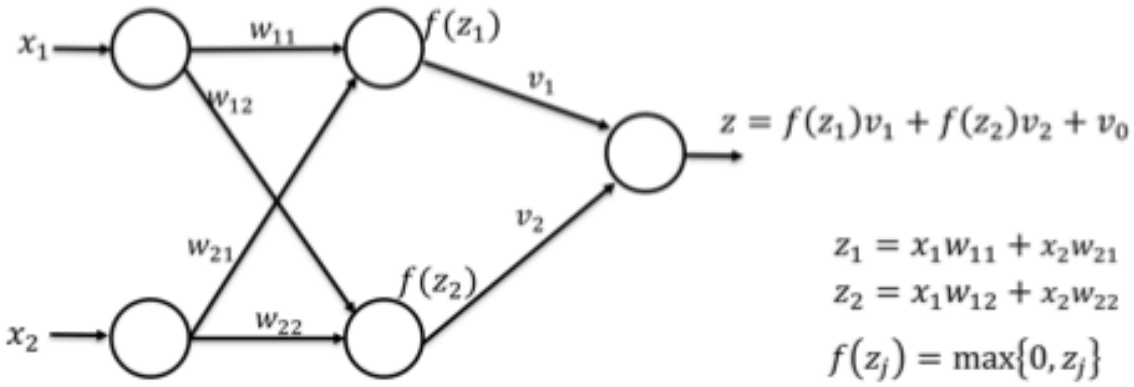
Assessment content removed

$Y = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$

Assessment content removed

edXvertical: Problem 5
edXtext: Problem 5

Consider a 2-layer feed-forward neural network that takes in $x \in \mathbb{R}^2$ and has two ReLU hidden units as defined in the figure below. **Note that hidden units have no offset parameters in this problem.**



chch

edXproblem: 5. (1)

The values of the weights in the hidden layer are set such that they result in the z_1 and z_2 "classifiers" as shown in the (x_1, x_2)**-space** in the figure below:

The z_1 "classifier" with the normal $w_1 = \begin{bmatrix} w_{11} & w_{21} \end{bmatrix}^T$ is the line given by $z_1 = x \cdot w_1 = 0$.
Similarly, the z_2 "classifier" with the normal $w_2 = \begin{bmatrix} w_{12} & w_{22} \end{bmatrix}^T$ is the line given by $z_2 = x \cdot w_2 = 0$.
The arrows labeled w_1 and w_2 point in the **positive** directions of the respective normal vectors.
The regions labeled I, II, III, IV **are the 4 regions defined by these two lines not including the boundaries.**

Choose the region(s) in (x_1, x_2) space which are mapped into each of the following regions in (f_1, f_2)-space, the 2-dimensional space of hidden unit activations $f(z_1)$ and $f(z_2)$. (For example, for the second column below, choose the region(s) in (x_1, x_2) space which are mapped into the f_1-axis in (f_1, f_2)-space.)

(Choose all that apply for each column.)

$\{(f_1, f_2) : f_1 > 0,\ f_2 > 0\}$: (Choose all that apply.)	f_1-axis:	f_2-axis:
Assessment content removed	**Assessment content removed**	**Assessment content removed**

Correction Note (July 30 03:00UTC): In an earlier version, the problem statement did not

include the emphasis "in the (x_1, x_2)**-space** " in the first sentence.

Correction Note (August 1 13:00UTC): In an earlier version, the caption under the figure did not include "not including the boundaries".

Correction Note (August 4 04:00UTC): Added "(For example, for the second column below, choose the region(s) in (x_1, x_2) space which are mapped into the f_1-axis in (f_1, f_2)-space.)"

Grading Note: Since we have separately asked for the region(s) that map(s) into the origin (in the fourth column), and we do not want to penalize on the interpretation on with axes include the origin, we will accept the region II as the answer for the f_1-axis and region IV as the answer for the f_2-axis.

edXproblem: 5. (2)

If we keep the hidden layer parameters above fixed but add and train additional hidden layers (applied after this layer) to further transform the data, could the resulting neural network solve this classification problem?

Assessment content removed

Suppose we stick to the 2-layer architecture but add many more ReLU hidden units, all of them without offset parameters. Would it be possible to train such a model to perfectly separate these points?

Grading Note: Since there is ambiguity as to whether some data points lie on the same line through the origin, we will accept both answers as correct.

Assessment content removed

edXproblem: 5. (3)

Which of the following statements is correct?

1. The gradient calculated in the backpropagation algorithm consists of the partial derivatives of the loss function with respect to each network weight.

   Assessment content removed

2. Initialization of the parameters is often important when training large feed-forward neural networks.

   If weights in a neural network with sigmoid units are initialized all the weights to close to zero values, then during early stochastic gradient descent steps, the network represents a nearly linear function of the inputs.

   Assessment content removed

3. On the other hand, if we randomly set all the weights to very large values, or donâĂŹt scale them properly with the number of units in the layer below, then the sigmoid units would behave like sign units.

   (Note that a sign unit is a unit with activation function $\text{sign}(x) = 1$ if $x > 0$ and $\text{sign}(x) = -1$ if $x < 0$. For the purpose of this question, it does not matter what $\text{sign}(0)$ is.)

   Assessment content removed

   Grading Note: Since the question did not specify whether "behave like sign units" allows for shifting or rescaling of the sign function, both "True" and "False" are accepted as correct.

4. If we use only sign units in a feedforward neural network, then the stochastic gradient descent update will

   Assessment content removed

5. Stochastic gradient descent differs from (true) gradient descent by updating only one network weight during each gradient descent step.

   Assessment content removed

Correction note (July 31 16:00UTC):. In the earlier version, the sign unit definition was not included.

edXproblem: 5. (4)

There are many good reasons to use convolutional layers in CNNs as opposed to replacing them with fully connected layers. Please check T or F for each statement.

Since we apply the same convolutional filter throughout the image, we can learn to recognize the same feature wherever it appears.

Assessment content removed

A fully connected layer for a reasonably sized image would simply have too many parameters

Assessment content removed

A fully connected layer can learn to recognize features anywhere in the image even if the features appeared preferentially in one location during training

Assessment content removed