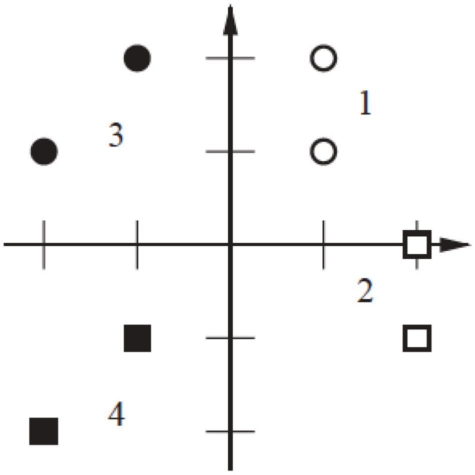
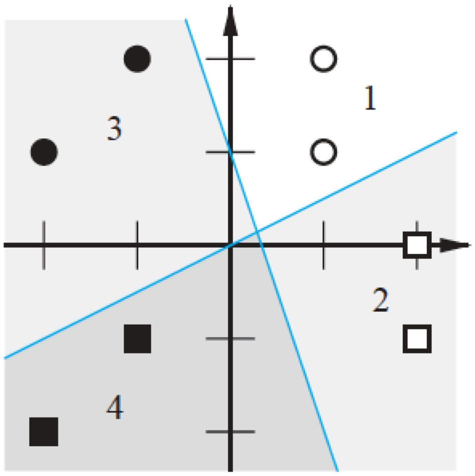
1. For the following classification problem, design a single-layer perceptron that has zero error on training set.

$$
\begin{aligned}
C_1 &= \left\{ \mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix},\ \mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right\} \\
C_2 &= \left\{ \mathbf{x}_3 = \begin{bmatrix} 2 \\ -1 \end{bmatrix},\ \mathbf{x}_4 = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \right\} \\
C_3 &= \left\{ \mathbf{x}_5 = \begin{bmatrix} -1 \\ 2 \end{bmatrix},\ \mathbf{x}_6 = \begin{bmatrix} -2 \\ 1 \end{bmatrix} \right\} \\
C_4 &= \left\{ \mathbf{x}_3 = \begin{bmatrix} -1 \\ -1 \end{bmatrix},\ \mathbf{x}_4 = \begin{bmatrix} -2 \\ -2 \end{bmatrix} \right\}
\end{aligned}
$$

Solution: The patterns are shown in the following figure:



And tentative decision boundaries can be those in the following figure:



It is left to the student to verify that a single layer perceptron with two neurons can classify the vectors given that the four classes are modeled using vectors

$$C_1 \rightarrow \mathbf{t}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$$C_2 \rightarrow \mathbf{t}_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$C_3 \rightarrow \mathbf{t}_3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$C_4 \rightarrow \mathbf{t}_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

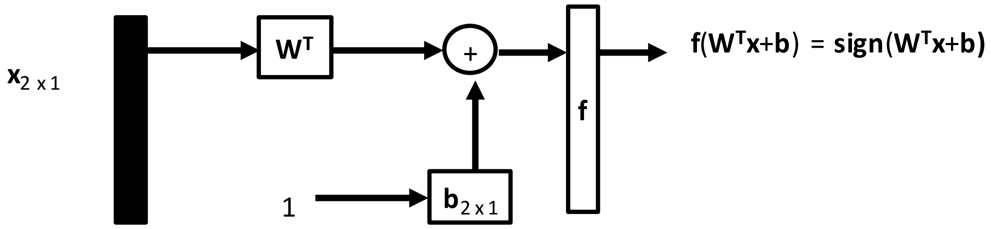
and the weight vectors are columns of the following matrix

$$\mathbf{W} = \begin{bmatrix} -3 & 1 \\ -1 & -2 \end{bmatrix}$$

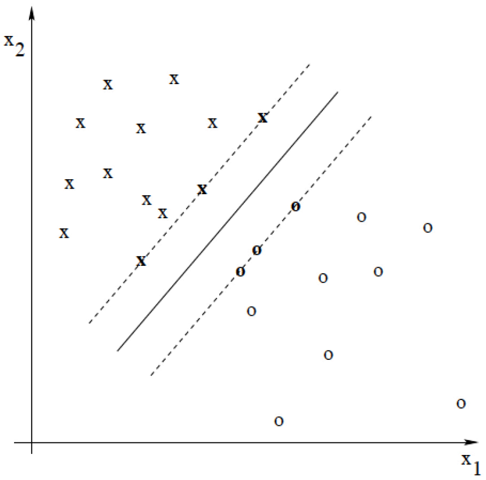
and the biases in the vector:

$$\mathbf{b} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The architecture of the Perceptron is:


$\mathbf{x}_{2 \times 1}$
$\mathbf{W}^T$
+
f
$\mathbf{f}(\mathbf{W}^T\mathbf{x}+\mathbf{b}) = \mathbf{sign}(\mathbf{W}^T\mathbf{x}+\mathbf{b})$
1
$\mathbf{b}_{2 \times 1}$

2. Answer the following questions about the figure:



(a) What is the leave-one-out cross-validation error estimate for maximum margin separation in the figure? (The answer is a number)

(b) True or False? We would expect the support vectors to remain the same in general as we move from a linear kernel to higher order polynomial kernels.

(c) If each of the classes is modeled with a normal distribution and the variances are assumed equal, how many parameters are needed to describe the decision boundary for classification?

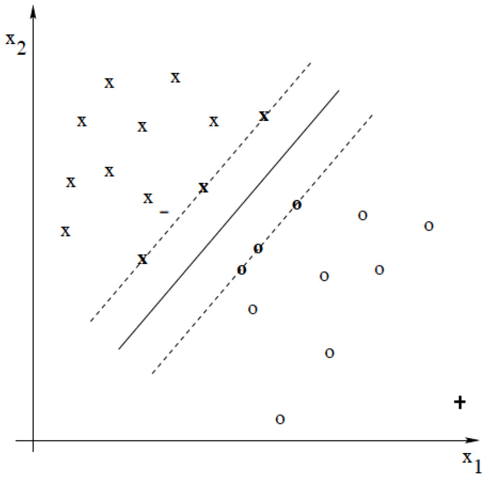
(d) Assume that all data points *outside* the margin are unlabeled. Mark on the figure with a - the first data point that has to be labeled, when implementing active learning. Also, mark on the figure with a + the first data point that has to be labeled, when implementing self-training.

Solution:

(a) Based on the figure we can see that removing any single point would not chance the resulting maximum margin separator. Since all the points are initially classified correctly, the leave-one-out error is zero.

(b) There are no guarantees that the support vectors remain the same. The feature vectors corresponding to polynomial kernels are non-linear functions of the original input vectors and thus the support points for maximum margin separation in the feature space can be quite different.

(c) Modeling classes with normal densities with euqal variances creates linear decision boundaries. A linear decision boundary with two variables x_1 and x_2 is described by three parameters (w_0, w_1, w_2), i.e.: $g(x_1, x_2) = w_0 + w_1x_1 + w_2x_2$.

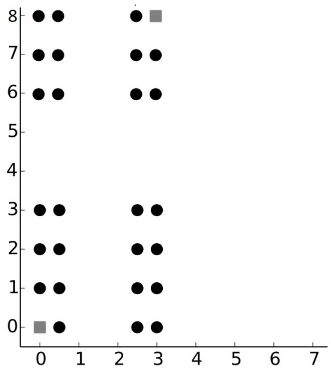


(d)

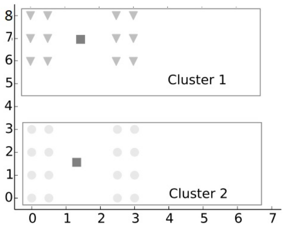
3. Consider the unlabeled two-dimensional data represented on the following figure.

   (a) Using the two points marked as squares as initial centroids, draw (on that same figure) the clusters obtained after one iteration of the k-means algorithm ($k = 2$).

   (b) Does your solution change after another iteration of the k-means algorithm?



Solution:



(a)

(b) No.

4. A company with headquarters in the Bay Area has two offices in Los Angeles and San Diego. An employee in San Diego office is sent to the Los Angeles office the next day with probability 0.2 and stays in San Diego office with probability 0.8. An employee in Los Angeles office is sent to the San Diego office with probability 0.1 and stays in Los Angeles office with probability 0.9. A new employee is assigned to Los Angeles office with probability 0.7 and to San Diego office with probability 0.3. An employee in San Diego office works between six and eight hours per day with probability 0.6, works more than eight hours with probability 0.3, and works less than six hours per day with probability 0.1. An employee in Los Angeles office works between six and eight hours per day with probability 0.15, works more than eight hours with probability 0.8, and works less than six hours per day with probability 0.05. A manager in the headquarters can only observe the number of hours each employee worked each day.

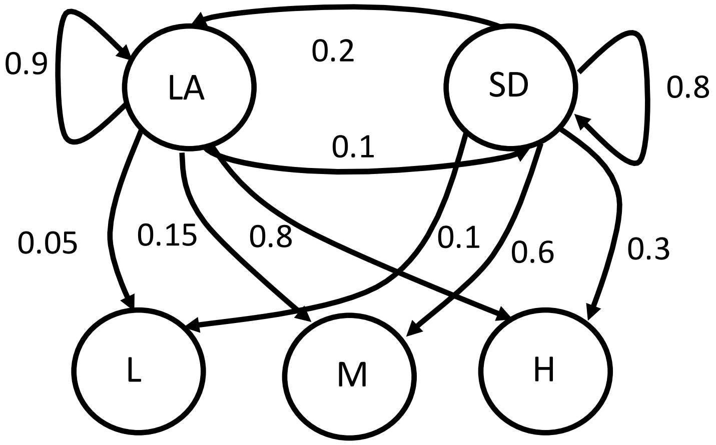
   (a) Construct a Hidden Markov Model that models the observations of the manager in their headquarters. Clearly show the parameters with matrices and vectors and draw a state transition graph for the model.

   (b) If the manager observes the number of hours a new employee worked in the first three consecutive days of work to be $5, 9, 7$, what is the most likely sequence of places at which the employee worked in those three days?

   (c) What sequence of three places has the maximum expected number of correct places?

Solution:

(a) The figure below shows the HMM model of the problem. We used L, M, H to show working less than six hours, between six and eight hours, and more than eight hours per day, respectively.



The parameters will be:

$$\pi_{x_0} = \begin{bmatrix} 0.7 & 0.3 \end{bmatrix}$$
$$A = \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix}$$
$$B = \begin{bmatrix} 0.05 & 0.15 & 0.8 \\ 0.1 & 0.6 & 0.3 \end{bmatrix}$$

where the rows show LA and SD.

(b) The sequence of observations is L, H, M. We do not need to use the Viterbi algorithm here. An exhaustive search is easy:

State	Probability
LA, LA, LA	$0.7 \times 0.9 \times 0.9 \times 0.05 \times .8 \times 0.15 = 0.003402$
LA, LA, SD	$0.7 \times 0.9 \times 0.1 \times 0.05 \times .8 \times 0.6 = 0.001512$
LA, SD, SD	$0.7 \times 0.1 \times 0.8 \times 0.05 \times .3 \times 0.6 = 0.000504$
LA, SD, LA	$0.7 \times 0.1 \times 0.2 \times 0.05 \times .3 \times 0.15 = 0.0000315$
SD, SD, SD	$0.3 \times 0.8 \times 0.8 \times 0.1 \times .3 \times 0.6 = \mathbf{0.003456}$
SD, LA, SD	$0.3 \times 0.2 \times 0.1 \times 0.1 \times .8 \times 0.6 = 0.000288$
SD, SD, LA	$0.3 \times 0.8 \times 0.2 \times 0.1 \times .3 \times 0.15 = 0.000216$
SD, LA, LA	$0.3 \times 0.2 \times 0.9 \times 0.1 \times .8 \times 0.15 = 0.000648$

Therefore, the most likely sequence of places is SD, SD, SD.

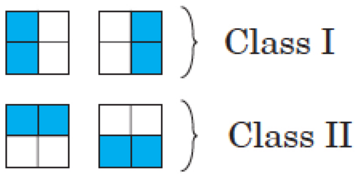
(c) For the maximum number of correct states (Expectation Maximization) we calculate the probability of each of states at each step from the table above:[1]

[1] Note that in principle, the probabilities have to be normalized and then summed up, because we are essentially calculating the conditional probability of each state occuring at each time step, *given the sequence of observations.* Nevertheless, normalization will not change the solution, so for simplicity, we do not do it.

State	0	1	2
$P(LA\|LHM)$	**0.5418345**	**0.5816555**	0.4272931
$P(SD\|LHM)$	0.458166	0.4183445	**0.5727069**

Hence, the answer using this method is LA, LA, SD.

5. Consider the two classes of patterns that are shown in the figure below. Design a multilayer network to distinguish these categories.

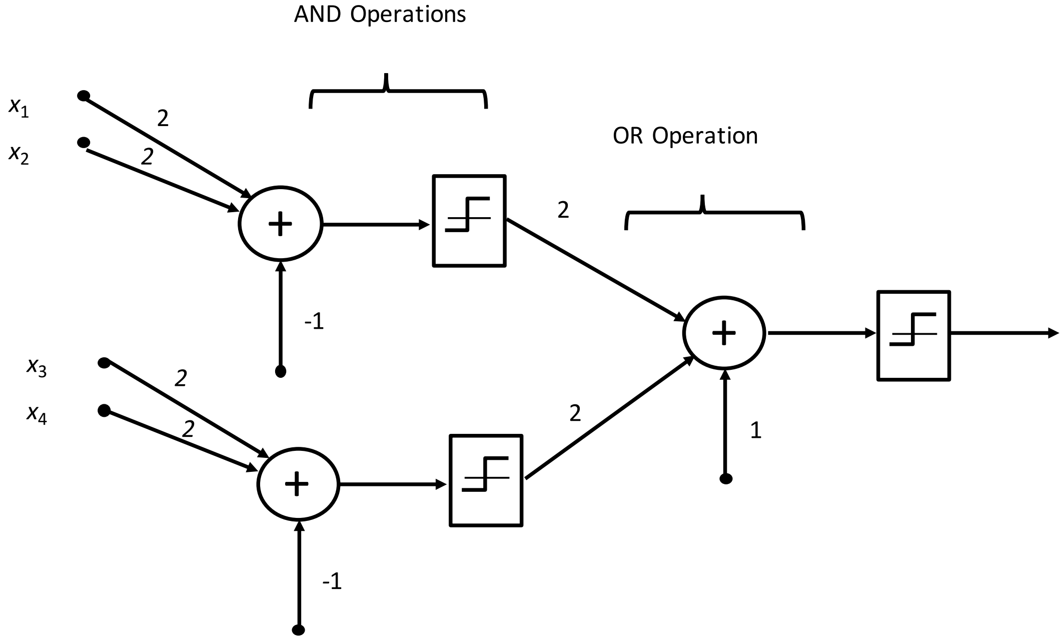


Solution:

There are many different multilayer networks that could solve this problem. The vectors in each class can be represented as:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{x}_3 = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{x}_4 = \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

We will design a network by first noting that for the Class I vectors either the first two elements or the last two elements will be "1". The Class II vectors have alternating "1" and "-1" patterns. This leads to the network shown in the figure below. Note that x_1, x_2, x_3, x_4 denote the four elements in each of the vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$.



The first neuron in the first layer tests the first two elements of the input vector. If they are both "1" it outputs a "1", otherwise it outputs a "-1". The second neuron in the first layer tests the last two elements of the input vector in the same way. Both of the neurons in the first layer perform AND operations. The second layer of the network tests whether either of the outputs of the first layer are "1". It performs an OR operation. In this way, the network will output a "1" if either the first two elements or the last two elements of the input vector are both "1".