CSCI-4/5588, Fall 2020
Machine Learning II
Study Guide for Test#2

(Please don't distribute this study guide.
The guide is for your study purpose only)

---------- **Chapter 03: SVM** ----------

1. While considering separable classes (see Figure (A)) for support vector machine development, we formulate our goal as:

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 \qquad \textbf{(1)}$$

subject to $y_i(\beta^T x_i + \beta_0) \geq 1$, ***i =1, ..., N.***

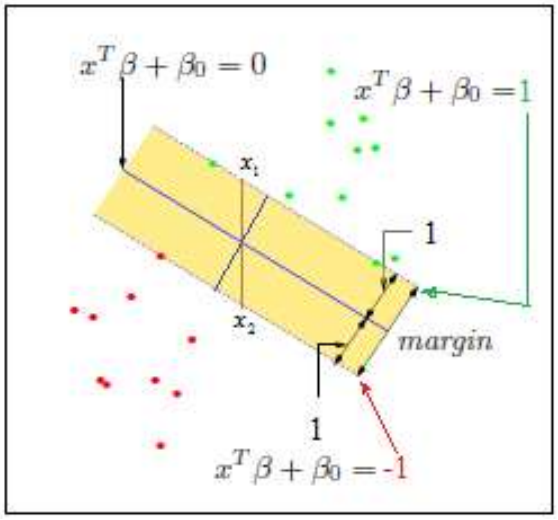


Figure A: Support vector classifier. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width, 1+1 = 2.

Show that the *Lagrange* (primal, L_p) function, to be minimized w.r.t. β and β_0, can be written as:

$$L_p(\beta,\beta_0,\alpha_i) = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N}\alpha_i\left[y_i(x_i{}^T\beta + \beta_0) - 1\right] \qquad \textbf{(2)}$$

Here, α_i is the Lagrange multiplier and $\alpha_i \geq 0$.

From (2), derive the dual

$$L_D(\alpha_i) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N}\alpha_i\alpha_k y_i y_k x_i{}^T x_k$$

Ans: It is to be noted that there is a minus sign in front of the ***Lagrange multiplier*** (equation #2) term because we are minimizing with respect to β and β_0 but maximizing with respect to α_i.

Setting the derivatives to zero, we obtain:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{3}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i \tag{4}$$

[Important: **Note here you need to show how we can obtain Equation #3 and #4, check class notes/video**]
and substituting these in (2) we obtain the so-called Wolfe dual

$$\begin{aligned} L_p(\beta, \beta_0, \alpha_i) &= \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i (x_i^T \beta + \beta_0) - 1 \right] \\ &= \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i y_i x_i^T \beta - \sum_{i=1}^{N} \alpha_i y_i \beta_0 + \sum_{i=1}^{N} \alpha_i \\ &= \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k - \beta_0 \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i \\ &= \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k - \sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k + \sum_{i=1}^{N} \alpha_i \qquad \left[\because \sum_{i=1}^{N} \alpha_i y_i = 0 \right] \\ &= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k \end{aligned}$$

Therefore, we can now express dual equation as a function of α_i as:

$$L_D(\alpha_i) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

2. Consider a feature space with two inputs X_1 and X_2, and a polynomial kernel of degree 2. That is

$$K(X, X') = (1 + \langle X, X' \rangle)^2 = (1 + X^T X')^2 \qquad [\text{where, } X^T = [X_1 \quad X_2],\ X' = \begin{bmatrix} X_1' \\ X_2' \end{bmatrix}]$$

Show that the kernel function can be expressed as an inner product in a feature space having a mapping from 2 dimensions to 6 dimensions (i.e., $\Re^2 \Rightarrow \Re^6$).

Ans:

$$K(X,X') = (1+\langle X,X'\rangle)^2 = (1+X^T X')^2 \qquad [\because\ X^T = [X_1 \quad X_2],\ X' = \begin{bmatrix} X_1' \\ X_2' \end{bmatrix}]$$

$$= (1 + X_1X_1' + X_2X_2')^2$$

$$= 1 + 2X_1X_1' + 2X_2X_2' + (X_1X')^2 + 2X_1X_1'X_2X_2' + (X_2X_2')^2$$

$$= [1 \quad \sqrt{2}X_1 \quad \sqrt{2}X_2 \quad X_1^2 \quad \sqrt{2}X_1X_2 \quad X_2^2] \begin{bmatrix} 1 \\ \sqrt{2}X_1' \\ \sqrt{2}X_2' \\ X_1'^2 \\ \sqrt{2}X_1'X_2' \\ X_2'^2 \end{bmatrix}$$

$$= h(X)^T\ h(X')$$

This kernel function, therefore, represents an inner product in a feature space having a mapping from 2 dimensions to 6 dimensions (i.e., $\Re^2 \Rightarrow \Re^6$), in which the mapping from input space to feature space is described by the vector function $h(X)$. The components of $h(X)$, are $h_1(X) = 1$, $h_2(X) = \sqrt{2}X_1$, $h_3(X) = \sqrt{2}X_2$, $h_4(X) = X_1^2$, $h_5(X) = \sqrt{2}X_1X_2$, and $h_6(X) = X_2^2$.

3. Show that the redial basis function (RBF) or Gaussian kernel, i.e., $\exp(-\gamma\|x - x'\|^2)$, can be expressed as an inner product in infinite-dimensional space. Assuming, $x \in \Re^1$ and $\gamma > 0$.

Ans:

$$K(x, x') = \exp(-\gamma\|x - x'\|^2)$$

$$= \exp(-\gamma(x - x')^2)$$

$$= \exp(-\gamma x^2 + 2\gamma xx' - \gamma x'^2)$$

$$= \exp(-\gamma x^2).\exp(-\gamma x'^2).\exp(2\gamma xx')$$

$$= \exp(-\gamma x^2).\exp(-\gamma x'^2).\left(1 + \frac{2\gamma xx'}{1!} + \frac{(2\gamma xx')^2}{2!} + \frac{(2\gamma xx')^3}{3!} + \ldots\right)$$

$$\left[\because e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots\right]$$

$$= \exp(-\gamma x^2).\exp(-\gamma x'^2).\left(1.1 + \sqrt{\frac{2\gamma}{1!}}x.\sqrt{\frac{2\gamma}{1!}}x' + \sqrt{\frac{(2\gamma)^2}{2!}}x^2.\sqrt{\frac{(2\gamma)^2}{2!}}x'^2 + \sqrt{\frac{(2\gamma)^3}{3!}}x^3.\sqrt{\frac{(2\gamma)^3}{3!}}x'^3 + \ldots\right)$$

$$= \left(\exp(-\gamma x^2).[1 \quad \sqrt{\frac{2\gamma}{1!}}x \quad \sqrt{\frac{(2\gamma)^2}{2!}}x^2 \quad \ldots]\right) \left(\exp(-\gamma x'^2).\begin{bmatrix} 1 \\ \sqrt{\frac{2\gamma}{1!}}x' \\ \sqrt{\frac{(2\gamma)^2}{2!}}x'^2 \\ \vdots \end{bmatrix}\right)$$

$$= h(x)^T\, h(x')$$

4. What are the usual steps to be followed for prediction using SVM.

Ans:
Steps to follow:

- Transform data to the format of an SVM package
- Conduct simple scaling on the data
- Consider the RBF kernel K(x; y) = $\exp(-\gamma\|x - y\|^2)$
- Use cross-validation to find the best parameter C and γ
- Use the best parameter C and γ to train the whole training set
- Test

---------- **Chapter 04: ANN** ----------

5. Draw the characteristic curve of (a) sigmoid function and (b) hyperbolic tangent function (see class note or, find it by yourself)

6. For the given Artificial Neural Network below write the vector-equations involved in forward propagation:

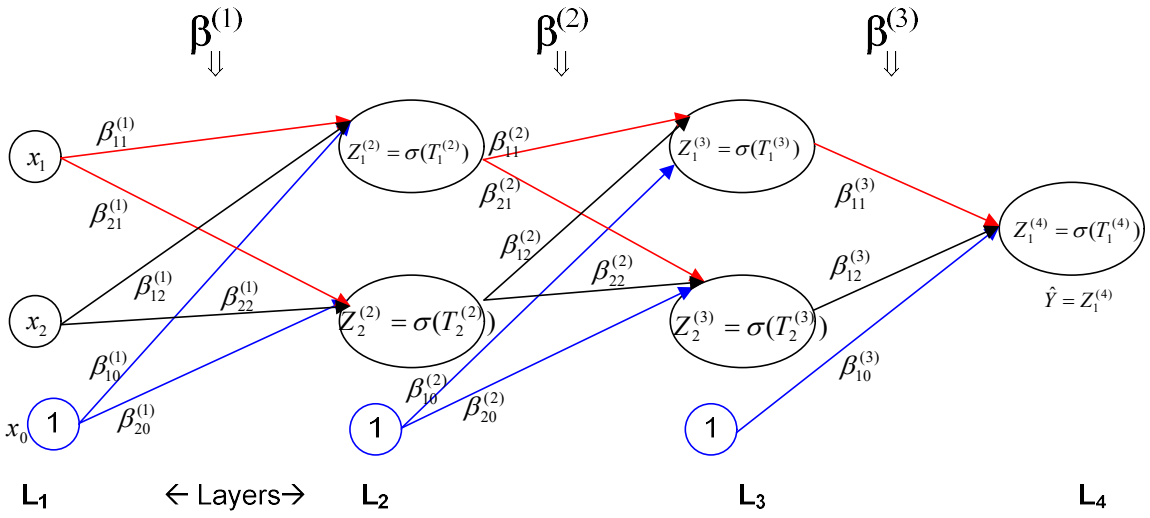


Figure: A multilayer neural network demonstrating the notations.

Assume that the transformation function σ = f_{sig} and bias unit (input "1") in each layer is presented as $Z_0^{(l)}$, where *l* is the layer number.

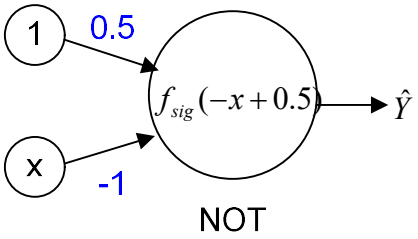
Ans:

$$Z^{(1)} = X$$
$$T^{(2)} = \beta^{(1)^T} Z^{(1)}$$
$$Z^{(2)} = f_{sig} \bullet (T^{(2)}) \quad \text{and add } Z_0^{(2)}$$
$$T^{(3)} = \beta^{(2)^T} Z^{(2)}$$
$$Z^{(3)} = f_{sig} \bullet \left(T^{(3)}\right) \text{ and add } Z_0^{(3)}$$
$$T^{(4)} = \beta^{(3)^T} Z^{(3)}$$
$$Z^{(4)} = \hat{Y} = f_{sig} \bullet (T^{(4)})$$

[Note: Don't forget the '.'s (dots) in the above equations, they indicate element-wise operations**]**

7. Design (i) NOT, (ii) OR and (iii) AND gates using a single neuron model and show the operation using truth table.

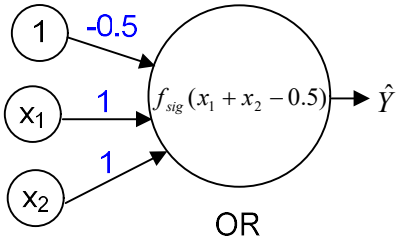
Ans:



x	$f_{sig}(-x+0.5)$	$\hat{Y}$
0	f_{sig} (+0.5)	1
1	f_{sig} (-0.5)	0

(a) (b)

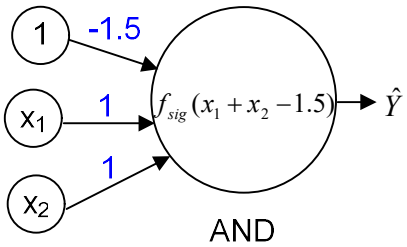
Figure A: Modeling (a) NOT and the corresponding (b) truth-table.



x_1	x_2	$f_{sig}(x_1+x_2-0.5)$	$\hat{Y}$
0	0	f_{sig} (-0.5)	**0**
0	1	f_{sig} (+0.5)	**1**
1	0	f_{sig} (+0.5)	**1**
1	1	f_{sig} (+1.5)	**1**

(a) (b)

Figure B: Modeling (a) OR and the corresponding (b) truth-table.



x_1	x_2	$f_{sig}(x_1+x_2-1.5)$	$\hat{Y}$
0	0	f_{sig} (-1.5)	**0**
0	1	f_{sig} (-0.5)	**0**
1	0	f_{sig} (-0.5)	**0**
1	1	f_{sig} (+0.5)	**1**

(a) (b)

Figure C: Modeling (a) AND and the corresponding (b) truth-table.

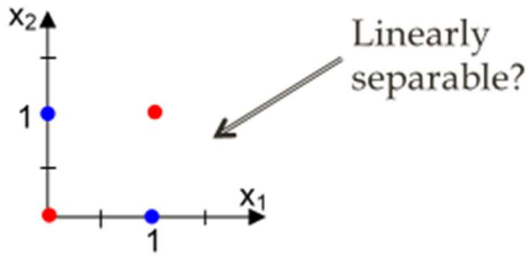
8. Show why logic function XOR is not linearly separable. Design an ANN for a two-input XOR function (See chapter # 4 (ANN), page 18-19 and class-notes).

Ans:

Hidden Layer, When? …

XOR

x_1	x_2	$\hat{Y}$
0	0	0
0	1	1
1	0	1
1	1	0



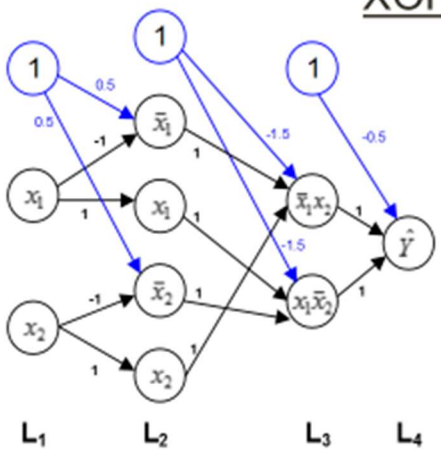
We can extend the truth-table to use the linearly separable logic functions, right?

x_1	x_2	$(\bar{x}_1 x_2 + x_1 \bar{x}_2)$	$\hat{Y}$
0	0	0 + 0	0
0	1	1 + 0	1
1	0	0 + 1	1
1	1	0 + 0	0

33

Hidden Layer, When? …

XOR



x_1	x_2	$(\bar{x}_1 x_2 + x_1 \bar{x}_2)$	$\hat{Y}$
0	0	0 + 0	0
0	1	1 + 0	1
1	0	0 + 1	1
1	1	0 + 0	0

Answer: 'Hidden layer', when the classes are **NOT** linearly separable.

34

9. Explain what should be the starting or initial values of the weight of an ANN. (See chapter #4 (ANN), page 28 …)

Ans: Starting values:

- If the weights are near zero, then the operative part of the sigmoid is roughly linear, and hence the neural network collapses into an approximately linear model. The use of exact zero weights leads to zero derivatives and perfect symmetry, and the algorithm never moves. Starting instead with large weights often leads to poor solutions.

- Usually starting values for weights are chosen to be random values near zero. Hence the model starts out nearly linear and becomes nonlinear as the weights increase.

- With standardized inputs, it is typical to take random uniform weights over the range [−0.7, +0.7].

10. Explain what should be the number of hidden units and layers in an ANN (see Chapter #4 (ANN), page 29 …).

11. Write down the delta-rule or, error back-propagation algorithm.

Ans: Algorithm: Error Back-propagation

BEGIN

1. From a data point (x_i, y_i), apply an input vector x_i to the network and forward propagate through the network and find the output error E. Then to perform the following steps to computation the rate of change of error w.r.t the network weights βs to compute the next value of βs.

2. For each of the output node/unit compute the error term δ^L:

$$\delta^{(L)} = (Z_k^{(L)} - Y_k)\, Z_k^{(L)}(1 - Z_k^{(L)})$$

3. For each of the hidden layer node /unit compute the error term $\delta^{(L-1)}$:

$$\delta^{(L-1)} = Z^{(L-1)}(1 - Z^{(L-1)}) \times \sum_{k=1}^{K} \delta^{(L)} \times \beta^{(L-1)}$$

And Compute: $\delta^{(L-2)}, \delta^{(L-3)}, ..., \delta^{(2)}$, except $\delta^{(1)}$ because it is the input layer and we do not want to change the original input data.

4. Update the weights (β) of the network:

$$\beta^{(i)}(t+1) = \beta^{(i)}(t) - \alpha\, \delta^{(i+1)}\, Z^{(i)}$$
$$\beta_0^{(i)}(t+1) = \beta_0^{(i)}(t) - \alpha\, \delta^{(i+1)} \qquad \text{[For bias terms]}$$

where, i=*1*, *2*, …, (*L*-1)

5. Go to Step 1 to loop or, exit if the exit-condition is met.

END

--- x ---