INTRODUCTION TO MACHINE LEARNING COMPSCI 4ML3

Lecture 17

Hassan Ashtiani

THE VERSION WITH "BIAS"

Hard-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$

solve:

$$(\mathbf{w}_0, b_0) = \underset{(\mathbf{w},b)}{\operatorname{argmin}} \|\mathbf{w}\|^2 \text{ s.t. } \forall i, \; y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \qquad (15.2)$$

output: $\hat{\mathbf{w}} = \frac{\mathbf{w}_0}{\|\mathbf{w}_0\|}, \quad \hat{b} = \frac{b_0}{\|\mathbf{w}_0\|}$

SOFT-MARGIN SVM

Soft-SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
parameter: $\lambda > 0$
solve:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \xi_i \right)$$

$$\text{s.t. } \forall i, \quad y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0$$

output: $\mathbf{w}, b$

$$\min_{\mathbf{w}} \left(\lambda \|\mathbf{w}\|^2 + L_S^{\text{hinge}}(\mathbf{w}) \right),$$

$$L_S^{\text{hinge}}(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^{m} \max\{0, 1 - y\overline{\langle \mathbf{w}, \mathbf{x}_i \rangle}\}.$$

- What if we need a non-linear classifier?

SVM WITH BASIS FUNCTIONS

$\phi(x): \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$

- Soft SVM in our notation

$$\min_{w} \frac{1}{n} \sum max\{0, 1 - y^i < w, x^i >\} + \lambda \|w\|_2^2$$

$w \in \mathbb{R}^{d_1}$

- Soft SVM with basis functions

$$\min_{w} \frac{1}{n} \sum max\{0, 1 - y^i < w, \phi(x^i) >\} + \lambda \|w\|_2^2$$

$w \in \mathbb{R}^{d_2}$

- $\phi(x^i)$ (and w) can be high-dimensional
  - How to deal with the prohibitive computational cost?

REPRESENTOR THEOREM

- $W^* = \underset{W}{\mathbf{argmin}} \frac{1}{n} \sum max\{0, 1 - y^i < w, \phi(x^i) >\} + \lambda \|w\|_2^2$
- THEOREM: THERE ARE REAL-VALUES $a_1, \dots, a_m$ SUCH THAT

$$W^* = \sum a_i \phi(x^i).$$

$a^T \phi$ where $\phi = [\phi(x^1) \cdots \phi(x^n)]^T$

- INSTEAD OF d_2 PARAMS, CAN USE n PARAMS.
  - BETTER IF $d_2 \gg n$.

KERNELIZED SVM - PREDICTION

- $W^* = \underset{W}{\mathbf{argmin}} \frac{1}{n} \sum \boldsymbol{max}\{0, 1 - y^i < w, \phi(x^i) >\} + \lambda \|w\|_2^2$
- $W^* = \sum a_i \phi(x^i)$.
- $K(x, z) = < \phi(x), \phi(z) >$ efficiently computable
- $K(x) = (K(x, x^1), \dots, K(x, x^n))$.
- PREDICTION FOR A NEW TEST POINT USING a AND K:

$$K_{n \times n} = \begin{array}{cc} & j \\ i & \begin{bmatrix} & K(x^i, x^j) \end{bmatrix} \end{array}$$

$$\hat{y} = sgn(\langle w^*, \phi(x) \rangle) = sgn(\langle \sum a_i \phi(x^i), \phi(x) \rangle)$$

$$= sgn(\sum a_i \underbrace{\langle \phi(x^i), \phi(x) \rangle}_{K(x, x^i)}) = sgn(a^T K(x))$$

KERNELIZED SVM (LEARNING)

$w^T w$

- $W^* = \underset{W}{\mathbf{argmin}} \frac{1}{n} \sum max\{0, 1 - y^i < w, \phi(x^i) >\} + \lambda \|w\|_2^2$

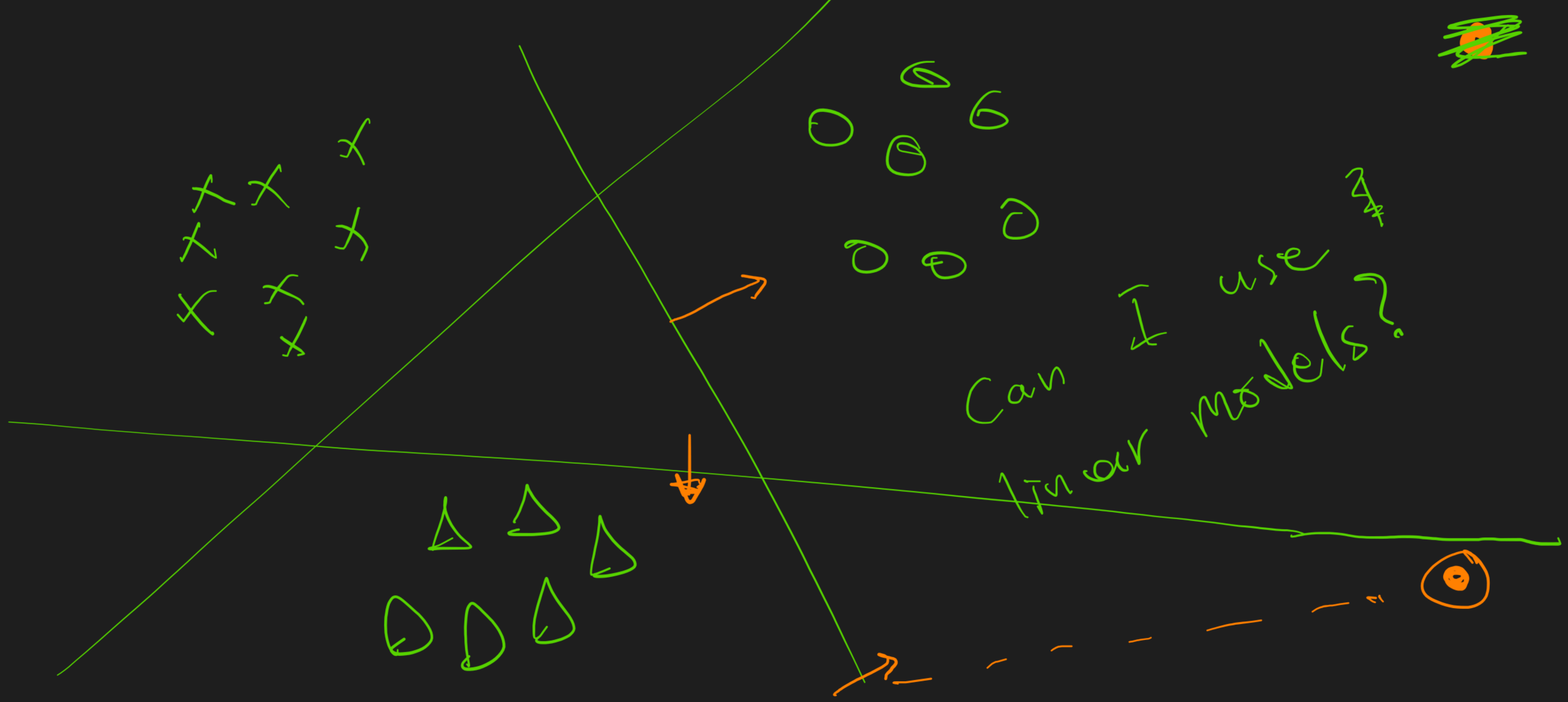
$$\underset{a}{\text{argmin}} \left[\frac{1}{n} \sum_i \max\{0, 1 - y^i \langle \sum a_j \phi(x^j), \phi(x^i) \rangle\} + \lambda \left\| \sum a_i \phi(x^i) \right\|_2^2 \right.$$

$$= \underset{a}{\text{argmin}} \left[\frac{1}{n} \sum_i \max\{0, 1 - y^i \langle a, K(x^i) \rangle\} + \lambda a^T K a \right.$$

$a \in R^n$

$$\langle \sum a_i \phi(x^i), \sum a_j \phi(x^j) \rangle = \sum_{i,j} a_i a_j \langle \phi(x^i), \phi(x^j) \rangle = a^T K a$$

LINEAR MODELS FOR MULTICLASS CLASSIFICATION



ONE-VERSUS-ALL CLASSIFICATION

- TRAIN k DIFFERENT BINARY CLASSIFIERS
  - CLASSIFIER i DISTINGUISHES SAMPLES FROM CLASS i VERSUS ALL OTHER CLASSES
  - $h_i(x) = \mathrm{SGN}(< w_i, x >)$
- NOW FOR A NEW TEST POINT x
  - $h^{one-verus-all}(x) = argmax_i(< w_i, x >)$

k binary classifiers

ALL-PAIRS CLASSIFICATION

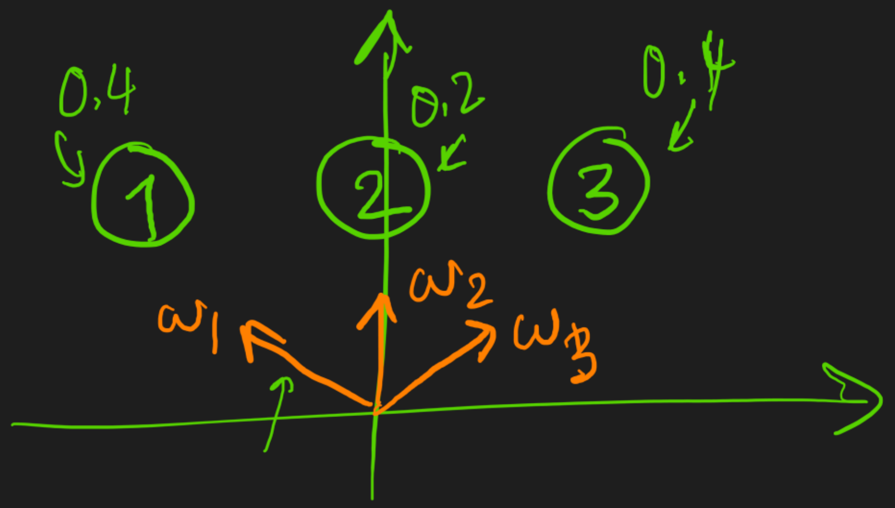
- For each distinct $i, j \in \{1,2,\dots,k\}$
  - Train a classifier to distinguish samples from class i and samples from class j
  - $h_{i,j}(x) = \mathrm{SGN}(< w_{i,j}, x >)$
- Now for a new test point x
  - Do a voting among $k(k-1)/2$ classifiers

if there was a tie, use the confidence values.

"GREEDY" VS "END-TO-END"

Show a scenario that the previous two approaches fail.

$d = 2, k = 3$



$[w_1 \quad w_2 \quad w_3] = W \in \mathbb{R}^6$, $w_1 \in \mathbb{R}^2$

pick w_1, w_2, w_3 carefully to get 0-error on train data.

LINEAR MULTI-CLASS PREDICTOR?

$x \in \mathbb{R}^d \,,\; y = 3$

- The multi-vector encoding
  - $y \in \{1,2,\dots,k\}$
  - (x,y) is encoded as $\Psi(x,y) = [0\ \dots\ 0\ x\ 0\ \dots\ 0]^T$ $\in \mathbb{R}^{d \times k}$
  - $h(x) = argmax_y < w, \Psi(x,y) >$

$\Psi(x,y) = [\,\overbrace{0000}^{d}\ \overbrace{0000}^{d}\ \overbrace{x}\ \overbrace{000}\ \dots\,]$

parameter

$\langle w, \dots \rangle$

END-TO-END VERSION OF ONE-VERSUS-ALL

Multiclass SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$

parameters:

regularization parameter $\lambda > 0$

loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$

class-sensitive feature mapping $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^{d \times K}$

e.g. $\ell^{01} = 1\{y \neq \hat{y}\}$

$\mathbb{R}^d$

solve:

$\langle w, \Psi(x^i, y^i)\rangle - \max_{y'} \langle w, \Psi(x^i, y')\rangle \geq 0$

$$\min_{\mathbf{w} \in \mathbb{R}^d} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \max_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}, \Psi(\mathbf{x}_i, y') - \Psi(\mathbf{x}_i, y_i) \rangle \right) \right)$$

output the predictor $h_{\mathbf{w}}(\mathbf{x}) = \mathrm{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}, y) \rangle$

- (More in Chapter 17 of Understanding Machine Learning)

CONFUSION MATRIX

		Actual class		
		Cat	Dog	Rabbit
Predicted class	Cat	5	2	0
	Dog	3	3	2
	Rabbit	0	1	11

wikipedia

- More detailed information about which classes are being misclassified with which

"RISK" MINIMIZATION

Multiclass SVM

input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
parameters:
regularization parameter $\lambda > 0$
loss function $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$
class-sensitive feature mapping $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$
solve:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \left(\lambda \|\mathbf{w}\|^2 + \frac{1}{m} \sum_{i=1}^{m} \max_{y' \in \mathcal{Y}} \left(\Delta(y', y_i) + \langle \mathbf{w}, \Psi(\mathbf{x}_i, y') - \Psi(\mathbf{x}_i, y_i) \rangle \right) \right)$$

output the predictor $h_{\mathbf{w}}(\mathbf{x}) = \operatorname{argmax}_{y \in \mathcal{Y}} \langle \mathbf{w}, \Psi(\mathbf{x}, y) \rangle$

		Actual class		
		Cat	Dog	Rabbit
Predicted class	Cat	5	2	0
	Dog	3	3	2
	Rabbit	0	1	11

$$\Delta = \begin{bmatrix} 0 & 0.1 & 0.5 \\ 0.1 & 0 & 0.8 \\ 0.6 & 0.7 & 0 \end{bmatrix}$$

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

DIFFERENT TYPES OF ERROR

you want to be sure that the phenomenon is present

- False positive (Type I error)
- False negative (Type II error)

		Actual class	
		Cat	Non-cat
Predicted class	Cat	5 True Positives	2 False Positives
	Non-cat	3 False Negatives	17 True Negatives

wikipedia

- Example: A classification method is used to predict whether there is a brain tumor, based on MRI data
  - Case 1: The outcome is used for deciding whether to do a brain surgery
  - Case 2: The outcome is used to decide whether more tests (e.g., CT scan) is required

you Don't want to miss

PRECISION VS RECALL

- TP = $True\ Positive$
- FP = $False\ Positive$
- TN = $True\ Negative$
- FN = $False\ Negative$

- ACCURACY $= \dfrac{TP+TN}{TP+TN+FP+FN}$
- PRECISION$= \dfrac{TP}{TP+FP}$, RECALL$= \dfrac{TP}{TP+FN}$
- ACCURACY IS NOT ALWAYS THE BEST MEASURE
- *BALANCED ERROR*
- $\dfrac{\alpha.TP+(1-\alpha)TN}{TP+TN+FP+FN}$

Drawbacks of SVM?

* Prediction-time complexity
* need to store all training data
* Dealing with $K_{n \times n}$ is difficult.
* Choice of kernel is tricky
  * often heuristic
  * Not very data dependent