

# Linear Multiclass Predictors

He He

CDS, NYU

March 30, 2021

- **Base Hypothesis Space:**  $\mathcal{H} = \{h : \mathcal{X} \rightarrow \mathbb{R}\}$  (score functions).
- **Multiclass Hypothesis Space** (for  $k$  classes):

$$\mathcal{F} = \left\{ x \mapsto \arg \max_i h_i(x) \mid h_1, \dots, h_k \in \mathcal{H} \right\}$$

- $h_i(x)$  scores how likely  $x$  is to be from class  $i$ .
- OvA objective:  $h_i(x) > 0$  for  $x$  with label  $i$  and  $h_i(x) < 0$  for  $x$  with all other labels.
- At test time, for  $(x, i)$  we only need

$$h_i(x) > h_j(x) \quad \forall j \neq i. \tag{1}$$

# Multiclass perceptron

- Base linear predictors:  $h_i(x) = w_i^T x$  ( $w \in \mathbb{R}^d$ ).

- Multiclass perceptron:

Given a multiclass dataset  $\mathcal{D} = \{(x, y)\}$ ;

Initialize  $w \leftarrow 0$ ;

**for**  $iter = 1, 2, \dots, T$  **do**

**for**  $(x, y) \in \mathcal{D}$  **do**

$\hat{y} = \arg \max_{y' \in \mathcal{Y}} w_{y'}^T x$ ;

**if**  $\hat{y} \neq y$  **then** // We've made a mistake

$w_y \leftarrow w_y + x$  ; // Move the target-class scorer towards  $x$

$w_{\hat{y}} \leftarrow w_{\hat{y}} - x$  ; // Move the wrong-class scorer away from  $x$

**end**

**end**

**end**

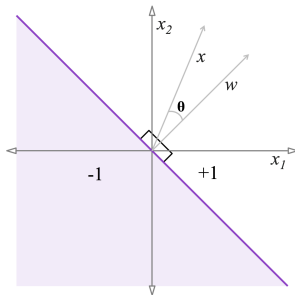
## Side note: Linear Binary Classifier Review

- Input Space:  $\mathcal{X} = \mathbb{R}^d$
- Output Space:  $\mathcal{Y} = \{-1, 1\}$
- Linear classifier score function:

$$f(x) = \langle w, x \rangle = w^T x$$

- Final classification prediction:  $\text{sign}(f(x))$
- Geometrically, when are  $\text{sign}(f(x)) = +1$  and  $\text{sign}(f(x)) = -1$ ?

## Side note: Linear Binary Classifier Review



Suppose  $\|w\| > 0$  and  $\|x\| > 0$ :

$$f(x) = \langle w, x \rangle = \|w\| \|x\| \cos \theta$$

$$f(x) > 0 \iff \cos \theta > 0 \iff \theta \in (-90^\circ, 90^\circ)$$

$$f(x) < 0 \iff \cos \theta < 0 \iff \theta \notin [-90^\circ, 90^\circ]$$

# Rewrite the scoring function

- Remember that we want to scale to very large # of classes and reuse algorithms and analysis for binary classification
  - $\implies$  a **single weight vector** is desired
- How to rewrite the equation such that we have one  $w$  instead of  $k$ ?

$$w_i^T x = w^T \psi(x, i) \tag{2}$$

$$h_i(x) = h(x, i) \tag{3}$$

- Encode labels in the feature space.
- Score for each label  $\rightarrow$  score for the “*compatibility*” of a label and an input.

# The Multivector Construction

How to construct the feature map  $\psi$ ?

- What if we stack  $w_i$ 's together (e.g.,  $x \in \mathbb{R}^2, y = \{1, 2, 3\}$ )

$$w = \left( \underbrace{-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}}_{w_1}, \underbrace{0, 1}_{w_2}, \underbrace{\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}}_{w_3} \right)$$

- And then do the following:  $\Psi: \mathbb{R}^2 \times \{1, 2, 3\} \rightarrow \mathbb{R}^6$  defined by

$$\Psi(x, 1) := (x_1, x_2, 0, 0, 0, 0)$$

$$\Psi(x, 2) := (0, 0, x_1, x_2, 0, 0)$$

$$\Psi(x, 3) := (0, 0, 0, 0, x_1, x_2)$$

- Then  $\langle w, \Psi(x, y) \rangle = \langle w_y, x \rangle$ , which is what we want.

## Rewrite multiclass perceptron

Multiclass perceptron using the multivector construction.

Given a multiclass dataset  $\mathcal{D} = \{(x, y)\}$ ;

Initialize  $w \leftarrow 0$ ;

**for**  $iter = 1, 2, \dots, T$  **do**

**for**  $(x, y) \in \mathcal{D}$  **do**

$\hat{y} = \arg \max_{y' \in \mathcal{Y}} w^T \psi(x, y')$  ; // Equivalent to  $\arg \max_{y' \in \mathcal{Y}} w_{y'}^T x$

**if**  $\hat{y} \neq y$  **then** // We've made a mistake

$w \leftarrow w + \psi(x, y)$  ; // Move the scorer towards  $\psi(x, y)$

$w \leftarrow w - \psi(x, \hat{y})$  ; // Move the scorer away from  $\psi(x, \hat{y})$

**end**

**end**

**end**

**Exercise:** What is the base binary classification problem in multiclass perceptron?



## [discussion] Geometric interpretation

# Feature templates

Toy multiclass example: Part-of-speech classification

- $\mathcal{X} = \{\text{All possible words}\}$
- $\mathcal{Y} = \{\text{NOUN, VERB, ADJECTIVE, ...}\}.$
- Features of  $x \in \mathcal{X}$ : [The word itself], ENDS\_IN\_ly, ENDS\_IN\_ness, ...

How to construct the feature vector?

- Multivector construction:  $w \in \mathbb{R}^{d \times k}$ —**doesn't scale**.
- **Feature templates**: directly design features for each class.

$$\Psi(x, y) = (\psi_1(x, y), \psi_2(x, y), \psi_3(x, y), \dots, \psi_d(x, y)) \quad (4)$$

- Size can be bounded by  $d$ .

# Feature templates

Sample training data:

The boy grabbed the apple and ran away quickly .

Feature templates:

$$\psi_1(x, y) = 1(x = \text{apple AND } y = \text{NOUN})$$

$$\psi_2(x, y) = 1(x = \text{run AND } y = \text{NOUN})$$

$$\psi_3(x, y) = 1(x = \text{run AND } y = \text{VERB})$$

$$\psi_4(x, y) = 1(x \text{ ENDS\_IN\_ly AND } y = \text{ADVERB})$$

...

- E.g.,  $\Psi(x = \text{run}, y = \text{NOUN}) = (0, 1, 0, 0, \dots)$
- After training, what's  $w_1, w_2, w_3, w_4$ ?
- No need to include feature templates unseen in training data.

# Feature templates: implementation

- Flexible, e.g., neighboring words, suffix/prefix.
- “Read off” features from the training data.
- Often sparse—efficient in practice, e.g., NLP problems.
- Can use a hash function:  $\text{template} \rightarrow \{1, 2, \dots, d\}$ .

Ingredients in multiclass classification:

- Scoring functions for each class (similar to ranking).
- Represent labels in the input space  $\implies$  single weight vector.

We've seen

- How to generalize the perceptron algorithm to multiclass setting.
- Very simple idea. Was popular in NLP for structured prediction (e.g., tagging, parsing).

Next,

- How to generalize SVM to the multiclass setting.
- **Concept check:** Why might one prefer SVM / perceptron?

# Margin for Multiclass

- Binary
- Margin for  $(x^{(n)}, y^{(n)})$ :

$$y^{(n)} w^T x^{(n)} \quad (5)$$

- Want margin to be large and positive ( $w^T x^{(n)}$  has same sign as  $y^{(n)}$ )

- Multiclass
- Class-specific margin for  $(x^{(n)}, y^{(n)})$ :

$$h(x^{(n)}, y^{(n)}) - h(x^{(n)}, y). \quad (6)$$

- Difference between scores of the correct class and each other class
- Want margin to be large and positive for all  $y \neq y^{(n)}$ .

# Multiclass SVM: separable case

## Binary

$$\min_w \quad \frac{1}{2} \|w\|^2 \quad (7)$$

$$\text{s.t.} \quad \underbrace{y^{(n)} w^T x^{(n)}}_{\text{margin}} \geq 1 \quad \forall (x^{(n)}, y^{(n)}) \in \mathcal{D} \quad (8)$$

**Multiclass** As in the binary case, take 1 as our target margin.

$$m_{n,y}(w) \stackrel{\text{def}}{=} \underbrace{\langle w, \Psi(x^{(n)}, y^{(n)}) \rangle}_{\text{score of correct class}} - \underbrace{\langle w, \Psi(x^{(n)}, y) \rangle}_{\text{score of other class}} \quad (9)$$

$$\min_w \quad \frac{1}{2} \|w\|^2 \quad (10)$$

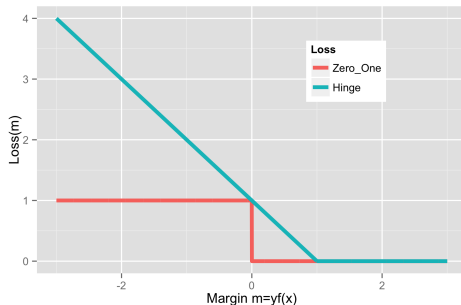
$$\text{s.t.} \quad m_{n,y}(w) \geq 1 \quad \forall (x^{(n)}, y^{(n)}) \in \mathcal{D}, y \neq y^{(n)} \quad (11)$$

**Exercise:** write the objective for the non-separable case

## Recap: hinge loss for binary classification

- Hinge loss: a convex upperbound on the 0-1 loss

$$\ell_{\text{hinge}}(y, \hat{y}) = \max(0, 1 - yh(x)) \quad (12)$$





## [discussion] Generalized hinge loss

- What's the zero-one loss for multiclass classification?

$$\Delta(y, y') = \mathbb{I}\{y \neq y'\} \quad (13)$$

- In general, can also have different cost for each class.
- Upper bound on  $\Delta(y, y')$ .

$$\hat{y} \stackrel{\text{def}}{=} \arg \max_{y' \in \mathcal{Y}} \langle w, \Psi(x, y') \rangle \quad (14)$$

$$\implies \langle w, \Psi(x, y) \rangle \leq \langle w, \Psi(x, \hat{y}) \rangle \quad (15)$$

$$\implies \Delta(y, \hat{y}) \leq \Delta(y, \hat{y}) + \langle w, (\Psi(x, \hat{y}) - \Psi(x, y)) \rangle \quad \text{When are they equal?} \quad (16)$$

- Generalized hinge loss:

$$\ell_{\text{hinge}}(y, x, w) \stackrel{\text{def}}{=} \max_{y' \in \mathcal{Y}} (\Delta(y, y') + \langle w, (\Psi(x, y') - \Psi(x, y)) \rangle) \quad (17)$$

# Multiclass SVM with Hinge Loss

- Recall the hinge loss formulation for binary SVM (without the bias term):

$$\min_{w \in \mathbb{R}^d} \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \max \left( 0, 1 - \underbrace{y^{(n)} w^T x^{(n)}}_{\text{margin}} \right).$$

- The multiclass objective:

$$\min_{w \in \mathbb{R}^d} \frac{1}{2} \|w\|^2 + C \sum_{n=1}^N \max_{y' \in \mathcal{Y}} \left( \Delta(y, y') - \underbrace{\langle w, (\Psi(x, y) - \Psi(x, y')) \rangle}_{\text{margin}} \right)$$

- $\Delta(y, y')$  as **target margin** for each class.
- If margin  $m_{n,y'}(w)$  meets or exceeds its target  $\Delta(y^{(n)}, y') \forall y \in \mathcal{Y}$ , then no loss on example  $n$ .

## Recap: What Have We Got?

- Problem: Multiclass classification  $\mathcal{Y} = \{1, \dots, k\}$
- Solution 1: One-vs-All
  - Train  $k$  models:  $h_1(x), \dots, h_k(x) : \mathcal{X} \rightarrow \mathbb{R}$ .
  - Predict with  $\arg \max_{y \in \mathcal{Y}} h_y(x)$ .
  - Gave simple example where this fails for linear classifiers
- Solution 2: Multiclass loss
  - Train one model:  $h(x, y) : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ .
  - Prediction involves solving  $\arg \max_{y \in \mathcal{Y}} h(x, y)$ .

# Does it work better in practice?

- Paper by Rifkin & Klautau: “In Defense of One-Vs-All Classification” (2004)
  - Extensive experiments, carefully done
    - albeit on relatively small UCI datasets
  - Suggests one-vs-all works just as well in practice
    - (or at least, the advantages claimed by earlier papers for multiclass methods were not compelling)
- Compared
  - many multiclass frameworks (including the one we discuss)
  - one-vs-all for SVMs with RBF kernel
  - one-vs-all for square loss with RBF kernel (for classification!)
- All performed roughly the same

# Why Are We Bothering with Multiclass?

- The framework we have developed for multiclass
  - compatibility features / scoring functions
  - multiclass margin
  - target margin / multiclass loss
- Generalizes to situations where  $k$  is very large and one-vs-all is intractable.
- Key idea is that we can generalize across outputs  $y$  by using features of  $y$ .