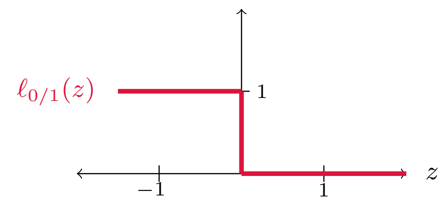
Objectives

Prediction error / zero-one loss

P is a distribution over $\mathcal{X} \times \{-1, +1\}$, and $(X, Y) \sim P$.

For any classifier $f : \mathcal{X} \to \{-1, +1\}$,

$$\operatorname{err}(f) \;=\; P\Big(f(X) \neq Y\Big) \;=\; \mathbb{E}\Big[\ell_{0/1}(Yf(X))\Big].$$



Also works with **real-valued predictors** $f : \mathcal{X} \to \mathbb{R}$; for example:

- k-**NN**: average of y-values of k nearest neighbors.
- **Trees**: leaf nodes with a real-valued output (e.g., average of y-values of training examples that reach a leaf). "Regression trees"
- **Linear classifiers**: $\boldsymbol{x} \mapsto \langle \boldsymbol{w}, \boldsymbol{x} \rangle - t$.
- **Classifiers from generative models**: $x \mapsto P_{\hat{\theta}}(Y = +1 \mid X = x) - 1/2$.

Often useful to adjust threshold (e.g., t and $1/2$ above).

Thresholds

Uses for adjusting threshold t

Often have **different costs for different kinds of mistakes**:

	$f(X) \le t$	$f(X) > t$
$Y = -1$	0	c
$Y = +1$	$1 - c$	0

Also, often interested in **different performance criteria**.

- **Precision**:
$$P(Y = +1 \mid f(X) > t)$$
- **Recall** (a.k.a. **Sensitivity**, **True Positive Rate**):
$$P(f(X) > t \mid Y = +1)$$
- **Specificity**:
$$P(f(X) \le t \mid Y = -1)$$
- **False Positive Rate**:
$$P(f(X) > t \mid Y = -1)$$

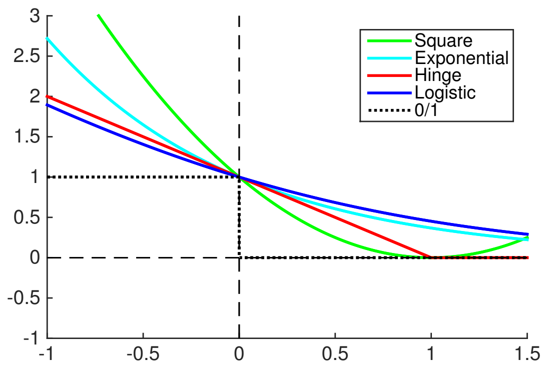
Conditional probability estimation

Sometimes would like real-valued predictor f to be related to the **conditional probability function** η

$$\eta(x) \;=\; P(Y = +1 \mid X = x).$$

- Straightforward when using generative models.
- Can use a loss function that is minimized by η (or some invertible transformation thereof).

Eliciting conditional probabilities



Goal: loss function that is minimized by (some invertible transformation of) the conditional probability function

$$\eta(x) \;=\; P(Y = +1 \mid X = x)\,.$$

Examples

Loss functions and their minimizers

- **Square loss**: $\ell_{\text{sq}}(z) = (1-z)^2$

  $\mathbb{E}[\ell_{\text{sq}}(Yf(x)) \mid X = x]$ is minimized by f s.t. $f(x) = 2\eta(x) - 1$.

  So $\eta(x) = (f(x)+1)/2$.

- **Logistic loss**: $\ell_{\text{logistic}}(z) = \ln(1+\exp(-z))$

  $\mathbb{E}[\ell_{\text{logistic}}(Yf(x)) | X = x]$ is minimized by f s.t. $f(x) = \ln\left(\frac{\eta(x)}{1-\eta(x)}\right)$.

  So $\eta(x) = (1+\exp(-f(x)))^{-1}$.

Non-example

- **Hinge loss**: $\ell_{\text{hinge}}(z) = \max\{0,\, 1-z\}$

  $\mathbb{E}[\ell_{\text{hinge}}(Yf(x)) | X = x]$ is minimized by f s.t. $f(x) = \text{sign}(2\eta(x)-1)$.

  Cannot recover $\eta(x)$ from $f(x)$.

Eliciting conditional probabilities

Using loss functions: easy with linear/affine functions whenever the loss function ℓ is a convex function:

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \quad R(\boldsymbol{w}) + \frac{1}{n}\sum_{i=1}^{n} \ell(y_i \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle)\,.$$

(Here, the regularization function R is also assumed to be convex.)

Caveat: Might not be possible to represent

$$\boldsymbol{x} \mapsto 2\eta(\boldsymbol{x}) - 1 \quad \text{or} \quad \boldsymbol{x} \mapsto \ln\left(\frac{\eta(\boldsymbol{x})}{1-\eta(\boldsymbol{x})}\right)$$

as (say) a linear function $\boldsymbol{x} \mapsto \langle \boldsymbol{w}, \boldsymbol{x} \rangle$.

Common remedies: enhance the feature space via feature expansion or kernels, or use more flexible models (e.g., tree models).

Structured output spaces

Sometimes $\mathcal{Y}$ is not just $\{0,1\}$ or $\{1,2,\ldots,K\}$, but rather a collection of *structured objects*.

Example: sequence tagging

- $\mathcal{X}$: sequences of English words
- $\mathcal{Y}$: sequences of parts-of-speech

the/D man/N saw/V the/D dog/N

(Verbs tend to follow Nouns.)

Many other examples:

- sentence parse trees
- web search result ranking
- visual scene labeling
- ...

Structured output prediction

Featurization

Create several input-output feature maps $\phi_1, \phi_2, \ldots, \phi_d \colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.

- e.g., $\phi_{1000}(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{1}\{i\text{-th word in } \boldsymbol{x} \text{ is "the", and } i\text{-th POS in } \boldsymbol{y} \text{ is "D"}\}$

For each possible $\boldsymbol{y} \in \mathcal{Y}$, consider an *input-output feature vector*:

$$\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y}) \; := \; \big(\phi_1(\boldsymbol{x}, \boldsymbol{y}), \phi_2(\boldsymbol{x}, \boldsymbol{y}), \ldots, \phi_d(\boldsymbol{x}, \boldsymbol{y})\big) \; \in \; \mathbb{R}^d .$$

Note: often d is enormous, but $\phi_i(\boldsymbol{x}, \boldsymbol{y}) = 0$ for most i.

Model

Prediction model is based on *linear functions of input-output feature vectors*:

$$\boldsymbol{x} \; \mapsto \; \arg\max_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

for weight vector $\boldsymbol{w} \in \mathbb{R}^d$.

Note: the $\arg\max$ can often be computed efficiently (e.g., via dynamic programming), even when $\mathcal{Y}$ is enormous.

Structured Perceptron training (Collins, 2002)

Online Structured Perceptron

input Labeled examples $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$ from $\mathcal{X} \times \mathcal{Y}$.

1: **initialize** $\hat{\boldsymbol{w}}_1 := \boldsymbol{0}$.
2: **for** $t = 1, 2, \ldots,$ **do**
3: Predict: $\hat{\boldsymbol{y}}_t := \arg\max_{\boldsymbol{y} \in \mathcal{Y}} \langle \hat{\boldsymbol{w}}_{t-1}, \boldsymbol{\Phi}(\boldsymbol{x}_t, \boldsymbol{y}) \rangle$
4: **if** $\hat{\boldsymbol{y}}_t \neq \boldsymbol{y}_t$ **then**
5: Update:

$$\hat{\boldsymbol{w}}_t \; := \; \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{\Phi}(\boldsymbol{x}_t, \boldsymbol{y}_t) - \boldsymbol{\Phi}(\boldsymbol{x}_t, \hat{\boldsymbol{y}}_t) .$$

6: **else**
7: No update: $\hat{\boldsymbol{w}}_t := \hat{\boldsymbol{w}}_{t-1}$
8: **end if**
9: **end for**

Can also help to make multiple passes through data, and also to employ averaging (as in Averaged Perceptron).

Key takeaways

1. Concept of real-valued predictors and thresholds; alternative performance criteria.
2. Eliciting conditional probabilities with loss functions.
3. High-level idea of structured output prediction.