Objectives

Prediction error / zero-one loss

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

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

\[ \text{err}(f) = P\left( f(X) \neq Y \right) = \mathbb{E}\left[ \ell_{0/1}(Yf(X)) \right]. \]

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

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

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:

<table>
<thead>
<tr>
<th>( Y )</th>
<th>( f(X) \leq t )</th>
<th>( f(X) &gt; t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-1)</td>
<td>( 0 )</td>
<td>( c )</td>
</tr>
<tr>
<td>(+1)</td>
<td>( 1 - c )</td>
<td>( 0 )</td>
</tr>
</tbody>
</table>

Also, often interested in different performance criteria.

- \( \text{Precision}: \)
  \[ P(Y = +1 | f(X) > t) \]
- \( \text{Recall} \) (a.k.a. Sensitivity, True Positive Rate):
  \[ P(f(X) > t | Y = +1) \]
- \( \text{Specificity}: \)
  \[ P(f(X) \leq t | Y = -1) \]
- \( \text{False Positive Rate}: \)
  \[ P(f(X) > t | Y = -1) \]

Conditional probability estimation

Sometimes would like real-valued predictor \( f \) to be related to the conditional probability function \( \eta \)

\[ \eta(x) = P(Y = +1 | X = x). \]

- Straightforward when using generative models.
- Can use a loss function that is minimized by \( \eta \) (or some invertible transformation thereof).
Eliciting conditional probabilities

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

\[ \eta(x) = P(Y = +1 \mid X = x). \]

**Using loss functions:** easy with linear/affine functions whenever the loss function \( \ell \) is a convex function:

\[
\min_{w \in \mathbb{R}^d} R(w) + \frac{1}{n} \sum_{i=1}^{n} \ell(y_i \langle w, x_i \rangle).
\]

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

**Caveat:** Might not be possible to represent

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

as (say) a linear function \( x \mapsto \langle w, x \rangle \).

**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**

\[
\mathcal{X}: \text{sequences of English words} \quad \mathcal{Y}: \text{sequences of parts-of-speech}
\]

\[
\text{the/D man/N saw/V the/D dog/N}
\]

(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 $\phi_1, \phi_2, \ldots, \phi_d : X \times Y \rightarrow \mathbb{R}$.

- e.g., $\phi_{1000}(x, y) = 1$ if $i$-th word in $x$ is “the”, and $i$-th POS in $y$ is “D”

For each possible $y \in Y$, consider an input-output feature vector:

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

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

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

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

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

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

---

**Structured Perceptron training (Collins, 2002)**

Online Structured Perceptron

input Labeled examples $\{(x_i, y_i)\}_{i=1}^n$ from $X \times Y$.

1. initialize $\hat{w}_1 := 0$.
2. for $t = 1, 2, \ldots, \text{do}$
3. Predict: $\hat{y}_t := \arg \max_{y \in Y} \langle \hat{w}_{t-1}, \Phi(x_t, y) \rangle$
4. if $\hat{y}_t \neq y_t$ then
5. Update: $\hat{w}_t := \hat{w}_{t-1} + \Phi(x_t, y_t) - \Phi(x_t, \hat{y}_t)$.
6. else
7. No update: $\hat{w}_t := \hat{w}_{t-1}$
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.