8. Supervised learning, MaxMin estimate

Given training data $\mathcal{I}_\ell = \{(x^j, s^j) \mid j = 1, .., \ell\}$ solve

$$\vec{u}_* = \operatorname*{argmax}_{\vec{u}} \min_j \log p_{\vec{u}}(x^j, s^j)$$

We write the task as a convex optimisation task

$$\log Z(\vec{u}) - c \to \min_{\vec{u}, c}$$

$$\text{s.t.} \quad c - \langle \vec{\varphi}^j, \vec{u} \rangle \leq 0 \quad \forall j = 1, .., \ell \tag{1}$$

where $\vec{\varphi}^j = \vec{\varphi}(x^j, s^j)$

Construct the Lagrange function and the dual task

$$L(c, \vec{u}, \vec{\lambda}) = \log Z(\vec{u}) - c + \sum_{j=1}^{\ell} \lambda_j \left[c - \langle \vec{\varphi}^j, \vec{u} \rangle \right]$$

$$\max_{\vec{\lambda} \geq 0} \min_{\vec{u}, c} L(c, \vec{u}, \vec{\lambda}) =$$

$$\max_{\vec{\lambda} \in \Delta^{\ell-1}} \underbrace{\min_{\vec{u}} \left[\log Z(\vec{u}) - \sum_j \lambda_j \langle \vec{\varphi}^j, \vec{u} \rangle \right]}_{\text{MLE}}$$

where $\Delta^{\ell-1}$ denotes the simplex $\Delta^{\ell-1} = \{\vec{\lambda} \in \mathbb{R}^\ell_+ \mid \sum_j \lambda_j = 1\}$

Algorithm Choose arbitrary $\vec{\lambda}^{(0)} \in \Delta^{n-1}$ e.g. $\lambda_j^{(0)} = \frac{1}{\ell}$

Iterate:

1) Solve the MLE task (see sec. 7) for $\vec{\lambda}^{(t)} \to \vec{u}^{(t)}$

2) find $i \in \operatorname*{argmin}_j p_{\vec{u}^{(t)}}(x^j, s^j) = \operatorname*{argmin}_j \langle \vec{\varphi}(x^j, s^j), \vec{u}^{(t)} \rangle$

set $\vec{\lambda}^{(t+1)} \sim (t+1)\vec{\lambda}^{(t)} + \vec{e}_i$

until

$$\sum_{j=1}^{\ell} \lambda_j \langle \vec{\varphi}^j, \vec{u} \rangle - \min_j \langle \vec{\varphi}^j, \vec{u} \rangle \leq \varepsilon$$

Theorem 1 (w/o proof)

The algorithm stops after a finite number of iterations

The obtained model $\vec{u}_*$ is ε-optimal, i.e.

$$\min_j \log p_{\vec{u}_0}(x^j, s^j) - \min_j \log p_{\vec{u}_*}(x^j, s^j) \leq \varepsilon,$$

where $\vec{u}_0$ is the optimal model.

9. Empirical risk minimisation

Given: • training data $\mathcal{T}_\ell = \{(x^j, s^j) \mid j = 1, .., \ell\}$

• loss function $C(s', s) = \mathbb{1}\{s' \neq s\}$, i.e.

Bayes optimal decision $q_{\vec{u}}(x) \in \operatorname{argmax}_{s \in K^n} p_{\vec{u}}(x, s)$

The task of empirical risk minimisation reads

$$\frac{1}{\ell} \sum_{j=1}^{\ell} \mathbb{1}\{s^j \neq q_{\vec{u}}(x^j)\} \to \min_{\vec{u}}$$

It is not tractable in general; the objective function is neither convex nor differentiable.

It is simple to solve if $\exists\, \vec{u}_*$ s.t.

$$s^j = \operatorname{argmax}_{s \in K^n} p_{\vec{u}_*}(x^j, s) \quad \forall\, j = 1, .., \ell$$

i.e. we must find a $\vec{u}$ s.t. inequalities

$$\langle \vec{\varphi}(x^j, s^j), \vec{u} \rangle > \langle \vec{\varphi}(x^j, s), \vec{u} \rangle \quad \forall\, s \neq s^j$$

hold for all $j = 1, .., \ell$

This task can be solved by the perceptron algorithm:

Start from arbitrary $\vec{u}$ and iterate:

- solve

$$\tilde{s}^j = \operatorname*{argmax}_{s \in K^n} \langle \vec{\varphi}(x^j, s), \vec{u} \rangle \qquad \text{(see sec. 4)}$$

$\forall\, j = 1, \dots, \ell$

- if for some j $\tilde{s}^j \neq s^j$, update $\vec{u}$ by

$$\vec{u} \to \vec{u} + \vec{\varphi}(x^j, s^j) - \vec{\varphi}(x^j, \tilde{s}^j)$$

Let us reconsider the general task

$$q_{\vec{u}}(x) \in \operatorname*{argmax}_{s \in K^n} \langle \vec{\varphi}(x, s), \vec{u} \rangle$$

$$\frac{1}{\ell} \sum_{j=1}^{\ell} \mathbb{1}\{ s^j \neq q_{\vec{u}}(x^j) \} \to \min_{\vec{u}}$$

Approximate the loss (as a function of $\vec{u}$) by a convex upper bound, e.g., as follows

$$\mathbb{1}\{ s' \neq q_{\vec{u}}(x) \} \leq \max_{s} \left\{ \mathbb{1}\{ s' \neq s \} - \langle \vec{\varphi}(x, s'), \vec{u} \rangle + \langle \vec{\varphi}(x, s), \vec{u} \rangle \right\}$$

The approx. task reads

$$\frac{1}{\ell} \sum_{j=1}^{\ell} \max_{s \in K^n} \left[\mathbb{1}\{ s' \neq s \} + \langle \vec{\varphi}(x^j, s) - \vec{\varphi}(x^j, s^j), \vec{u} \rangle \right] \to \min_{\vec{u}}$$

Solve it

- by subgradient descent or
- by cutting plane algorithm or
- ...

10. Unsupervised learning, EM algorithm

Given: i.i.d. training data $T_\ell = \{x^j \in F^n \mid j = 1, \dots, \ell\}$

Task: $\vec{u}_* \in \operatorname{argmax}_{\vec{u}} \sum_{x \in F^n} \beta(x) \log \sum_{s \in K^n} p_{\vec{u}}(x,s)$

Substituting $p_{\vec{u}}$ we get

$$\underbrace{\log Z(\vec{u})}_{g(\vec{u})} - \underbrace{\sum_{x \in F^n} \beta(x) \log \sum_{s \in K^n} \exp \langle \vec{\varphi}(x,s), \vec{u} \rangle}_{h(\vec{u})} \to \min_{\vec{u}} \quad (1)$$

The objective function is a difference of convex functions, i.e. we have to solve a DC-program

A. DC-programs, DC-duality, DC-algorithm

Definition 1 The Fenchel conjugate of a function $g: \mathbb{R}^n \to [-\infty, +\infty]$ is the function $g^*: \mathbb{R}^n \to [-\infty, +\infty]$ defined by

$$g^*(\vec{v}) = \sup_{\vec{u} \in \mathbb{R}^n} \{ \langle \vec{v}, \vec{u} \rangle - g(\vec{u}) \}$$

- The function g^* is convex
- If g is convex and closed then

$$\vec{v} \in \partial g(\vec{u}) \iff \vec{u} \in \partial g^*(\vec{v}) \quad (2)$$

- Each DC program has a DC-dual program

$$g(\vec{u}) - h(\vec{u}) \to \min_{\vec{u}}$$

$$h^*(\vec{v}) - g^*(\vec{v}) \to \min_{\vec{v}}$$

The optimal values of both tasks coincide

- The <u>DC-algorithm</u> aims at solving both tasks simultaneously by constructing a <u>pair</u> of sequences $\vec{u}^{(t)}, \vec{v}^{(t)}$ in an alternating way

a) $\vec{v}^{(t)} \in \partial h(\vec{u}^{(t)})$

b) $\vec{u}^{(t+1)} \in \partial g^*(\vec{v}^{(t)})$

B. <u>DC-algorithm $\hat{=}$ EM algorithm for (1)</u>

Applying DCA for (1) is a „reincarnation" of the EM-algorithm:

Choose an arbitrary $\vec{u}^{(0)}$. Iterate

a) E-step: compute

$$\vec{v}^{(t)} = \nabla h(u^{(t)})$$

$$= \sum_x \beta(x) \sum_s \exp\langle \vec{\varphi}(x,s), \vec{u}^{(t)} \rangle \vec{\varphi}(x,s) \Big/ \sum_s \exp\langle \varphi(x,s), \vec{u}^{(t)} \rangle$$

$$= \sum_{x,s} \beta(x)\, p_{\vec{u}^{(t)}}(s|x)\, \vec{\varphi}(x,s)$$

i.e. $\vec{v}^{(t)}$ is the vector of marginal statistics of the distr. $\beta(x)\, p_{\vec{u}^{(t)}}(s|x)$

For this it is necessary to compute pairwise posterior marginals $p_{\vec{u}^{(t)}}(s_{i-1}, s_i | x) \quad \forall i = 2, \dots, n$ for each $x \in \mathcal{L}_c$ (see sec. 5).

b) M-step: compute $\vec{u}^{(t+1)} \in \partial g^*(\vec{v}^{(t)})$

From Def. 1 and (2) we have

$$\vec{u}_* \in \underbrace{\operatorname{argmax}_{\vec{u}} \left[\langle \vec{v}^{(t)}, \vec{u} \rangle - \log Z(\vec{u}) \right]}_{\text{MLE}} \Rightarrow \vec{u}_* \in \partial g^*(\vec{v}^{(t)})$$

But this task is easy to solve (see sec. 7)!

∎

Theorem 1 (w/o proof)

- The sequences $g(\vec{u}^{(t)}) - h(\vec{u}^{(t)})$ and $h^*(\vec{v}^{(t)}) - g^*(\vec{v}^{(t)})$ are non-increasing
- The sequence $\vec{v}^{(t)}$ is convergent. Its fixpoint is a local minimum of $h^*(\vec{v}) - g^*(\vec{v})$. ∎