

Lq-regularised sparse classifiers: A PAC-Bayes analysis

Ata Kabán (A.Kaban@cs.bham.ac.uk)

- Training set $\{(\mathbf{x}_j, y_j)\}_{j=1}^n$, where $\mathbf{x}_j \in \mathbb{R}^m$ inputs and $y_j \in \{-1, 1\}$ their labels.
- Lq-regularised logistic regression:

$$\max_{\mathbf{w}} \sum_{i=1}^n -\log(1 + \exp(-y\mathbf{w}^T \mathbf{x}_i)) - \lambda \|\mathbf{w}\|_q^q \quad (1)$$

where $\|\mathbf{w}\|_q = (\sum_{i=1}^m |w_i|^q)^{1/q}$.

- $q \leq 1$: non-differentiable at zero \Rightarrow sparsity.
- L1-regularisation ($q = 1$) is convenient but $q < 1$ seems to have added value in certain cases.
- Previously derived a data-independent PAC-bound – polynomial, too loose
- Derive a data-dependent PAC-Bayes bound, by interpreting the regularisation term as the log prior of the following form:

$$GGD(w_j | \mu_j, \lambda, q) = \frac{q\lambda^{1/q}}{2\Gamma(1/q)} \exp\{-\lambda|w_j - \mu_j|^q\} \quad (2)$$

where $\lambda > 0$ and we are interested in $q \in (0, 1]$.

PAC-Bayes Thm for Lq-reg. logistic regression.

$\forall D, \forall P(h), \forall \delta \in (0, 1]$,

$$\Pr_{S \sim D^n} \left\{ \forall Q(h) : KL_+[\hat{Q}_S || Q_D] \leq \frac{1}{n} [KL(Q || P) + \ln \frac{n+1}{\delta}] \right\} \geq 1 - \delta \quad (3)$$

- $Q := GGD(w_j | \hat{w}_j, \tilde{\lambda}, \tilde{q})$ and $\tilde{\lambda}, \tilde{q}$ can be chosen to tighten the bound
- $\hat{Q}_S \equiv \mathbb{E}_{h \sim Q} [\frac{1}{n} \sum_{i=1}^n I(h(\mathbf{x}_i) \neq y_i)]$,

$$\hat{Q}_S \approx \frac{1}{n} \sum_{i=1}^n \Phi \left\{ - \frac{y_i \hat{\mathbf{w}}^T \mathbf{x}_i}{\tilde{\lambda}^{-1/\tilde{q}} \sqrt{\frac{\Gamma(3/\tilde{q})}{\Gamma(1/\tilde{q})}} \|\mathbf{x}_i\|_2} \right\} \quad (4)$$

- $KL(Q || P) = \mathbb{E}_{h \sim Q} \ln \frac{Q(h)}{P(h)} = \sum_{j=1}^m KL[Q(w_j) || P(w_j)]$ is intractable, but $KL[Q(w_j) || P(w_j)] \leq$
 $\leq KL[GGD(w_j | 0, \tilde{\lambda}, \tilde{q}) || GGD(w_j | 0, \lambda, q)] + \lambda |\hat{w}_j|^q \quad (5)$
 where '=' holds whenever $\hat{w}_j = 0$.
- $Q_D \equiv Pr_{(\mathbf{x}, y) \sim D} \{h(\mathbf{x}) \neq y\}$ is the true (generalisation) error, which is solved numerically from (3).

