Advanced Topics in Machine Learning

2. Hyperparameter Learning

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
University of Hildesheim, Germany

Outline

1. Grid Search

Outline

Alternating Parameter / Hyperparameter Learning

Let f be an objective function depending on both, parameters Θ and hyperparameters H, e.g.,

$$f(\Theta, H) := R(\mathcal{D}_{\text{train}}; \Theta, H) + r(\Theta, H)$$

with a risk R and a regularization r, where usually

$$R(\mathcal{D}; \Theta, H) := \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(y, \hat{y}(x; \Theta, H))$$

with a loss ℓ and a prediction function $\hat{y}$.

Alternating Parameter / Hyperparameter Learning

For such an objective function f one could learn parameters and hyperparameters in an alternating manner ("EM style"):

initialize Θ, H
while *not yet converged* **do**
  $\Theta := \arg\min_\Theta f_H(\Theta) := f(\Theta, H)$
  $H := \arg\min_H f_\Theta(H) := f(\Theta, H)$
end
return (Θ, H)

E.g., for a linear SVM (with $\Theta := (\beta, \beta_0), H := (C)$) minimize:

$$f(\beta, \beta_0, C) := \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} [1 - y(\beta_0 + \beta^T x)]_+ + C\beta^T\beta$$

Alternating Parameter / Hyperparameter Learning

For such an objective function f one could learn parameters and hyperparameters in an alternating manner ("EM style"):

initialize Θ, H
while *not yet converged* **do**
  $\Theta := \arg\min_\Theta f_H(\Theta) := f(\Theta, H)$
  $H := \arg\min_H f_\Theta(H) := f(\Theta, H)$
end
return (Θ, H)

E.g., for a linear SVM (with $\Theta := (\beta, \beta_0), H := (C)$) minimize:

$$f(\beta, \beta_0, C) := \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} [1 - y(\beta_0 + \beta^T x)]_+ + C\beta^T\beta$$

This will not work as f is linear in C, i.e., minimal for $C = 0$.

The Hyperparameter Learning Problem

Let $f_{\text{calib}}(\Theta, H)$ be a calibration function, usually just

$$f_{\text{calib}}(\Theta, H) := R(\mathcal{D}_{\text{calib}}, \Theta, H)$$

the risk on a calibration sample.

Hyperparameter Learning Problem: Find Θ, H s.t.

$$\text{(i)}\ H := \arg\min_H f_{\text{calib},\Theta\ \text{opt}}(H) := f_{\text{calib}}(\arg\min_\Theta f(\Theta, H), H)$$

$$\text{(ii)}\ \Theta := \arg\min_\Theta f_H(\Theta) := f(\Theta, H)$$

Example: Grid Search

If H consists of K different hyperparameters η_k and for each hyperparameter η_k there are some candidate values

$$H_k := \{\eta_{k,1}, \eta_{k,2}, \ldots, \eta_{k,K_k}\}$$

plausibly spaced in some plausible range, then

$$H := \arg\min_{H \in \prod_{k=1}^{K} H_k} f_{\text{calib},\Theta\ \text{opt}}(H) := f_{\text{calib}}(\arg\min_{\Theta} f(\Theta, H), H)$$

are called optimal hyperparameters for grid $\prod_k H_k$ (grid search).

- grid search is trivially parallelizable.
- if an optimal hyperparameter is at the border of the grid, its range should be expanded.
- often a second, narrower grid is searched centered around the optimal hyperparameters of the first, coarse grid.

Example: Early Stopping

early-stopping(iterate, f, f_{calib}, $t_{\text{lookahead}}$):
$t := 0$, $t^* := 0$
initialize $\Theta^{(0)}$
while $t - t^* < t_{lookahead}$ **do**
    $\Theta^{(t+1)} :=$ iterate$(f, \Theta^{(t)})$ `// with` $f(\Theta^{(t+1)}) < f(\Theta^{(t)})$
    if $f_{calib}(\Theta^{(t+1)}) < f_{calib}(\Theta^{(t^*)})$ **then**
        $t^* := t + 1$
    end
    $t := t + 1$
end
return $\Theta^{(t^*)}$

Can early stopping also be understood as hyperparameter learning?

Example: Early Stopping

early-stopping(iterate, f, f_{calib}, $t_{\text{lookahead}}$):
$t := 0,\ t^* := 0$
initialize $\Theta^{(0)}$
while $t - t^* < t_{lookahead}$ **do**
  $\Theta^{(t+1)} :=$ iterate$(f, \Theta^{(t)})$ `// with` $f(\Theta^{(t+1)}) < f(\Theta^{(t)})$
  if $f_{calib}(\Theta^{(t+1)}) < f_{calib}(\Theta^{(t^*)})$ **then**
    $t^* := t + 1$
  end
  $t := t + 1$
end
return $\Theta^{(t^*)}$

Can early stopping also be understood as hyperparameter learning?

- Yes, we learn the hyperparameter t^* number of iterations.
- sequential search with lookahead.

Example: Ridge Regression (w/o intercept)

Minimize:

$$f_{\text{calib}}(\lambda) := \frac{1}{|\mathcal{D}_{\text{calib}}|} \sum_{(x,y)\in\mathcal{D}_{\text{calib}}} (y - \beta(\lambda)^T x)^2$$

with

$$\beta(\lambda) := \arg\min_{\lambda} f_\lambda(\beta)$$

$$f_\lambda(\beta) := \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} (y - \beta^T x)^2 + \lambda \beta^T \beta$$

Idea: can we replace the search over hyperparameter λ by proper minimization using gradients (e.g., a gradient descent)?

[Chapelle et al. 2002; Keerthi, Sindhwani, and Chapelle 2007]

Example: Ridge Regression (w/o intercept)

Minimize:

$$f_{\text{calib}}(\lambda) := \frac{1}{|\mathcal{D}_{\text{calib}}|} \sum_{(x,y)\in\mathcal{D}_{\text{calib}}} (y - \beta(\lambda)^T x)^2$$

with

$$\beta(\lambda) := \arg\min_{\lambda} f_{\lambda}(\beta)$$

$$f_{\lambda}(\beta) := \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} (y - \beta^T x)^2 + \lambda\beta^T\beta$$

Idea: can we replace the search over hyperparameter λ by proper minimization using gradients (e.g., a gradient descent)?

[Chapelle et al. 2002; Keerthi, Sindhwani, and Chapelle 2007]

$$\frac{\partial f_{\text{calib}}}{\partial\lambda}(\lambda) = \frac{1}{|\mathcal{D}_{\text{calib}}|} \sum_{(x,y)\in\mathcal{D}_{\text{calib}}} -2(y - \beta(\lambda)^T x)\frac{\partial\beta}{\partial\lambda}(\lambda)^T x$$

Example: Ridge Regression (w/o intercept)

In matrix notation (and with rescaled λ): minimize:

$$f_{\text{calib}}(\lambda) := (y_{\text{calib}} - X_{\text{calib}}\beta(\lambda))^T(y_{\text{calib}} - X_{\text{calib}}\beta(\lambda))$$

with

$$\beta(\lambda) := \arg\min_{\lambda} f_\lambda(\beta) := (y_{\text{train}} - X_{\text{train}}\beta)^T(y_{\text{train}} - X_{\text{train}}\beta) + \lambda\beta^T\beta$$

$$\text{i.e. } (X_{\text{train}}^T X_{\text{train}} + \lambda I)\beta(\lambda) = X_{\text{train}}^T y_{\text{train}}$$

Thus

$$(X_{\text{train}}^T X_{\text{train}} + \lambda I)\frac{\partial\beta}{\partial\lambda}(\lambda) + I\beta(\lambda) \stackrel{!}{=} 0$$

$$\rightsquigarrow \frac{\partial\beta}{\partial\lambda}(\lambda) = -(X_{\text{train}}^T X_{\text{train}} + \lambda I)^{-1}\beta(\lambda)$$

$$\frac{\partial f_{\text{calib}}}{\partial\lambda}(\lambda) = -2(y_{\text{calib}} - X_{\text{calib}}\beta)^T X_{\text{calib}}\frac{\partial\beta}{\partial\lambda}(\lambda)$$

Example: Ridge Regression (w/o intercept)

$$
\begin{aligned}
\frac{\partial f_{\text{calib}}}{\partial \lambda}(\lambda) &= -2(y_{\text{calib}} - X_{\text{calib}}\beta(\lambda))^T X_{\text{calib}} \frac{\partial \beta}{\partial \lambda}(\lambda) \\
&= -2(X_{\text{calib}}^T(y_{\text{calib}} - X_{\text{calib}}\beta(\lambda)))^T \frac{\partial \beta}{\partial \lambda}(\lambda) \\
&= 2(X_{\text{calib}}^T(y_{\text{calib}} - X_{\text{calib}}\beta(\lambda)))^T (X_{\text{train}}^T X_{\text{train}} + \lambda I)^{-1}\beta(\lambda) \\
&= 2((X_{\text{train}}^T X_{\text{train}} + \lambda I)^{-1}(X_{\text{calib}}^T(y_{\text{calib}} - X_{\text{calib}}\beta(\lambda))))^T \beta(\lambda) \\
&= 2d^T\beta(\lambda)
\end{aligned}
$$

with

$$
(X_{\text{train}}^T X_{\text{train}} + \lambda I)d = X_{\text{calib}}^T(y_{\text{calib}} - X_{\text{calib}}\beta(\lambda))
$$

Example: Ridge Regression (w/o intercept)

ridge-regression-auto-hyper$(X_{\text{train}}, y_{\text{train}}, X_{\text{calib}}, y_{\text{calib}}, \lambda_0, \eta, \epsilon)$:

$\lambda := \lambda_0,\ g := \epsilon$

while $|g| \geq \epsilon$ **do**

    compute β: $(X_{\text{train}}^T X_{\text{train}} + \lambda I)\beta = X_{\text{train}}^T y_{\text{train}}$

    compute d: $(X_{\text{train}}^T X_{\text{train}} + \lambda I)d = X_{\text{calib}}^T (y_{\text{calib}} - X_{\text{calib}}\beta)$

    $g := 2d^T\beta$

    $\lambda := [\lambda - \eta g]_+$

end

return (β, λ)

Example: SVM

Lemma

Let α, β_0 be the solution of a SVM with hyperparameters H (i.e., C and evtl. kernel parameters). Then for $\tilde{H}$ close to H, the solution of the SVM with hyperparameters $\tilde{H}$ is $\tilde{\alpha}, \tilde{\beta}_0$ with

$$\begin{pmatrix} (K_{u,u} \odot y_u y_u^T) & 0 \\ (K_{u,u} \odot y_u y_u^T) & y_u \\ y_u^T & 0 \end{pmatrix} \begin{pmatrix} \tilde{\alpha}_u \\ \tilde{\beta}_0 \end{pmatrix} = \begin{pmatrix} e_u - C(K_{u,C} \odot y_u y_C^T) e_C \\ e_u - C(K_{u,C} \odot y_u y_C^T) e_C \\ - C e_C^T y_C \end{pmatrix},$$

$$\tilde{\alpha}_C := \tilde{C}, \quad \tilde{\alpha}_0 := 0$$

with

$$I_0 := \{i \,|\, \alpha_i = 0\}$$
$$I_C := \{i \,|\, \alpha_i = C\}$$
$$I_u := \{i \,|\, 0 < \alpha_i < C\}$$

References

Chapelle, O. et al. (2002): *Choosing multiple parameters for support vector machines*. In: *Machine Learning* 46.1, 131–159.

Keerthi, S. S, V. Sindhwani, and O. Chapelle (2007): *An efficient method for gradient-based adaptation of hyperparameters in SVM models*. In: *Advances in neural information processing systems* 19, p. 673.