MIT 9.520/6.860
Statistical Learning Theory and Applications

Class 06: Online learning

Lorenzo Rosasco

Learning so far

ERM

$$\widehat{w}_\lambda = \operatorname*{arg\,min}_{w \in \mathbb{R}^d} \hat{L}^\lambda(w); \quad \hat{L}^\lambda(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, x_i^\top w) + \lambda \|w\|^2.$$

Gradient iterations

$$w_{t+1} = w_t - \gamma_t g_t$$

where either

$$g_t = \nabla \hat{L}^\lambda(w_t) \qquad \text{or} \qquad g_t \in \partial \hat{L}^\lambda(w_t)$$

Batch learning: Data are processed all at once

Online learning

Data are processed incrementally.

Important when:

- data are streaming,
- data-sets are large.

Goal: adapt solutions to new data without retraining from scratch.

Warming up with recursive least squares

Recall regularized least squares with n examples[1],

$$\widehat{w}^{\lambda}_{n+1} = \min_{w \in \mathbb{R}^d} \sum_{i=1}^{n} (y_i - x_i^\top w)^2 + \lambda \|w\|^2.$$

Recursive LS (regularized)

$$\begin{aligned} \widehat{\Gamma}_{t+1} &= \widehat{\Gamma}_t - \frac{\widehat{\Gamma}_t x_t x_t^\top \widehat{\Gamma}_t}{1 + x_t^\top \widehat{\Gamma}_t x_t}, \qquad \widehat{\Gamma}_0 = \frac{1}{\lambda} I \\ \widehat{w}_{t+1} &= \widehat{w}_t - \widehat{\Gamma}_{t+1}^{-1} x_t \left(x_t^\top \widehat{w}_t - y_t \right), \qquad w_0 = 0. \end{aligned}$$

- Recursive LS is online: one point added at each iteration.
- We next derive recursive LS and show that for all $t = 1, \ldots, n$

$$\widehat{w}_{t+1} = \widehat{w}^{\lambda}_{t+1}.$$

[1] Omitting $1/n$ makes the derivation easier, and the indexing $n+1$ instead of n is convenient later.

Deriving recursive least squares I

Recalling and rewriting the expression of $\widehat{w}^{\lambda}$ for t points,

$$\begin{aligned}
\widehat{w}_{t+1} &= (\widehat{X}^{\top}\widehat{X} + \lambda I)^{-1}\widehat{X}^{\top}\widehat{Y} \\
&= \widehat{\Sigma}_{t+1}^{-1} \sum_{i=1}^{t} x_i y_i \\
&= \widehat{\Sigma}_{t+1}^{-1} x_t y_t + \widehat{\Sigma}_{t+1}^{-1} \sum_{i=1}^{t-1} x_i y_i,
\end{aligned}$$

where we dropped the dependence on λ, added the one to t and let

$$\widehat{\Sigma}_{t+1} = (\widehat{X}^{\top}\widehat{X} + \lambda I).$$

Deriving recursive least squares II

Subtracting $\widehat{w}_t$ on both sides and rearranging terms, we have

$$\widehat{w}_{t+1} - \widehat{w}_t = \widehat{\Sigma}_{t+1}^{-1}(x_t y_t) + \left(\widehat{\Sigma}_{t+1}^{-1} - \widehat{\Sigma}_t^{-1}\right)\sum_{i=1}^{n-1} x_i y_i,$$

and using the identities

$$\widehat{\Sigma}_{t+1}^{-1} - \widehat{\Sigma}_t^{-1} = \widehat{\Sigma}_{t+1}^{-1}\left(\widehat{\Sigma}_t - \widehat{\Sigma}_{t+1}\right)\widehat{\Sigma}_t^{-1},$$

$$\widehat{\Sigma}_{t+1} - \widehat{\Sigma}_t = (\sum_{j=1}^{t} x_j x_j^T + \lambda I) - (\sum_{j=1}^{t-1} x_j x_j^T + \lambda I) = x_t x_t^T,$$

we have

$$\begin{aligned} \widehat{w}_{t+1} - \widehat{w}_t &= \widehat{\Sigma}_{t+1}^{-1}(x_t y_t) - \widehat{\Sigma}_{t+1}^{-1}(x_t x_t^T)\widehat{w}_t \\ &= \widehat{\Sigma}_{t+1}^{-1}\left[x_t\left(y_t - x_t^\top \widehat{w}_t\right)\right]. \end{aligned}$$

Deriving recursive least squares II

$$\widehat{w}_{t+1} = \widehat{w}_t - \widehat{\Sigma}_{t+1}^{-1}\left[x_t\left(x_t^\top \widehat{w}_t - y_t\right)\right].$$

The above expression is recursive but requires inverting $\widehat{\Sigma}_{t+1}$.

Sherman-Woodbury formula for rank 1 update

$$(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1}uv^\top A^{-1}}{1 + v^\top A^{-1}u}.$$

Let $A = \widehat{\Sigma}_t$, $u = v = x_n$, then for all $t = 1, \ldots, n$

$$\widehat{\Sigma}_{t+1}^{-1} = \widehat{\Sigma}_t^{-1} - \frac{\widehat{\Sigma}_t^{-1}x_t x_t^T \widehat{\Sigma}_t^{-1}}{1 + x_t^\top \widehat{\Sigma}_t^{-1}x_t}.$$

The proof is concluded letting $\widehat{\Gamma}_0 = \frac{1}{\lambda}I$ and $\widehat{\Gamma}_{t+1} = \widehat{\Sigma}_{t+1}^{-1}$.

Recursive least squares

$$\begin{aligned}
\widehat{\Gamma}_{t+1} &= \widehat{\Gamma}_t - \frac{\widehat{\Gamma}_t x_t x_t^T \widehat{\Gamma}_t}{1 + x_t^\top \widehat{\Gamma}_t x_t}, \qquad \widehat{\Gamma}_0 = \frac{1}{\lambda} I \\
\widehat{w}_{t+1} &= \widehat{w}_t - \widehat{\Gamma}_{t+1}^{-1} x_t \left(x_n^\top \widehat{w}_t - y_n \right), \qquad w_0 = 0.
\end{aligned}$$

- At each iteration one point is used with $O(nd)$ cost.
- The matrices $\widehat{\Gamma}_t$ need be computed/stored using $O(d^2)$ memory.

From recursive least squares to SGD

In

$$\widehat{w}_{t+1} = \widehat{w}_t - \widehat{\Gamma}_{t+1} x_t \left(x_t^\top \widehat{w}_t - y_t \right),$$

replace the matrix Γ_{t+1} with a scalar γ_t, so that

$$\widehat{w}_{t+1} = w_t + \gamma_t x_t (y_t - x_t^\top w_t).$$

The above iteration

- is recursive,
- does not require storing the matrix $\widehat{\Gamma}_t$,
- requires vector/vector, rather than matrix/vector, multiplication.

However,

$$w_{t+1} \neq \widehat{w}^{\lambda}_{t+1}.$$

A convergence analysis and a choice γ_t are needed.

SGD beyond least square

Note that

$$\nabla(y_t - x_t^\top w)^2 = -2x_t[y_t - x_t^\top w].$$

More generally, consider

$$\widehat{w}_{t+1} = \widehat{w}_t + \gamma_t \nabla \ell(y_t, x_t^\top \widehat{w}_t),$$

Note: SGD is the name used in ML, but note that

- it is not a descent method,
- subgradients are used if the loss is non differentiable.

$$\nabla \ell(y_t, x_t^\top \widehat{w}_t) \mapsto \partial \ell(y_t, x_t^\top \widehat{w}_t).$$

SGD and expected risk minimization

The name SGD stems from the observation that for all $w \in \mathbb{R}^d$,

$$\mathbb{E}_{(y_t,x_t)\sim P}[\nabla \ell(y_t, x_t^\top w)] = \nabla\, \mathbb{E}_{(y_t,x_t)\sim P}[\ell(y_t, x_t^\top w)] = \nabla[L(w)],$$

where

$$L(w) = \mathbb{E}_{(y,x)\sim P}[\ell(y, x^\top w)].$$

In this view, SGD can be seen as an approach to solve

$$\min_{w\in\mathbb{R}^d} L(w)$$

by the gradient iteration using the approximation

$$\nabla \ell(y_t, x_t^\top w) \approx \nabla L(w).$$

SGD and ERM

J random variable with uniform distribution $U([n])$ on $[n] = \{1, \dots, n\}$.

Then,

$$\hat{L}(w) = \frac{1}{n} \sum_{j=1}^{n} \ell(y_j, w^\top x_j) = \mathbb{E}_{J \sim \mathrm{U}([\mathrm{n}])}[\ell(y_J, w^\top x_J)]$$

and

$$\mathbb{E}_{J \sim \mathrm{U}([\mathrm{n}])}[\nabla \ell(y_J, x_J^\top w)] = \nabla \, \mathbb{E}_{J \sim \mathrm{U}([\mathrm{n}])}[\ell(y_J, x_J^\top w)] = \nabla[\hat{L}(w)].$$

In this view, SGD can be seen as an approach to solve

$$\min_{w \in \mathbb{R}^d} \hat{L}(w)$$

by the gradient iteration using the approximation

$$\nabla \ell(y_t, x_t^\top w) \approx \nabla \hat{L}(w).$$

SGD and risks gradients

$$\nabla \ell(y_t, x_t^\top w)$$

- Is the expected risk gradient, if we take expectation w.r.t data generating distribution:
  each point is used once (one pass SGD).

- Is the empirical risk gradient, if we take expectation w.r.t. empirical distribution:
  each point is used multiple times (multi-pass SGD).

Other SGD flavors: different sampling

$$\widehat{w}_{t+1} = \widehat{w}_t + \gamma_t \nabla \ell(y_t, x_t^\top \widehat{w}_t),$$

- Sample data uniformly at random.
- Visit data in a fixed prescribed order.
- Visit data sequentially, reshuffling after each pass.

Other SGD flavors: mini-batching

$$\widehat{w}_{t+1} = \widehat{w}_t + \gamma_t \frac{1}{b} \sum_{j=(b(t-1)+1)}^{bt} \nabla \ell(y_j, x_j^\top \widehat{w}_t),$$

- Uses a more accurate gradient estimate.
- Allows efficient use of memory.
- First step towards using distributed resources.

Other SGD flavors: momentum

$$\begin{aligned} \widehat{w}_{t+1} &= \hat{v}_t + \gamma_t \nabla \ell(y_t, x_t^\top \hat{v}_t) \\ \hat{v}_t &= \widehat{w}_t + \beta_t (\widehat{w}_t - \widehat{w}_{t-1}) \end{aligned}$$

- The momentum is $\widehat{w}_t - \widehat{w}_{t-1}$.
- Two previous iterations are used, with potential increased speed.
- Above, the momentum is used as proposed by Nesterov.

Other SGD flavors: averaging

$$\overline{w}_t = \frac{1}{t} \sum_{j=s}^{t} a_j \widehat{w}_j$$

- $s = a_t = 1$ uniform averaging.
- $s > 1$ tail averaging.
- $a_t \neq 1$ weighted (Cesàro) averages.

SGD convergence

Let F be convex G-Lipschitz, $\|w^*\| \leq B$, and

$$w_{t+1} = w_t - \gamma_t g_t, \qquad \mathbb{E}[g_t] = \nabla F(w_t).$$

If $\gamma_t = \frac{B}{G\sqrt{t}}$, $w_1 = 0$ and $\bar{w}_t = \frac{1}{t}\sum_{j=1}^{t} w_j$ then,

$$\mathbb{E}\, F(\bar{w}_t) - F(w^*) \leq \frac{BG}{\sqrt{t}}.$$

Remarks

- For constrained optimization problems over a set C, do projected gradient step

$$w_{t+1} = \text{Proj}_C \left(w_t - \gamma_t g_t \right)$$

  Proof essentially the same.

- Knowledge of G and B not necessary (with appropriate changes).

- Faster convergence under additional assumptions on F (smoothness, strong convexity).

Summing up

- Online learning algorithms.
- Recursive least squares.
- SGD: different view and different flavors.