CPSC 540: Machine Learning

Fenchel Duality and Large-Scale Kernel Methods

Mark Schmidt

University of British Columbia

Winter 2018

Outline

Motvation: Getting Rid of the Step-Size

- SVMs are a widely-used model but objective is non-differentiable.
  - We can't apply coordinate optimization or proximal-gradient or SAG.
  - The non-differentiable part is the loss, which isn't nice.
- Stochastic subgradient methods achieve $O(1/\epsilon)$ without dependence on n.
  - But choosing the step-size is painful.
- Can we develop a method where choosing the step-size is easy?
  - To do this, we first need the concept of the Lagrangian...

Lagrangian Function for Equality Constraints

- Consider minimizing a differentiable f with linear equality constraints,

$$\underset{Ax=b}{\text{argmin}}\, f(x).$$

- The Lagrangian of this problem is defined by

$$L(x,z) = f(x) + z^T(Ax - b),$$

for a vector $z \in \mathbb{R}^n$ (with A being n by d).

- At a solution of the problem we must have

$$\nabla_x L(x,z) = \nabla f(x) + A^T z = 0 \qquad \text{(gradient is orthogonal to constraints)}$$
$$\nabla_z L(x,z) = Ax - b = 0 \qquad \text{(constraints are satisfied)}$$

- So solution is stationary point of Lagrangian.

Dual Function

- But we can't just minimize with respect to x and z.
- The solution for convex f is actually a saddle point,

$$\max_z \min_x L(x, z).$$

(in cases where the $\max$ and $\min$ have solutions)

- One way to solve this is to eliminate x,

$$\max_z D(z),$$

where $D(z) = \min_x L(x, z)$ is called the dual function.

- Another method is eliminate constraints (see Michael Friedlander's course).

(find a feasible x, find basis for null-space of A, optimize f over null-space.)

Digression: Supremum and Infimum

- To handle case where $\min_x f(x)$ is not achieved for any x, we can use infimum.
- Generalization of $\min$ that includes limits:

$$\min_{x\in\mathbb{R}} x^2 = 0, \quad \inf_{x\in\mathbb{R}} x^2 = 0,$$

but

$$\min_{x\in\mathbb{R}} e^x = \text{DNE}, \quad \inf_{x\in\mathbb{R}} e^x = 0.$$

- The infimum of a function f is its largest lower-bound,

$$\inf f(x) = \max_{y|y\leq f(x)} y.$$

- The analogy for $\max$ is called the supremum $(\sup)$.

Dual function

- Even for non-smooth convex f solution is a saddle point of the Lagrangian,

$$\max_z \inf_x \underbrace{f(x) + z^T(Ax - b)}_{L(x,z)}.$$

(restricted to z where the $\max$ is finite)

- We're going to eliminate x by working with the dual function,

$$\max_z D(z),$$

with $D(z) = \inf_x \{f(x) + z^T(Ax - b)\}$.

(D is concave for any f, so $-D$ is convex)

- Why?????
  - If f is strongly-convex, dual is smooth (not obvious).
  - Dual sometimes has sparse kernel representation.
  - Dual has fewer variables if $n < d$.
  - Dual gives lower bound, $D(z) \leq f(x)$ (weak duality).
  - We can solve dual instead of primal, $D(z^*) = f(x^*)$ (strong duality).

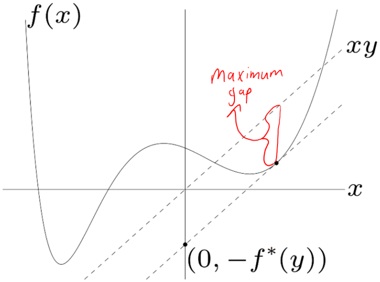
(see Michael Friedlander's class for details/conditions.)

Convex Conjugate

- The convex conjugate f^* of a function f is given by

$$f^*(y) = \sup_{x \in \mathcal{X}} \{y^T x - f(x)\},$$

where $\mathcal{X}$ is values where $\sup$ is finite.



http://www.seas.ucla.edu/~vandenbe/236C/lectures/conj.pdf

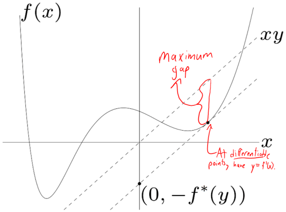
- It's the maximum that the linear function $y^T x$ can get above $f(x)$.

Convex Conjugate

- The convex conjugate f^* of a function f is given by

$$f^*(y) = \sup_{x \in \mathcal{X}} \{y^T x - f(x)\},$$

where $\mathcal{X}$ is values where $\sup$ is finite.



http://www.seas.ucla.edu/~vandenbe/236C/lectures/conj.pdf

- If f is differentable, then $\sup$ occurs at x where $y = \nabla f(x)$.
- Note that f^* is convex even if f is not (but we may lose strong duality).
- If f is convex then $f^{**} = f$ ("closed" f).

Convex Conjugate Examples

- If $f(x) = \frac{1}{2}\|x\|^2$ we have
  - $f^*(y) = \sup_x\{y^T x - \frac{1}{2}\|x\|^2\}$ or equivalently (by taking derivative and setting to 0):

$$0 = y - x,$$

and pluggin in $x = y$ we get

$$f^*(y) = y^T y - \frac{1}{2}\|y\|^2 = \frac{1}{2}\|y\|^2.$$

- If $f(x) = a^T x$ we have

$$f^*(y) = \sup_x\{y^T x - a^T x\} = \sup_x\{(y-a)^T x\} = \begin{cases} 0 & y = a \\ \infty & \text{otherwise.} \end{cases}$$

- For other examples, see Boyd & Vandenberghe.

Fenchel Dual

- In machine learning our primal problem is usually (for convex f and r)

$$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}}\, f(Xw) + r(w).$$

- If we introduce equality constraints,

$$\underset{v = Xw}{\operatorname{argmin}}\, f(v) + r(w).$$

then dual has a special form called the Fenchel dual,

$$\underset{z \in \mathbb{R}^n}{\operatorname{argmax}}\, D(z) = -f^*(-z) - r^*(X^T z),$$

where we're maximizing the (negative) convex conjugates f^* and r^*.

(bonus slide)

- If r is strongly-convex, dual will be smooth...

Fenchel Dual of SVMs

- Consider support vector machines,

$$\underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\} + \frac{\lambda}{2}\|w\|^2.$$

- The Fenchel dual is given by

$$\underset{0\leq z\leq 1}{\operatorname{argmax}} \sum_{i=1}^{n} z_i - \frac{1}{2\lambda}\underbrace{\|X^T Y z\|^2}_{z^T Y X X^T Y z},$$

  with $w^* = \frac{1}{\lambda}X^T Y z^*$ and constraints coming from $f^* < \infty$.
- A couple magical things have happened:
  - We can apply kernel trick.
  - Non-negativity makes dual variables z sparse (non-zeroes are "support vectors"):
    - Can give faster training and testing.
  - Dual is differentiable (though not strongly-convex).
    - And for this function coordinate optimization is efficient.

Stochastic Dual Coordinate Ascent

- If we have an L2-regularized linear model,

$$\operatorname*{argmin}_{w\in\mathbb{R}^d} \sum_{i=1}^n f_i(w^Tx_i) + \frac{\lambda}{2}\|w\|^2,$$

then Fenchel dual is a problem where we can apply coordinate optimization,

$$\operatorname*{argmax}_{z\in\mathbb{R}^n} - \underbrace{\sum_{i=1}^n f_i^*(z_i)}_{\text{separable}} - \frac{1}{2\lambda}\underbrace{\|X^Tz\|^2}_{z^TXX^Tz}.$$

- It's known as stochastic dual coordinate ascent (SDCA):
  - Only needs to looks at one training example on each iteration.
  - Obtains $O(\log(1/\epsilon))$ rate if ∇f_i are L-Lipschitz.
    - Performance similar to SAG for many problems, worse if $\mu >> \lambda$.
  - Obtains $O(1/\epsilon)$ rate for non-smooth f:
    - Same rate/cost as stochastic subgradient, but we can use exact/adaptive step-size.

Outline

Large-Scale Kernel Methods

- Let's go back to the basic L2-regularized least squares setting,

$$\hat{y} = \hat{K}(K + \lambda I)^{-1}y.$$

- Obvious drawback of kernel methods: we can't compute/store K.
  - It has $O(n^2)$ elements.

- Standard general approaches:
  1. Kernels with special structure.
  2. Subsampling methods.
  3. Explicit feature construction.

Kernels with Special Structure

- The bottleneck in fitting the model is $O(n^3)$ cost of solving the linear system

$$(K + \lambda I)v = y.$$

- Consider using the "identity" kernel,

$$k(x^i, x^j) = \mathbb{I}[x^i = x^j].$$

- In this case K is diagonal so we can solve linear system in $O(n)$.

- More interesting special K structures that support fast linear algebra:
  - Band-diagonal matrices.
  - Sparse matrices (via conjugate gradient).
  - Diagonal plus low-rank, $D + UV^T$.
  - Toeplitz matrices.
  - Kronecker product matrices.
  - Fast Gauss transform.

Subsampling Methods

- In subsampling methods we only use a subset of the kernels.
- For example, some loss functions have support vectors.
  - But this mainly helps at testing time, and some problems have $O(n)$ support vectors.
- Nystrom approximation chooses a random and fixed subset of training examples.
  - Many variations exist such as greedily choosing kernels.
- A common variation is the subset of regressors approach....

Subsampling Methods

- Consider partitioning our matrices as

$$K = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} = \begin{bmatrix} K_1 & K_2 \end{bmatrix}, \quad \hat{K} = \begin{bmatrix} \hat{K}_1 & \hat{K}_2 \end{bmatrix},$$

  where K_{11} corresponds to a set of m training examples
  - K is m by m, K_1 is n by m.

- In subset of regressors we use the approximation

$$K \approx K_1 K_{11}^{-1} K_1^T, \quad \hat{K} \approx \hat{K}_1 K_{11}^{-1} K_1^T.$$

- Which for L2-regularized least squares can be shown to give

$$\hat{y} = \hat{K}_1 \underbrace{(K_1^T K_1 + \lambda K_{11})^{-1} K_1^T y}_{v}.$$

- Given K_1 and K_{11}, computing v costs $O(m^2 n + m^3)$ which is cheap for small m.

Explicit Feature Construction

- In explicit feature methods, we form Z such that $Z^T Z \approx K$.
  - But where Z has a small number of columns of m.
- We then use our non-kernelized approach with features Z,

$$w = (Z^T Z + \lambda I)^{-1}(Z^T y).$$

- Random kitchen sinks approach does this for translation-invariant kernels,

$$k(x^i, x^j) = k(x^i - x^j, 0),$$

  by sampling elements of inverse Fourier transform (not obvious).
- In the special case of the Gaussian RBF kernel this gives $Z = \exp(iXR)$.
  - R is a d by m matrix with elements sampled from the Gaussian (same variance).
  - i is $\sqrt{-1}$ and $\exp$ is taken element-wise.

Summary

- Fenchel dual re-writes sum of convex functions with convex conjugates:
  - Dual may have nice structure: differentiable, sparse, coordinate optimization.
- Large-scale kernel methods is an active research area.
  - Special K structures, subsampling methods, explicit feature construction.

Bonus Slide: Fenchel Dual

- Lagrangian for constrained problem is
$$L(v, w, z) = f(v) + r(w) + z^T(Xw - v),$$
so the dual function is
$$D(z) = \inf_{v,w}\{f(v) + r(w) + z^T(Xw - v)\}$$
- For the $\inf$ wrt v we have
$$\inf_v\{f(v) - z^Tv\} = -\sup_v\{v^Tz - f(v)\} = -f^*(z).$$
- For the $\inf$ wrt w we have
$$\inf_w\{r(w) + z^TXw\} = -r^*(-X^Tz).$$
- This gives
$$D(z) = -f^*(z) - r^*(-X^Tz),$$
but we could alternately get this in terms of $-z$ by replacing $(Xw - v)$ with $(v - Xw)$ in the Lagrangian.