10-725/36-725: Convex Optimization **Fall 2016**

Lecture 6: September 19

Lecturer: Ryan Tibshirani *Scribes: Shushman Choudhury, Jun Li, Dimitrios Stamoulis*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

6.1 Review of Gradient Descent

6.1.1 Method

Review from previous lecture: For a convex and smooth function f and a starting point $x^{(0)} \in \mathbb{R}^n$, we repeat:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f(x^{(k-1)}), k = 1, 2, 3, ...$$

i.e., we repeatedly evaluate its gradient at the current point we are at, and we are moving to the direction of the negative gradient multiplied by a step size, until we converge.

6.1.2 Convergence Analysis

In previous lecture, we discussed convergence analysis covering two "regimes":

- **Under Lipschitz continuous gradient with constant** $L > 0$: We proved that we need $O(1/\epsilon)$ iterations to arrive to an ϵ-suboptimal solution $f(x^{(k)}) - f^\star \leq \epsilon$.
- **Under strong convexity**: Under Lipschitz assumption and also strong convexity of f, we saw that we need $O(log(1/\epsilon))$ iterations to arrive to an ϵ-suboptimal solution $f(x^{(k)}) - f^\star \leq \epsilon$; thus yielding a faster convergence rate, called *linear convergence.* Important caveat: The c constant of this $O(c^k)$ rate depends on the condition number of our problem, since higher number L/m results in slower rate.

6.1.3 Nestorov's Theorem

A first-order iterative method has updates $x^{(k)}$ in

$$x^{(0)} + \text{span}\{\nabla f(x^{(0)}), \nabla f(x^{(1)}), ...\nabla f(x^{(k-1)})\}$$

The following theorem provides an important result for all first-order methods.

Theorem 6.1. *For any $k \leq (n-1)/2$ and any starting point $x^{(0)}$, there is a function f in the problem class such that any first-order method satisfies*

$$f(x^{(k)}) - f^\star \geq \frac{3L||x^{(0)} - x^\star||_2^2}{32(k+1)^2}$$

This provides a lower bound on the convergence of all first-order methods. So in the context of the lecture on first order methods, it is worth noticing that gradient descent has fairly good rate (yet not the optimal compared to methods covered later on).

6.2 Gradient Boosting

6.2.1 Background on the prediction model

Key idea: To construct a flexible (non-linear) model, we model the prediction u_i of the mean of observations $y = (y_1, ...y_n) \in \mathbb{R}^n$'s as a linear combination of trees in our variable $x_i \in \mathbb{R}^p, i = 1, ...n$. That is:

$$u_i = \sum_{j=1}^{m} \beta_j \cdot T_j(x_i), i = 1, ...n$$

Example of Tree T_j

$$x_i \ = \ (x_{i1}, ...x_{ip})$$

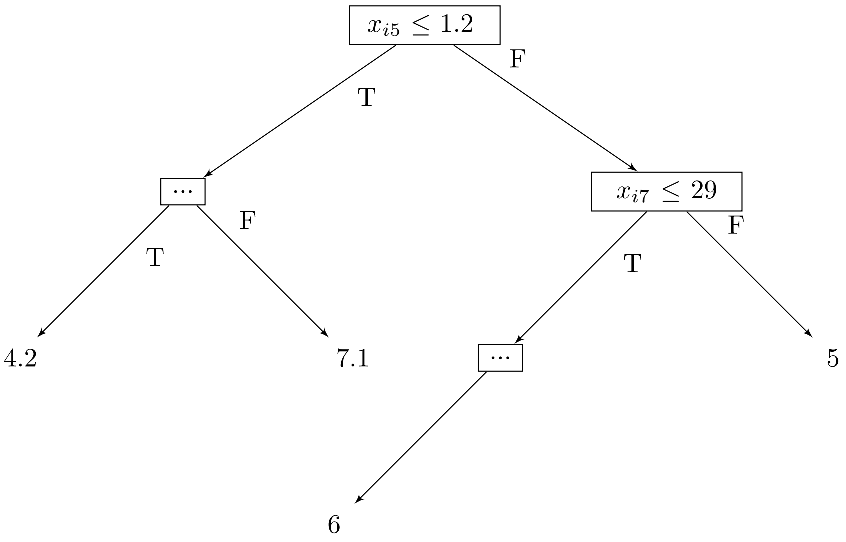

Figure 6.1: Example of (arbitrary) tree drawn in class. Each such tree T_j is passed a vector for measurement $x_i = (x_{i1}, ...x_{ip})$ and at its leaves it makes simple constant-value predictions.

So we want to solve:

$$\min_\beta \sum_{i=1}^{n} L(y_i, \sum_{j=1}^{M} \beta_j \cdot T_j(x_i))$$

Key problem: This corresponds to an enormous space of possible trees since M is huge (*e.g.,* different depths, different conditions per node etc.) and we allow arbitrary linear combinations of these trees. (N.B.: But this "expressiveness" is why gradient boosting is a powerful black-box prediction method).

6.2.2 Gradient Boosting: Key idea and Algorithm

Key insight: Gradient boosting runs gradient descent on this loss function forcing at every stage the predictions to be trees. So instead of minimizing over some function (as in default gradient descent), we minimize over the predicted values. That way, we parameterize over the n predicted values rather than the M coefficients. The key steps are listed below.

Repeat

- Compute the negative gradient ("direction d") at the latest prediction $u^{(k-1)}$:

$$d_i = -\left[\frac{\partial L(y_i, u_i)}{\partial u_i}\right]|_{u_i = u_i^{(k-1)}}, i = 1, ...n$$

- Find a tree that is close to d, according to

$$\min_{\text{trees T}} \sum_{i=1}^{n} (d_i - T(x_i))^2$$

  which we can solve greedily.

- Update the prediction:
$$u^{(k)} = u^{(k-1)} + t \cdot T_k, \text{ with step } t$$

Thus, this yields a sum of (weighted) trees.

6.3 Stochastic Gradient Descent

Stochastic gradient descent is an approximation of conventional gradient descent for the purpose of minimizing the sum of differentiable functions in an iterative manner.

6.3.1 Method

For an objective of the following form,
$$F(x) = \min_{x} \sum_{i=1}^{m} f_i(x)$$
conventional gradient descent would be prohibitive, as computing the gradient of the objective, $\nabla F(x)$, would require computing the gradients of each of the f_i's.

In stochastic gradient descent, rather than computing the gradients of all the f_i functions and adding them, a single function f_{i_k} is selected (according to some rule) at each timestep and its gradient is computed. Therefore,
$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f_{i_k}(x^{(k-1)})$$

6.3.2 Selection Rules

There are two ways (rules) by which the i_k indices are selected.

1. **Cyclic Rule**: According to this, $i_1 = 1$ and $i_{k+1} = (i_k + 1) \bmod m$. So it cycles through the indices repeatedly.
2. **Randomized Rule**: According to this, i_k is chosen from 1 to m randomly according to a uniform distribution.

6.3.3 Batch vs. Stochastic methods

Typical gradient descent is called a 'batch' method as the gradient is computed for several of the constituent functions in F. Consider a batch of size m. The amount of work done for a single step of gradient descent for such a batch is roughly equivalent to m steps of stochastic gradient descent.

At the k^{th} step, the difference in progress between the two methods is

$$\sum_{i=1}^{m} \left[\nabla f_i(x^{(k+i-1)}) - \nabla f_i(x^{(k)}\right]$$

Thus, convergence is expected if the variation of ∇f_i is not too drastic. In general, SGD has great progress far away from the optimum but slows down as it approaches it. There is also the additional benefit of memory-efficiency, which is particularly relevant for huge datasets.

6.4 Subgradients

A subgradient of a convex function $f : \mathbb{R}^n \to \mathbb{R}$ is any value $g \in \mathbb{R}^n$ such that

$$f(y) \geq f(x) + g^T(y - x),\ \forall y$$

Subgradients can be viewed as a generalization of gradients for convex functions that are not necessarily differentiable. The subgradient always exists for convex functions, except for certain pathological conditions, though it may not exist for non-convex functions, though the same definition holds. Additionally,

$$\text{f differentiable at x} \implies g = \nabla f(x) \text{ [uniquely]}$$

Example 1
Let $f : \mathbb{R} \to \mathbb{R}, f(x) = |x|$ [Convex by observation]

For any non-zero x, $|y| \geq |x| + g^T(y - x)$ and since $|y| \geq \pm y$, to make $|x| - g^T x = 0$, set $g = \text{sign}(x)$.
For $x = 0$, $|y| \geq g^T y \implies g \in [-1, 1]$.

Example 2
Let $f : \mathbb{R}^n \to \mathbb{R}, f(x) = ||x||_2$
The function $f(x)$ is differentiable for $x \neq 0$, and the sub-gradient is uniquely $\frac{x}{||x||_2}$.
For $x = 0$, $||y||_2 \geq g^T y \implies g \in \{z : ||z||_2 \leq 1\}$.

Example 3
Consider two convex and differentiable functions $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$. And consider $f = \max\{f_1(x), f_2(x)\}$.
In the case where either of f_1 or f_2 is greater than the other, f is the corresponding greater function (and thus differentiable), so the subgradient is uniquely $\nabla f_1(x)$ or $\nabla f_2(x)$, as the case may be.
For the case where $f_1(x) = f_2(x)$, then clearly both $\nabla f_1(x)$ and $\nabla f_2(x)$ are valid subgradients, but additionally, so are all points on the line segment joining them.

6.5 Subdifferentials

The subdifferential $\partial f(x)$ of a convex function f at x is the set of all subgradients:

$$\partial f(x) = \{g \in \mathbb{R}^n : g \text{ is a subgradient of } f \text{ at } x\}.$$

Properties:

- $\partial f(x)$ is a closed and convex set (even for nonconvex f).
- $\partial f(x)$ is nonempty (can be empty for nonconvex f).
- If f is differentiable at x, then $\partial = \{\nabla f(x)\}$.
- If $\partial f(x) = \{g\}$, then f is differentiable at x and $\nabla f(x) = g$.

6.5.1 Connection to convexity geometry

For a convex set $C \subseteq \mathbb{R}^n$, consider indicator function $I_C : \mathbb{R}^n \to \mathbb{R}$,

$$I_C(x) = I\{x \in C\} = \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{if } x \notin C \end{cases}$$

Lemma 6.2. *For $x \in C$, $\partial I_C(x) = \mathcal{N}_C(x)$, where $\mathcal{N}_C(x)$ is the normal cone of C at x: $\mathcal{N}_C(x) = \{g \in \mathbb{R}^n : g^T x \geq g^T y \; \forall y \in C\}$.*

Proof. First we show that if $g \in \partial I_C(x)$, then $g \in \mathcal{N}_C(x)$. By definition of subgradient,

$$I_C(y) \geq g^T(y - x), \; \forall y$$

where we have employed $I_C(x) = 0$ since $x \in C$. For any $y \in C$, $I_C(y) = 0 \geq g^T(y - x)$, i.e. $g^T x \geq g^T y$. Therefore, $g \in \mathcal{N}_C(x)$.

Next we show that if $g \in \mathcal{N}_C(x)$, then $g \in \partial I_C(x)$.

- For $y \in C$, by definition of normal cone, $g^T x \geq g^T y \Rightarrow 0 \geq g^T(y - x)$. Also $I_C(y) = 0$ because of $y \in C$. Thus $I_C(y) \geq g^T(y - x)$.
- For $y \notin C$, $I_C(y) = \infty > g^T(y - x)$, where the inequality is because for any given x and y, $g^T(y - x)$ is finite.

■

6.5.2 Subgradient calculus

Basic rules for subdifferential of convex functions:

- **Scaling**: $\partial(af) = a \cdot \partial f$ provided $a > 0$. $a > 0$ is important here because if $a < 0$, assuming f is convex, then af is concave. The equality obviously does not hold.

- **Addition**: $\partial(f_1 + f_2) = \partial f_1 + \partial f_2$, where RHS is addition of two sets: $A + B = \{a + b : a \in A, b \in B\}$.
- **Affine composition**: if $g(x) = f(Ax + b)$, then

  $$\partial g(x) = A^T \partial f(Ax + b).$$

  This is similar to the chain rule for gradient of differential functions.
- **Finite pointwise maximum**: if $f(x) = \max_{i=1,\ldots,m} f_i(x)$, then $\partial f(x)$ is the convex hull of the union of subdifferentials of all active functions (functions that achieve the maximum) at x:

  $$\partial f(x) = \text{conv}\left(\bigcup_{i:f_i(x)=f(x)} \partial f_i(x) \right).$$

  It extends the maximum of two functions in the previous section.
- **General pointwise maximum**: if $f(x) = \max_{s \in S} f_s(x)$, then

  $$\partial f(x) \supseteq \text{cl}\left\{ \text{conv}\left(\bigcup_{s:f_s(x)=f(x)} \partial f_s(x) \right) \right\},$$

  where S could be an infinite or uncountable set. Because convex hull of the union of infinite sets may not be closed, and subdifferential is a closed convex set, we take the closure of the convex hull, which is the smallest closed set containing the convex hull. Under some regularity conditions (for example, S being compact, f_s being continuous in s), we get an equality above.
- **Norms**: let $p, q > 0$ such that $\frac{1}{p} + \frac{1}{q} = 1$. Consider $f(x) = \|x\|_p$, where $\|x\|_p$ is defined as:

  $$\|x\|_p = \max_{\|z\|_q \leq 1} z^T x,$$

  then

  $$\partial f(x) = \underset{\|z\|_q \leq 1}{\text{argmax}}\, z^T x.$$

  Note this is just a special case of the general pointwise maximum above. To see this, consider $S = \{z : \|z\|_q \leq 1\}$, $f_z(x) = z^T x$. Then $f(x) = \max_{\|z\|_q \leq 1} f_z(x)$. Let $f_{z^*}(x) = f(x)$, then $z^* \in \text{argmax}_{\|z\|_q \leq 1} z^T x$. Also note $\partial f_{z^*}(x) = z^*$. Then $\cup \partial f_{z^*}(x)$ is the union of all z^*'s, i.e. $\text{argmax}_{\|z\|_q \leq 1} z^T x$. Since $z^T x$ is linear, taking the convex hull and then the closure does not add anything more. Therefore, by general pointwise maximum, $\partial f(x) = \text{argmax}_{\|z\|_q \leq 1} z^T x$.

6.5.3 Why subgradients

Subgradients are important for two reasons:

1. **Convex analysis**: optimality characterization via subgradients, monotonicity, relationship to duality. (Will be discussed more when we talk about optimality and duality.)
2. **Convex optimization**: if you can compute subgradients, then you can minimize (almost) any convex function. (It might be slow, but you can do it. Talked about more in the next lecture Subgradient method.)

6.5.4 Subgradient optimality condition

Lemma 6.3. *For any function f (convex or not), x^* is a minimizer if and only if 0 is a subgradient of f at x^*:*

$$f(x^*) = \min_x f(x) \Longleftrightarrow 0 \in \partial f(x^*)$$

Proof. $f(x^*) = \min_x f(x) \Longleftrightarrow f(y) \geq f(x^*)\ \forall y \Longleftrightarrow f(y) \geq f(x^*) + 0^T(y - x^*)\ \forall y \Longleftrightarrow 0 \in \partial f(x^*)$. ■

Note in the proof above, we did not use any convexity of f. The conclusion is fully general.