COS-511: Learning Theory **Spring 2017**

Lecture 10: Applications for Boosting & Convex Learning Problems

Lecturer: Roi Livni

10.1 Boosting– Some applications

We begin this lecture by giving some applications to the Boosting method we developed in previous lecture:

Example 10.1. *Let $\mathcal{H}$ be a class of hypotheses and set*

$$\mathcal{H}_k = \{\wedge_{i=1}^k h_i : h_i \in \mathcal{H}\}$$

Then if $\mathcal{H}$ is effieciently learnable, then $\mathcal{H}_k$ is efficiently learnable in the realizable model.

To prove the statement we rely on an Algorithm A that returns an optimal classifier against the class $\mathcal{H}$. We then show that for every distribution, the learner outputted by A is a $\frac{1}{2} - \frac{1}{k^2}$ weak learner.

Specifically, let D be a distribution and set $S = (\mathbf{x}_i, y_i)_{i=1}^m$ a sample drawn IID according to D. recall that we assume that $y_i = h^*(\mathbf{x}_i)$ for some $h^* \in \mathcal{H}_k$. Let us fix $h^* = \wedge_{j=1}^k h_j$.

For any $\mathbf{x}_i$ such that $y_i = 1$ we have that for every j, $h_j(\mathbf{x}_i) = y_i = 1$. We next show that for any distribution P at least one classifier h_j we have that : $P(h_j(\mathbf{x}) = 0 | h^*(\mathbf{x}) = 0) > 1 - \frac{1}{k^2}$:

$$\begin{aligned} 1 = P(\wedge_{j=1}^k h_j(\mathbf{x}) = 0 | h^*(\mathbf{x}) = 0) = P(\cup_{j=1}^k (h_j(\mathbf{x}) = 0) | h^*(\mathbf{x}) = 0) &\leq \sum_{j=1}^k P(h_j(\mathbf{x}) = 0 | h^*(\mathbf{x}) = 0) \quad \text{By union bound} \\ &\leq k \cdot \max_j P(h_j(\mathbf{x}) = 0 | h^*(\mathbf{x}) = 0). \end{aligned}$$

Overall we have that for any distribution P there is j such that $P(h_j(\mathbf{x}) = 0 | h^*(\mathbf{x}) = 0) \geq \frac{1}{k}$. The zero one error of j is then given by

$$\text{err}(h_j) = P(h^*(\mathbf{x}) = 0) \cdot P(h_j(\mathbf{x}) = 1 | h^*(\mathbf{x}) = 0) \leq P(h^*(\mathbf{x}) = 0)(1 - \frac{1}{k}).$$

Now suppose $P(h^*(\mathbf{x}) = 0) \geq \frac{1}{2} + \frac{1}{k^2}$, then the constant classifier $h(\mathbf{x}) = 0$ has an error of $\frac{1}{2} - \frac{1}{k^2}$ and is hence a $\frac{1}{k^2}$-weak classifier. On the other hand if not, then by equation above we have

$$\text{err}(h_j) \leq P(h^*(\mathbf{x}) = 0)(1 - \frac{1}{k}) \leq (\frac{1}{2} + \frac{1}{k^2})(1 - \frac{1}{k}) \leq \frac{1}{2} - \frac{1}{2}k^2.$$

where the inequality holds for any positive integer k.

Example 10.2. *Under appropriate assumptions, agnostic learning of halfspaces is also hard.*

The result follows from applying Thm. 8.2 and the example above to intersection of halfspaces with $k = \text{poly}(n)$. Indeed if we could learn halfspaces agnostically, we could learn the class of intersection of halfspaces to accuracy $e^{-2k^2 T}$.

10.2 Learning Convex Functions

Hardness of learning binary problems is often attributed to the *non–convexity* of the problem. This has led to the construction of what is referred to as *convex relaxations* for classification tasks. As we will see in this next part of the course, convex functions are extremely attractive as there are multiple efficient algorithms for optimizing over this class of functions (under mild assumptions).

10.2.1 Convex functions and convex sets

We next define what is a convex problem, we begin by defining convex functions:

Definition 10.1. *A function $f : X \to \mathbb{R}$ over a domain $X \subseteq \mathbb{R}^d$ is called convex if for every $0 \leq \lambda \leq 1$ and $x, y \in X$:*

$$f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y)$$

If $-f$ is a convex function, then f is called concave.

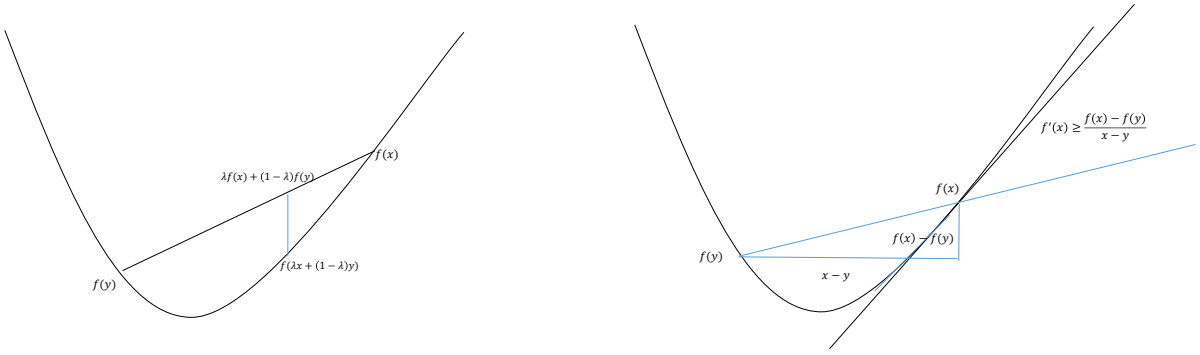
If f is a differentiable function at x and convex, then the following bound always hold:

$$f(y) - f(x) \geq \nabla f(x) \cdot (y - x) \tag{10.1}$$

When f is not differentiable at x, we define the subdifferential of f at x denoted

$$\partial f(x) = \{v : f(y) - f(x) \geq v \cdot (y - x),\ \forall y\}.$$

For a convex function the set $\partial f(x)$ is always non empty, and x is the a minimizer of f iff $0 \in \partial f(x)$.

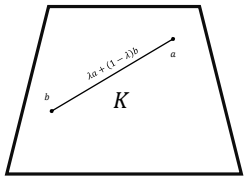


(a) $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$ (b) $f'(x) \geq \frac{f(x)-f(y)}{x-y}$

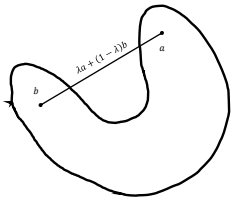
Figure 10.1: Convex Function: illustration of two characterizations

Example 10.3. *The following functions are convex*

1. ℓ_2 ***norm squared:*** *The function $\|\mathbf{x}\|^2 = \sum_{i=1}^n x_i^2$ is a convex function.*
2. ℓ_1 ***norm:*** *The function $\|\mathbf{x}\|_1 = \sum |\mathbf{x}_i|$ is known to be convex, in particular the absolute value function is convex.*
3. ℓ_p ***norm:*** *The function $\|\mathbf{x}\|_p = \sqrt[q]{\sum x_i^p}$ is always convex if $1/q + 1/p = 1$.*



(a) A convex set



(b) Not a convex set

4. ***General Norms:*** *A function $\|\cdot\|$ is called a norm if*
   - $\|x\| \geq 0$ *and* $\|x\| = 0$ *iff* $x = 0$.
   - $\|\lambda x = |\lambda|\|x\|$.
   - $\|x + y\| \leq \|x\| + \|y\|$.

   Then any norm is a convex function.
5. ***Logistic loss*** *the function $f_\alpha(x) = \ln(1 + \exp(\alpha x))$ is convex for any α.*

The following facts are usefull for the construction of new convex functions

Fact 10.1.

If f, g are convex then so is $f + g$.

If f is convex and $\alpha \geq 0$ then $\alpha \cdot f$ is convex.

If f, g are convex then $\max(f, g)$ is convex.

If f is convex and A is a linear operator then $f(A \cdot \mathbf{x})$ is also convex.

Next, we define a convex set

Definition 10.2. *A set $K \subseteq \mathbb{R}^n$ is called convex if for every $x, y \in K$ and $0 \leq \lambda \leq 1$:*

$$\lambda x + (1 - \lambda)y \in K.$$

Example 10.4. *The followin are convex sets*

1. *The positive cone: $\{\mathbf{x} : \mathbf{x}_i \geq 0 i = 1, \ldots, n\} \subseteq \mathbb{R}^n$ is convex.*
2. *The Euclidean ball: $\{\mathbf{x} : \|\mathbf{x}\| \leq 1\}$ is a convex set.*
3. *The unit–cube $\{\mathbf{x} : \max\{\mathbf{x}_i\} \leq 1\}$.*
4. *In general, if f is convex then the empigraph set $\{\mathbf{x} : f(\mathbf{x}) \leq a\}$ is a convex set.*

Definition 10.3. *A problem of the form*

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in K \end{array}$$

is called a convex program if f is convex and K is a convex set.

Equivalently, a concave program is a problem of the form

$$\begin{aligned} \text{maximize} \quad & g(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in K \end{aligned}$$

when g is concave and K is convex.

We next consider learning problems $(\mathcal{Z}, \mathcal{C}, \ell)$ In general, convex problem are considered "easy", hence it is always preferable if one can relax a problem into a convex formulation. We will develop tools for optimizing over convex functions in the next few lectures, however before we do that we will first consider a few examples and develop generalization theory for convex learning problems.

10.2.2 Convex learning problems

Roughly, convex learning problems $(\mathcal{Z}, \mathcal{C}, \ell)$ are those where $\mathcal{C}$ is identified with some convex set in Euclidean space $\mathbb{R}^d$ and $\ell(h, \mathbf{z})$ is a convex function over h for every fixed $\mathbf{z}$. The simplest examples, are linear problems over convex loss function, in future lectures we will see how this framework may be generlized to incorporate certain non-linearities operations over χ.

Example 10.5 (Linear Regression)**.** *Sometimes convex learning problems are interesting of their own merit. Such is the example of linear regression. In linear regression we observe samples $(\mathbf{x}, y) \subseteq \mathbb{R}^n \times \mathbb{R}$ and we aim to find a* linear regressor *i.e. a function $f_{\mathbf{w}}(\mathbf{x}) = y$ that would best fit the distribution:*

$$\text{minimize} \quad \frac{1}{m} \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x}^{(i)} - y_i)^2$$

Sometimes, we might wish to "regulate" the *complexity* of $\mathbf{w}$.

For example, consider a case where $\mathbf{x} = (1, x, x^2, x^3, \ldots, x^n)$, where $n \gg m$. We know that for every sample $S = \{x^{(i)}, y_i\}_{i=1}^m$, we can regress some n-degree polynomial $\mathbf{w}_0 + \mathbf{w}_1 \cdot x_i + \mathbf{w}_2 x_i^2 + \cdots \mathbf{w}_n \mathbf{x}_i^n = y_i$. Hence, with such small data set we may easily overfit.

Therefore it is customery to consider two versions of linear regression that inhibit the complexity of the model

Constrained linear regression:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{m} \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x}^{(i)} - y_i)^2 \\ \text{subject to} \quad & \|\mathbf{w}\| < B \end{aligned}$$

For some norm $\|\cdot\|$: Here we control the complexity of the model by selecting B. As we allow B to grow we allow more expressive power to the model–as we will later see, B increasing B will also increase the sample complexity of the problem. Another (and in fact equivalent) formulation is the regulerized version, which we will write for ℓ_2–regulerized linear regression (aka *Ridge Regression*):

$$\text{minimize} \frac{\lambda}{2} \|\mathbf{w}\|^2 + \quad \frac{1}{m} \sum_{i=1}^{m} (\mathbf{w} \cdot \mathbf{x}^{(i)} - y_i)^2$$

In future lecture we will see the exact effect of such regularization on the learning problem.