CSCI B609:
"Foundations of Data Science"

Lecture 13/14: Gradient Descent, Boosting and Learning from Experts

Slides at http://grigory.us/data-science-class.html

Grigory Yaroslavtsev

http://grigory.us

Constrained Convex Optimization

- Non-convex optimization is NP-hard:

$$\sum_i x_i^2 (1 - x_i)^2 = 0 \Leftrightarrow \forall i\colon\ x_i \in \{0,1\}$$

- Knapsack:
  - Minimize $\sum_i c_i x_i$
  - Subject to: $\sum_i w_i x_i \leq W$
- Convex optimization can often be solved by ellipsoid algorithm in $poly(n)$ time, but too slow

Convex multivariate functions

- Convexity:
  - $\forall x, y \in \mathbb{R}^n$: $f(x) \geq f(y) + (x - y)\nabla \mathrm{f}(\mathrm{y})$
  - $\forall x, y \in \mathbb{R}^n$, $0 \leq \lambda \leq 1$:
    $$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$$
- If higher derivatives exist:
  $$f(x) = f(y) + \nabla f(y) \cdot (x - y) + (x-y)^T \nabla^2 f(x)(x-y) + \cdots$$
- $\nabla^2 f(x)_{ij} = \dfrac{\partial^2 f}{\partial x_i \partial x_j}$ is the Hessian matrix
- f is convex iff it's Hessian is positive semidefinite, $y^T \nabla^2 f y \geq 0$ for all y.

Examples of convex functions

- ℓ_p-norm is convex for $1 \leq p \leq \infty$:

$$\left|\left|\lambda x + (1-\lambda)y\right|\right|_p \leq \left|\left|\lambda x\right|\right|_p + \left|\left|(1\ -\lambda)y\right|\right|_p$$
$$= \lambda\left|\left|x\right|\right|_p + (1\ -\lambda)\left|\left|y\right|\right|_p$$

- $f(x) = \log(e^{x_1} + e^{x_2} + \cdots e^{x_n})$

$$\max(x_1, \ldots, x_n) \leq f(x) \leq \max(x_1, \ldots, x_n) + \log n$$

- $f(x) = x^T A x$ where A is a p.s.d. matrix, $\nabla^2 f = A$
- Examples of constrained convex optimization:
  - (Linear equations with p.s.d. constraints):

  minimize: $\frac{1}{2}\mathrm{x}^T A x - b^T x$ (solution satisfies $Ax = b$)

  - (Least squares regression):

  Minimize: $\left|\left|Ax - b\right|\right|_2^2 = \mathrm{x^T A^T A\, x\ - 2\ (Ax)^T b + b^T b}$

Constrained Convex Optimization

- General formulation for convex f and a convex set K:

$$minimize: f(x) \qquad \text{subject to: } x \in K$$

- Example (SVMs):
  - Data: $X_1, \ldots, X_N \in \mathbb{R}^n$ labeled by $y_1, \ldots, y_N \in \{-1,1\}$ (spam / non-spam)
  - Find a linear model:

$$W \cdot X_i \geq 1 \Rightarrow X_i \text{ is spam}$$
$$W \cdot X_i \leq -1 \Rightarrow X_i \text{ is non-spam}$$
$$\forall i: 1 - y_i W X_i \leq 0$$

- More robust version:

$$minimize: \sum_i Loss\left(1 - W(y_i X_i)\right) + \lambda \left|\left|W\right|\right|_2$$

  - E.g. hinge loss Loss(t)=max(0,t)
  - Another regularizer: $\lambda \left|\left|W\right|\right|_1$ (favors sparse solutions)

Gradient Descent for Constrained Convex Optimization

- (Projection): $x \notin K \to y \in K$
$$\mathrm{y} = \mathrm{argmin}_{z \in K} \left| |z - x| \right|_2$$
- Easy to compute for $\left| |\cdot| \right|_2^2$: $y = x / \left| |x| \right|_2^2$
- Let $\left| |\nabla f(x)| \right|_2 \leq G, \max_{x,y \in K} \left(\left| |x - y| \right|_2 \right) \leq D.$
- Let $T = \frac{4D^2G^2}{\epsilon^2}$
- Gradient descent (gradient + projection oracles):
  - Let $\eta = D / G\sqrt{T}$
  - Repeat for $i = 0, \ldots, T$:
    - $y^{(i+1)} = x^{(i)} - \eta \nabla f\left(x^{(i)}\right)$
    - $x^{(i+1)} =$ projection of $y^{(i+1)}$ on K
  - Output $z = \frac{1}{T} \sum_i x^{(i)}$

Gradient Descent for Constrained Convex Optimization

- $\left\|x^{(i+1)} - x^*\right\|_2^2 \le \left\|y^{(i+1)} - x^*\right\|_2^2$

$$= \left\|x^{(i)} - x^* - \eta \nabla f\left(x^{(i)}\right)\right\|_2^2$$

$$= \left\|x^{(i)} - x^*\right\|_2^2 + \eta^2 \left\|\nabla f\left(x^{(i)}\right)\right\|_2^2 - 2\eta \nabla f\left(x^{(i)}\right) \cdot \left(x^{(i)} - x^*\right)$$

- Using definition of G:

$$\nabla f\left(x^{(i)}\right) \cdot \left(x^{(i)} - x^*\right) \le \frac{1}{2\eta}\left(\left\|x^{(i)} - x^*\right\|_2^2 - \left\|x^{(i+1)} - x^*\right\|_2^2\right) + \frac{\eta}{2} G^2$$

- $f\left(x^{(i)}\right) - f(x^*) \le \frac{1}{2\eta}\left(\left\|x^{(i)} - x^*\right\|_2^2 - \left\|x^{(i+1)} - x^*\right\|_2^2\right) + \frac{\eta}{2} G^2$
- Sum over $i = 1, \ldots, T$:

$$\sum_{i=1}^{T} f\left(x^{(i)}\right) - f(x^*) \le \frac{1}{2\eta}\left(\left\|x^{(0)} - x^*\right\|_2^2 - \left\|x^{(T)} - x^*\right\|_2^2\right) + \frac{T\eta}{2} G^2$$

Gradient Descent for Constrained Convex Optimization

- $\sum_{i=1}^{T} f\left(x^{(i)}\right) - f(x^*) \leq \frac{1}{2\eta}\left(\left\|x^{(0)} - x^*\right\|_2^2 - \left\|x^{(T)} - x^*\right\|_2^2\right) + \frac{T\eta}{2} G^2$
- $f\left(\frac{1}{T}\sum_i x^{(i)}\right) \leq \frac{1}{T}\sum_i f\left(x^{(i)}\right)$:
$$f\left(\frac{1}{T}\sum_i x^{(i)}\right) - f(x^*) \leq \frac{D^2}{2\eta T} + \frac{\eta}{2} G^2$$
- Set $\eta = \frac{D}{G\sqrt{T}} \Rightarrow$ RHS $\leq \frac{DG}{\sqrt{T}} \leq \epsilon$

Online Gradient Descent

- Gradient descent works in a more general case:
- $f \rightarrow$ sequence of convex functions $f_1, f_2 \dots, f_T$
- At step i need to output $x^{(i)} \in K$
- Let x^* be the minimizer of $\sum_i f_i(w)$
- Minimize regret:

$$\sum_i f_i\left(x^{(i)}\right) - f_i(x^*)$$

- Same analysis as before works in online case.

Stochastic Gradient Descent

- (Expected gradient oracle): returns g such that $\mathbb{E}_g[g] = \nabla f(x)$.
- Example: for SVM pick randomly one term from the loss function.
- Let g_i be the gradient returned at step i
- Let $f_i = g_i^T x$ be the function used in the i-th step of OGD
- Let $z = \frac{1}{T} \sum_i x^{(i)}$ and x^* be the minimizer of f.

Stochastic Gradient Descent

- **Thm.** $\mathbb{E}[f(z)] \leq f(x^*) + \frac{DG}{\sqrt{T}}$ where G is an upper bound of any gradient output by oracle.
- $f(z) - f(x^*) \leq \frac{1}{T}\sum_i (f\left(x^{(i)}\right) - f(x^*))$ (convexity)

$$\leq \frac{1}{T}\sum_i \nabla f\left(x^{(i)}\right)\left(x^{(i)} - x^*\right)$$

$= \frac{1}{T}\sum_i \mathbb{E}\left[g_i^T (x^{(i)} - x^*)\right]$ (grad. oracle)

$$= \frac{1}{T}\sum_i \mathbb{E}[f_i(x^{(i)}) - f_i(x^*)]$$

$$= \frac{1}{T}\mathbb{E}[\sum_i f_i(x^{(i)}) - f_i(x^*)]$$

- $\mathbb{E}[]$ = regret of OGD , always $\leq \epsilon$

VC-dim of combinations of concepts

- For $\boldsymbol{k}$ concepts $h_1, \ldots, h_{\boldsymbol{k}}$ + a Boolean function f:

$comb_f(h_1, \ldots\ h_{\boldsymbol{k}}) = \{x \in X: f(h_1(x), \ldots h_{\boldsymbol{k}}(x)) = 1\}$

- Ex: H = lin. separators, $f = AND$ / $f = Majority$
- For a concept class H + a Boolean function f:

$$COMB_{f,\boldsymbol{k}}(H) = \{comb_f(h_1, \ldots\ h_{\boldsymbol{k}}): h_i \in H\}$$

- **Lem**. If VC-dim(H)= $\boldsymbol{d}$ then for any f:

$$VC\text{-dim}\left(COMB_{f,\boldsymbol{k}}(H)\right) \leq O(\boldsymbol{kd} \log(\boldsymbol{kd}))$$

VC-dim of combinations of concepts

- **Lem**. If VC-dim(H)$= d$ then for any f:

$$VC\text{-dim}\left(COMB_{f,\boldsymbol{k}}(H)\right) \leq O(\boldsymbol{kd}\log(\boldsymbol{kd}))$$

- Let $n = VC\text{-dim}\left(COMB_{f,\boldsymbol{k}}(H)\right)$
- $\Rightarrow \exists$ set S of n points shattered by $COMB_{f,\boldsymbol{k}}(H)$
- Sauer's lemma $\Rightarrow \leq n^d$ ways of labeling S by H
- Each labeling in $COMB_{f,\boldsymbol{k}}(H)$ determined by $\boldsymbol{k}$ labelings of S by $H \Rightarrow \leq (n^{\boldsymbol{d}})^{\boldsymbol{k}} = n^{\boldsymbol{kd}}$ labelings
- $2^n \leq n^{\boldsymbol{kd}} \Rightarrow n \leq \boldsymbol{kd}\log n \Rightarrow n \leq 2\boldsymbol{kd}\log \boldsymbol{kd}$

Back to the batch setting

- Classification problem
  - Instance space X: $\{0,1\}^{\boldsymbol{d}}$ or $\mathbb{R}^{\boldsymbol{d}}$ (feature vectors)
  - Classification: come up with a mapping $X \rightarrow \{0,1\}$
- Formalization:
  - Assume there is a probability distribution D over X
  - $\boldsymbol{c}^*$= "target concept" (set $\boldsymbol{c}^* \subseteq X$ of positive instances)
  - Given labeled i.i.d. samples from D produce $\boldsymbol{h} \subseteq X$
  - **Goal:** have $\boldsymbol{h}$ agree with $\boldsymbol{c}^*$ over distribution D
  - Minimize: $err_D(\boldsymbol{h}) = \Pr_D[\boldsymbol{h}\,\Delta\,\boldsymbol{c}^*]$
  - $err_D(\boldsymbol{h})$ = "true" or "generalization" error

Boosting

- **Strong learner**: succeeds with prob. $\geq 1 - \epsilon$
- **Weak learner**: succeeds with prob. $\geq \frac{1}{2} + \gamma$
- **Boosting (informal):** weak learner that works under any distribution $\Rightarrow$ strong learner
- **Idea:** run weak leaner $\boldsymbol{A}$ on sample S under <u>reweightings</u> focusing on misclassified examples

Boosting (cont.)

- H = class of hypothesis produced by $\boldsymbol{A}$
- Apply majority rule to $h_1, \ldots, h_{\boldsymbol{t_0}} \sim H$:

$$\text{VC-dim} \leq O(\boldsymbol{t_0}\text{VC-dim}(H)\log(\boldsymbol{t_0}\text{VC–dim}(H)))$$

Algorithm:

- Given $S = (x_1, \ldots, x_n)$ set $w_i = 1$ in $\boldsymbol{w} = (w_1, \ldots, w_n)$
- For $t = 1, \ldots, \boldsymbol{t_0}$ do:
  - Call weak learner on $(S, \boldsymbol{w}) \Rightarrow$ hypothesis h_t
  - For misclassified x_i multiply w_i by $\alpha = (\frac{1}{2} + \gamma)/(\frac{1}{2} - \gamma)$
- Output: $\text{MAJ}(h_1, \ldots, h_{\boldsymbol{t_0}})$

Boosting: analysis

- **Def (γ-weak learner on sample):** For labeled examples x_i weighted by w_i with weight of correct

$$\geq \left(\frac{1}{2} + \gamma\right) \sum_{i=1}^{n} w_i$$

- **Thm.** If A is γ-weak learner on $S \Rightarrow$

for $\boldsymbol{t_0} = O\left(\frac{1}{\gamma^2} log\ n\right)$ boosting achieves 0 error on S.

- **Proof.** m = # mistakes of the final classifier
  - Each was misclassified $\geq \frac{\boldsymbol{t_0}}{2}$ times $\Rightarrow$ weight $\geq \alpha^{\boldsymbol{t_0}/2}$
  - Total weight $\geq m\alpha^{\boldsymbol{t_0}/2}$
  - Total weight at $t = \mathrm{W(t)}$
  - $W(t+1) \leq \left(\alpha\left(\frac{1}{2} - \gamma\right) + \left(\frac{1}{2} + \gamma\right)\right) W(t) = (1 + 2\gamma) W(t)$

Boosting: analysis (cont.)

- $W(0) = n \Rightarrow W(\boldsymbol{t_0}) \leq n(1+2\gamma)^{\boldsymbol{t_0}}$
- $m\alpha^{\boldsymbol{t_0}/2} \leq W(\boldsymbol{t_0}) \leq n(1+2\gamma)^{\boldsymbol{t_0}}$
- $\alpha = (\frac{1}{2}+\gamma)/(\frac{1}{2}-\gamma) = (1+2\gamma)/(1-2\gamma)$
- $m \leq n(1-2\gamma)^{\boldsymbol{t_0}/2}\,(1+2\gamma)^{\boldsymbol{t_0}/2} = n(1-4\gamma^2)^{\boldsymbol{t_0}/2}$
- $1-x \leq e^{-x} \Rightarrow\ m \leq ne^{-2\gamma^2 \boldsymbol{t_0}} \Rightarrow \boldsymbol{t_0} = O\left(\frac{1}{\gamma^2}\log n\right)$

Comments:

- Applies even if the weak learners are **adversarial**
- VC-dim bounds $\Rightarrow n = \tilde{O}\left(\frac{1}{\epsilon}\frac{VC\text{−}\dim(H)}{\gamma^2}\right)$