CPSC 540: Machine Learning

How Much Data?

Winter 2020

Admin

- Registration forms:
  - I will sign them at the end of class (need to submit prereq form first).
- Website/Piazza:
  - http://www.cs.ubc.ca/~schmidtm/Courses/540-W20
  - https://piazza.com/ubc.ca/winterterm22019/cpsc540
- Tutorials: start today after class (no need to formally register).
- Assignment 1 due Friday of next week.
  - Gradescope submission instructions coming soon.
  - Prereq form submitted separately with assignment.

Last Time: Strict/Strong Convexity

- We discussed 3 levels of convexity, and their implications:
  - Convexity: all stationary points are global minimum (may be none or ∞).
  - Strict convexity: there is at most one stationary point (may be 0 or 1).
  - Strong convexity: there is exactly one stationary point (for closed domain).

- For twice-differentiable functions ("C^2"), related to Hessian:
  - Convexity: Hessian eigenvalues are non-negative everywhere. $\nabla^2 f(w) \succeq 0$
  - Strict convexity: eigenvalues are positive everywhere. $\nabla^2 f(w) \succ 0$
  - Strong convexity: eigenvalues are at least $\mu > 0$ everywhere. $\nabla^2 f(w) \succeq \mu I$

The Question I Hate the Most…

- How much data do we need?

- A difficult if not impossible question to answer.

- My usual answer: “more is better”.
  - With the warning: “as long as the quality doesn’t suffer”.

- Another popular answer: “ten times the number of features”.

A Discrete Sanity Check: Coupon Collecting

- Assume we have a categorical variable with 50 possible values:
  - {Alabama, Alaska, Arizona, Arkansas,…}.
- Assume each category has probability of 1/50 of being chosen:
  - How many examples do we need to see before we expect to see them all?
- Expected value is ~225.
- Coupon collector problem: O(n log n) in general.
  - Gotta Catch'em all!
- Obvious sanity check, is need more samples than categories:
  - Situation is worse if they don't have equal probabilities.
  - Typically want to see categories more than once to learn anything.

The Question I Hate the Most…

- Let's assume you have a new supervised learning application.
  - But you have no data.
- You have some way to collect IID samples.
  - So you have to decide how much data to collect.

- Since it's supervised learning, our goal is to minimize a test error:

$$\tilde{f}(w) = \mathbb{E}[f_i(w)] \quad \text{"test error"}$$

  - Expected loss over IID examples from the test distribution.
  - Here, $f_i(w)$ could be the squared error or some other loss.

Usual Approach: Collect Data then Optimize

- We want to minimize the test error (which we cannot compute):

$$\tilde{f}(w) = \mathbb{E}[f_i(w)] \quad \text{"test error"}$$

- We approximate this with training error over 'n' IID samples:

$$f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w) \quad \text{"train error"}$$

- And we need to decide how large 'n' should be.

- But first, let's quickly review stochastic gradient descent (SGD).
  - Among most common approaches for minimizing the training erorr.

1-Slide Review of Stochastic Gradient Descent (SGD)

- To optimize training error, could use stochastic gradient descent:

$$w^{k+1} = w^k - \alpha_k \nabla f_{i_k}(w^k)$$

  - This generates a sequence of iterates w^0, w^1, w^2,…
  - We have a sequence of step sizes α_k.
  - Each iteration 'k' chooses uses a random training example i_k.
    - Based on an unbiased estimate of the gradient of the training error (uniform i_k):

$$E[\nabla f_i(w)] = \sum_{i=1}^{n} p(i) \nabla f_i(w) = \sum_{i=1}^{n} \left(\frac{1}{n}\right) \nabla f_i(w) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w) = \nabla f(w)$$

  - Converges to a stationary point (under reasonable assumptions) if: $\dfrac{\sum_{k=1}^{\infty} \alpha_k^2}{\sum_{k=1}^{\infty} \alpha_k} = 0$
    - Typical choices: α_k= O(1/k) or α_k= O(1/$\sqrt{k}$) which is more robust.

SGD Speed of Convergence (Training Error)

- "How much data" can be related to "how fast does SGD converge"?

- Assumptions:
  - 'f' is strongly-convex: $\nabla^2 f(w) \succeq \mu I$
  - 'f' is strongly-smooth: $LI \succeq \nabla^2 f(w)$
  - "Variance" of gradients is bounded: $\frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(w) - \nabla f(w)\|^2 \leq \sigma^2$

- Under these assumptions (and suitable α_k):
  - $E[f(w^k)] - f^* = O(1/k)$, where f* is training error of the global optimum.
  - Implies we need $k=O(1/\epsilon)$ iterations to have $f(w^k) - f^* \leq \epsilon$.

Training Error vs. Testing Error

- We don't care about training error, we want to minimize test error.
  - And our goal was to decide how many examples 'n' to collect.

- We considered SGD on collected data (Approach 1):
  - Choose a random training example i_k (among the 'n' training examples).
  - Perform the SGD step.

- Now consider SGD while collecting data (Approach 2):
  - Collect a new random example i_k (IID from the true distribution).
  - Perform the SGD step.

- Approach 1 uses unbiased estimates of training error gradient.
- Approach 2 uses unbiased estimates of test error gradient.

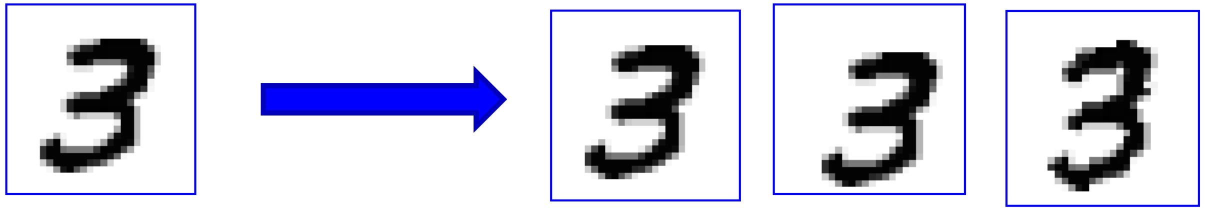
SGD Speed of Convergence (Test Error)

- With Approach 1, train error after 'k' iterations is O(1/k).
- With Approach 2, test error after 'k' iterations is O(1/k).
  - And we are using 1 new example on each iteration.
  - So with 'n' examples, this approach has test error of O(1/n).
  - And we need $n=O(1/\epsilon)$ training examples to get within ϵ of best test error.

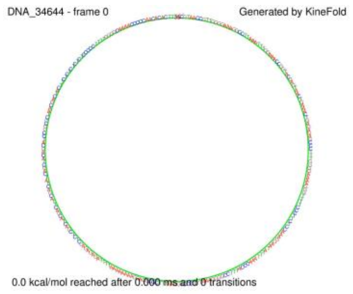
- Notice that there is no overfitting.
  - Approach 2 is doing SGD on the test error.
  - It's like doing SGD with $n=\infty$, where train error = test error.

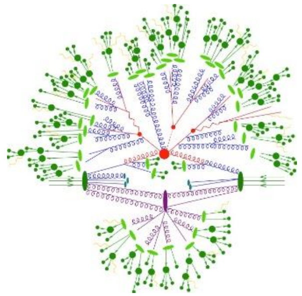
Scenarios where you can use Approach 2

- Here are some scenarios where you effectively have "n = ∞":
  - A dataset that is so large we cannot even go through it once (Gmail).
  - A function you want to minimize that you can't measure without noise.
  - You want to encourage invariance with a continuous set of transformation:
    - You consider infinite number of translations/rotations instead of a fixed number.



  - Learning from simulators with random numbers (physics/chem/bio):





http://kinefold.curie.fr/cgi-bin/form.pl
https://sciencenode.org/feature/sherpa-and-open-science-grid-predicting-emergence-jets.php

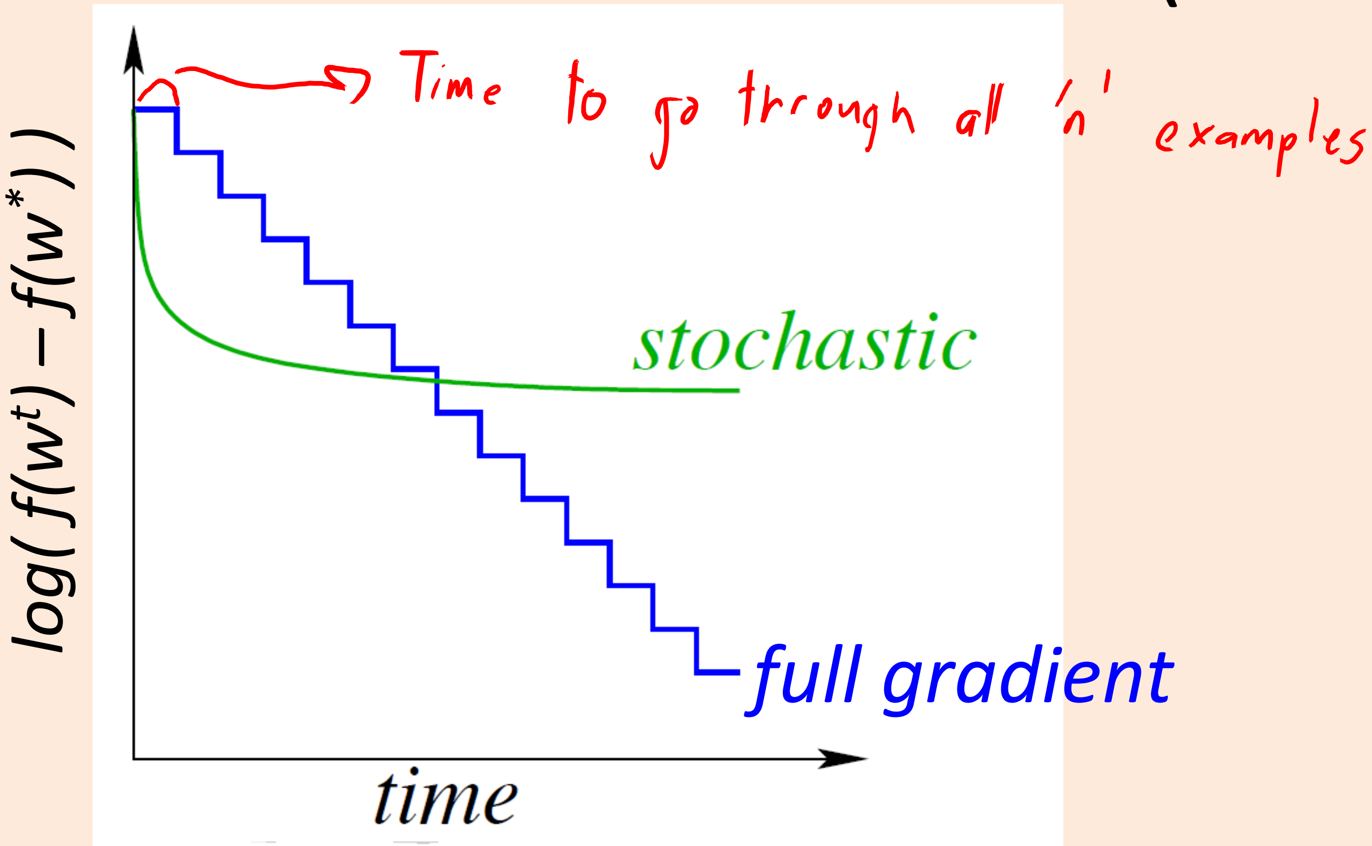
One-Pass SGD, Multi-Pass, and Caveats

- One-pass SGD:
  - If you already have a training set, you can simulate 'n' steps of Approach 2.
  - Go through your 'n' examples once, doing SGD step on each example.
    - Gets within O(1/n) of optimal test error.

- Under (ugly) assumptions, this "O(1/n) rate with 'n' examples" is unimprovable.
  - Even for methods that go through the dataset more than once or that minimize train error.

- In practice: one-pass SGD often doesn't work well.
  - Doing multiple passes almost always helps.
  - Multiple passes can potentially improve constants in O(1/n) rate.
  - One-pass SGD is also very sensitive to the step-size.
  - Our "loss" might not be the error. For example, 0-1 error is approximated by logistic loss.
  - Some recent works have been exploring assumptions where O(1/n) is improvable.
  - So if you have n=∞, but finite time: may be better to work with large-but-finite dataset.
    - "Optimize better on less data".

Digression: Gradient Descent vs. SGD (Finite Data)



- 2012: methods with cost of stochastic gradient, progress of full gradient.
  - Key idea: if ‘n’ is finite, build an estimator of gradient whose variance goes to 0.
  - First was stochastic average gradient (SAG), “low-memory” version is SVRG.

A Practical Answer to "How Much Data"?

- Whether we use one-pass or SGD or minimize training error,

  E[test error of model fit on training set] – (best test error in class) = $O(1/n)$.

  (under reasonable assumptions, and with parametric model)

- You rarely know the constant factor, but this gives some guidelines:
  - Adding more data helps more on small datasets than on large datasets.
    - Going from 10 training examples to 20, difference with best possible error gets cut in half.
      - If the best possible error is 15% you might go from 20% to 17.5% (this does **not** mean 20% to 10%).
    - Going from 110 training examples to 120, gap only goes down by ~10%.
    - Going from 1M training examples to 1M+10, you won't notice a change.
  - Doubling the data size cuts the error in half:
    - Going from 1M training to 2M training examples, gap gets cut in half.
    - If you double the data size and your test error doesn't improve, more data might not help.