CPSC 340:
Machine Learning and Data Mining

Maximum Likelihood Estimation

Fall 2018

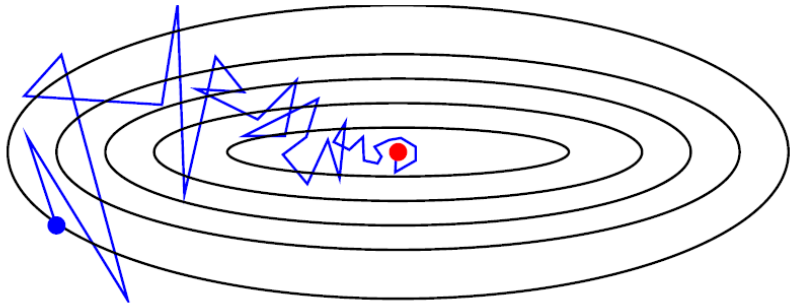
Admin

- Assignment 4:
  - Due Friday.
- Midterm:
  - Can view exam during my office hours today or Mike's office hours Friday.
- 532M Projects:
  - "No news is good news": e-mails sent.

Last Time: Stochastic Gradient

- Stochastic gradient minimizes average of smooth functions:

$$f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$$



  - Function $f_i(w)$ is error for example 'i'.

- Iterations perform gradient descent on one random example 'i':

$$w^{t+1} = w^t - \alpha^t \nabla f_i(w^t)$$

  - Cheap iterations even when 'n' is large, but doesn't always decrease 'f'.
  - But solves problem if α^t goes to 0 at an appropriate rate.
    - Classic theory says use $\alpha^t = O(1/t)$, new theory/practice uses $O(1/\sqrt{t})$ or constant.

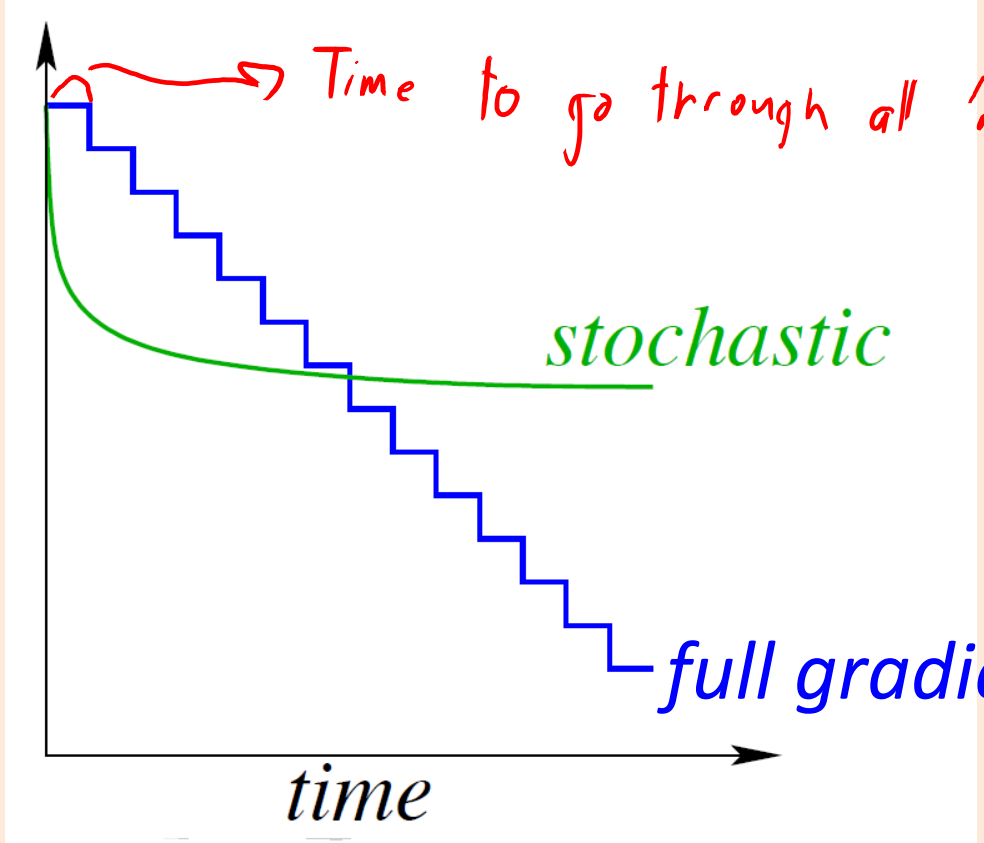
A Practical Strategy for Deciding When to Stop

- In gradient descent, we can stop when gradient is close to zero.

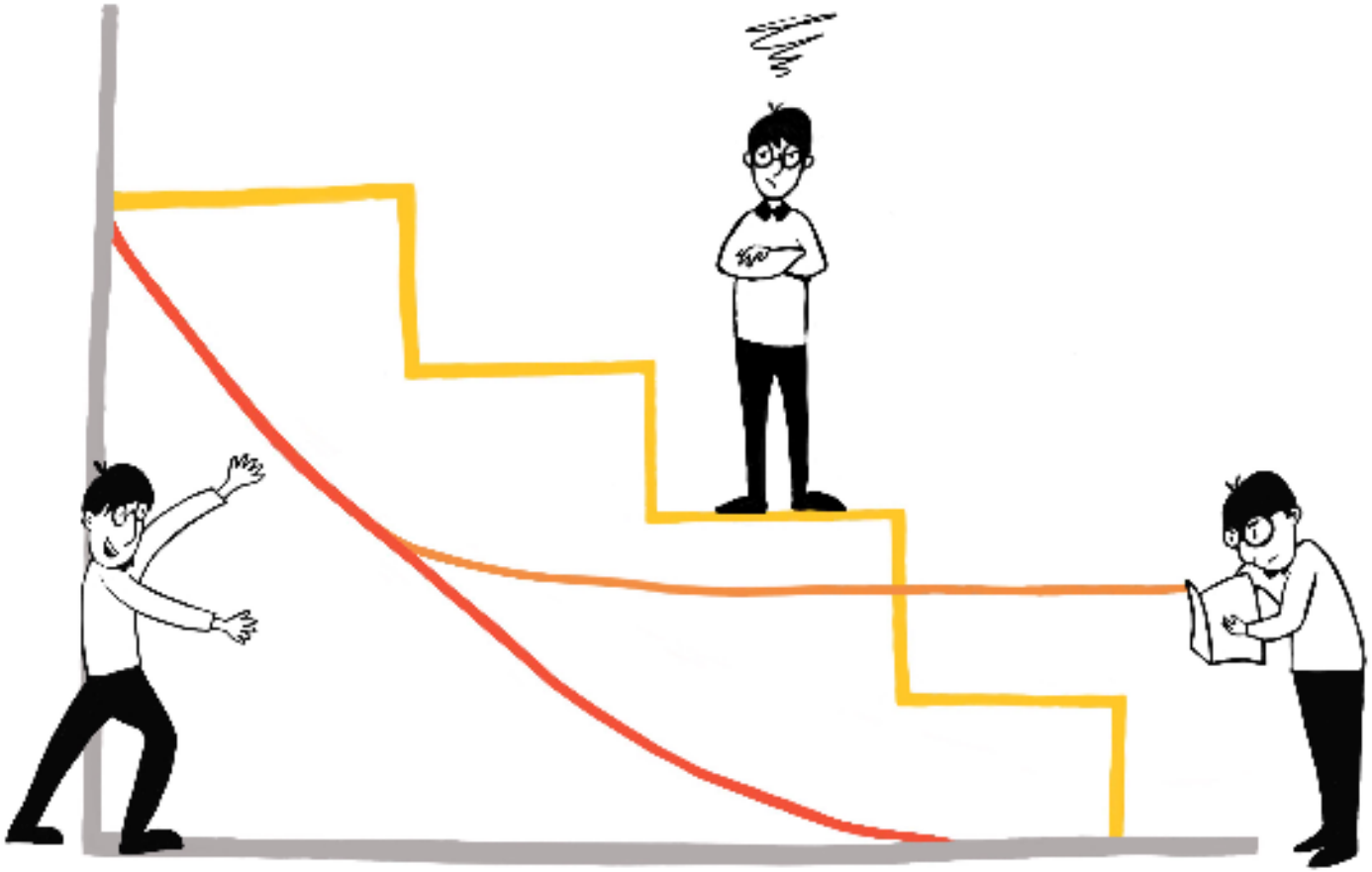
- In stochastic gradient:
  - Individual gradients don't necessarily go to zero.
  - We can't see full gradient, so we don't know when to stop.

- Practical trick:
  - Every 'k' iterations (for some large 'k'), measure validation set error.
  - Stop if the validation set error "isn't improving".
    - We don't check the gradient, since it takes a lot longer for the gradient to get small.

Gradient Descent vs. Stochastic Gradient



- 2012: methods with cost of stochastic gradient, progress of full gradient.
  - Key idea: if 'n' is finite, you can use a memory instead of having α_t go to zero.
  - First was stochastic average gradient (SAG), "low-memory" version is SVRG.



This graph shows how algorithms have become fast and more efficient over time. The horizontal axis represents time and the vertical axis represents error. Older algorithms (yellow) were very slow but had very little error. Faster algorithms were created by only analyzing some of the data (orange). The method was faster but had an accuracy limit. Schmidt's algorithm is faster and has no accuracy limit. *Aiken Lao / The Ubyssey*

https://www.ubyssey.ca/science/schmidt-sloan-fellowship/

Stochastic Gradient with Infinite Data

- Amazing property of stochastic gradient:
  - The classic convergence analysis does not rely on 'n' being finite.
- Consider an infinite sequence of IID samples.
  - Or any dataset that is so large we cannot even go through it once.
  - Or a function you want to minimize that you can't measure without noise.
- Approach 1 (exact optimization on finite 'n'):
  - Grab 'n' data points, for some really large 'n'.
  - Fit a regularized model on this fixed dataset ("empirical risk minimization").
- Approach 2 (stochastic gradient for 'n' iterations):
  - Run stochastic gradient iteration for 'n' iterations.
  - Each iterations considers a new example, never re-visit any example.

Stochastic Gradient with Infinite Data

- Approach 2 only looks at a data point once:
  - Each example is an unbiased approximation of test data.

- So Approach 2 is doing stochastic gradient on test error:
  - It cannot overfit.

- Up to a constant, Approach 1 and 2 have same test error bound.
  - This is sometimes used to justify SG as the “ultimate” learning algorithm.
    - “Optimal test error by computing gradient of each example once!”
  - In practice, Approach 1 usually gives lower test error.
    - The constant factor matters!

(pause)

Motivation for Learning about MLE and MAP

- Next topic: maximum likelihood estimation (MLE) and MAP estimation.
  - Crucial to understanding advanced methods, notation can be difficult at first.

- Why are we learning about these?
  - Justifies the naïve Bayes "counting" estimates for probabilities.
  - Shows the connection between least squares and the normal distribution.
  - Makes connection between "robust regression" and "heavy tailed" probabilities.
  - Shows that regularization and Laplace smoothing are doing the same thing.
  - Gives interpretation of $w^T x_i$ in logistic regression in terms of probabilities.
  - Gives a way to write complicated ML problems as optimization problems.
    - How do you define a loss for "number of Facebook likes" or "1-5 star rating"?
  - Crucial to understanding advanced methods.

The Likelihood Function

- Suppose we have a dataset ‘D’ with parameters ‘w’.

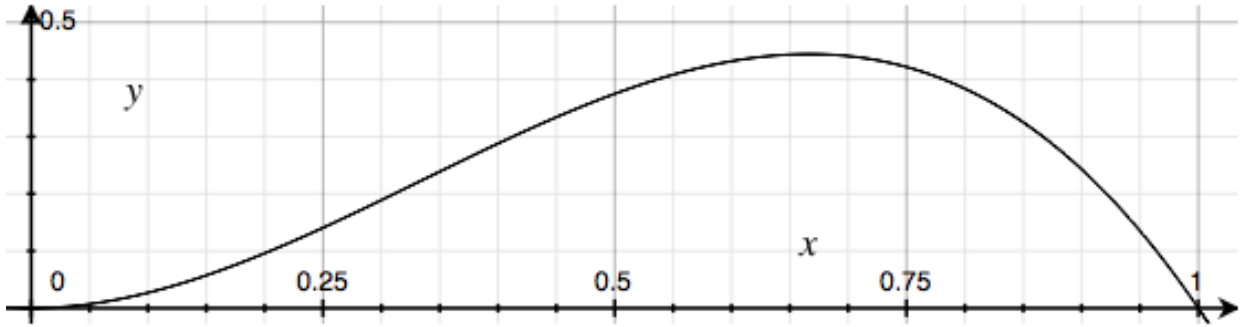
- For example:
  - We flip a coin three times and obtain D={“heads”, “heads”, “tails”}.
  - The parameter ‘w’ is the probability that this coin lands “heads”.

- We define the likelihood as a probability mass function $p(D \mid w)$.
  - “Probability of seeing this data, given the parameters”.
  - If ‘D’ is continuous it would be a probability “density” function.

- If this is a “fair” coin (meaning it lands “heads” with probability 0.5):
  - The likelihood is $p(HHT \mid w=0.5) = (1/2)(1/2)(1/2) = 0.125$.
  - If w = 0 (“always lands tails”), then $p(HHT \mid w = 0) = 0$ (data is less likely for this ‘w’).
  - If w = 0.75, then $p(HHT \mid w = 0.75) = (3/4)(3/4)(1/4) \approx 0.14$ (data is more likely).

Maximum Likelihood Estimation (MLE)

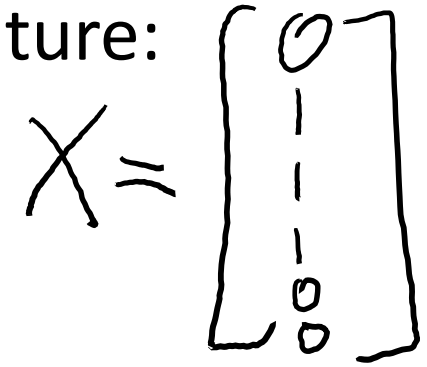
- We can plot the likelihood p(HHT | w) as a function of 'w':



- Notice:
  - Data has probability 0 if w=0 or w=1 (since we have 'H' and 'T' in data).
  - Data doesn't have highest probability at 0.5 (we have more 'H' than 'T').
  - This is a probability distribution over 'D', not 'w' (area isn't 1).
- Maximum likelihood estimation (MLE):
  - Choose parameters that maximize the likelihood: $\hat{w} \in \operatorname{argmax}_w \{ p(D|w) \}$
    - In this example, MLE is 2/3.

MLE for Binary Variables (General Case)

- Consider a binary feature:



- Using 'w' as "probability of 1", the maximum likelihood estimate is:

$$\hat{w} = \frac{\#\text{ of ones}}{\#\text{ of examples}}$$

- This is the "estimate" for the probabilities we used in naïve Bayes.
  - The conditional probabilities we used in naïve Bayes are also MLEs.
    - The derivation is tedious, but if you're interested I put it here.

(pause)

Maximum Likelihood Estimation (MLE)

- Maximum likelihood estimation (MLE) for fitting probabilistic models.
  - We have a dataset D.
  - We want to pick parameters 'w'.
  - We define the likelihood as a probability mass/density function p(D | w).
  - We choose the model $\hat{w}$ that maximizes the likelihood:

$$\hat{w} \in \arg\max_{w} \{p(D|w)\}$$

- Appealing "consistency" properties as n goes to infinity (take STAT 4XX).
  - "This is a reasonable thing to do for large data sets".

Least Squares is Gaussian MLE

- It turns out that most objectives have an MLE interpretation:
  - For example, consider minimizing the squared error:

$$f(w) = \frac{1}{2}\|Xw - y\|^2$$

  - This is MLE of a linear model under the assumption of IID Gaussian noise:

$$y_i = w^T x_i + \varepsilon_i$$

where each ε_i is sampled independently from standard normal

    - "Gaussian" is another name for the "normal" distribution.
  - Remember that least squares solution is called the "normal equations".

Minimizing the Negative Log-Likelihood (NLL)

- To maximize likelihood, usually we equivalently minimize the negative "log-likelihood" (NLL):
  - "Log-likelihood" is short for "logarithm of the likelihood".

$$\hat{w} \in \operatorname{argmax}_w \{p(D|w)\} \equiv \operatorname{argmin}_w \{-\log(p(D|w))\}$$

"equivalent"

- Why are these equivalent?
  - Logarithm is monotonic: if $\alpha > \beta$, then $\log(\alpha) > \log(\beta)$.
    - So location of maximum doesn't change if we take logarithm.
  - Changing sign flips max to min.

- See Max and Argmax notes if this seems strange.

Minimizing the Negative Log-Likelihood (NLL)

- We use logarithm because it turns multiplication into addition:

$$\log(\alpha\beta) = \log(\alpha) + \log(\beta)$$

- More generally: $\log\left(\prod_{i=1}^{n} a_i\right) = \sum_{i=1}^{n} \log(a_i)$

- If data is 'n' IID samples then $p(D|w) = \prod_{i=1}^{n} p(D_i|w)$

$p(D_i|w)$ → likelihood of example 'i'

and our MLE is $\hat{w} \in \underset{w}{\text{argmax}} \left\{ \prod_{i=1}^{n} p(D_i|w) \right\} \equiv \underset{w}{\text{argmin}} \left\{ -\sum_{i=1}^{n} \log(p(D_i|w)) \right\}$

Least Squares is Gaussian MLE (Gory Details)

- Let's assume that $y_i = w^T x_i + \varepsilon_i$, with ε_i following standard normal:

$$p(\varepsilon_i) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{\varepsilon_i^2}{2}\right)$$

also known as "Gaussian" distribution

- This leads to a Gaussian likelihood for example 'i' of the form:

$$p(y_i \mid x_i, w) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)$$

- Finding MLE (minimizing NLL) is least squares:

$$f(w) = -\sum_{i=1}^{n} \log(p(y_i \mid w, x_i))$$

$$= -\sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)\right)$$

$$= -\sum_{i=1}^{n} \left[\log\left(\frac{1}{\sqrt{2\pi}}\right) + \log\left(\exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)\right)\right]$$

constant in 'w'

operations cancel

$$= -\sum_{i=1}^{n} \left[(\text{constant}) - \frac{1}{2}(w^T x_i - y_i)^2\right]$$

$$= (\text{constant}) + \frac{1}{2}\sum_{i=1}^{n} (w^T x_i - y_i)^2$$

$$= (\text{constant}) + \frac{1}{2}\|Xw - y\|^2$$

Loss Functions and Maximum Likelihood Estimation

- So least squares is MLE under Gaussian likelihood.

$$\text{If } p(y_i \mid x_i, w) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(w^T x_i - y_i)^2}{2}\right)$$

$$\text{then MLE of } w \text{ is minimum of } f(w) = \frac{1}{2}\|Xw - y\|^2$$

- With a Laplace likelihood you would get absolute error.

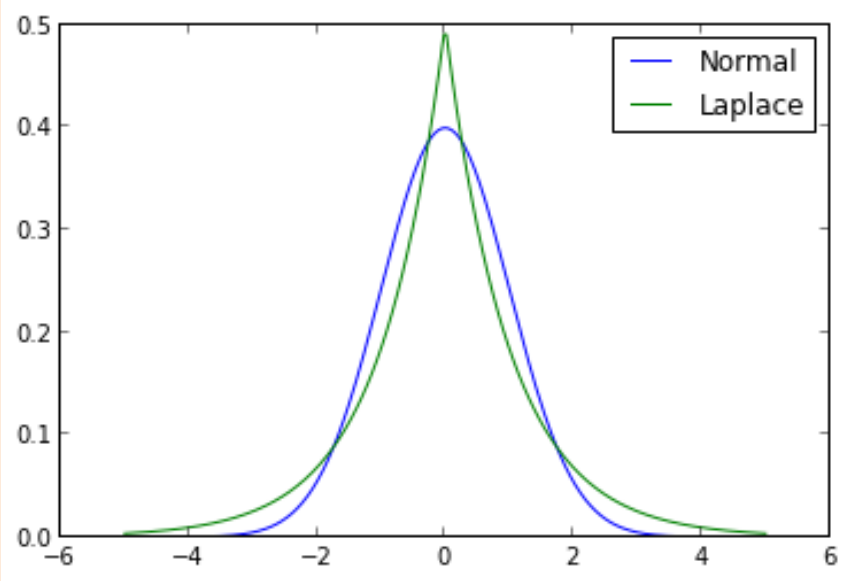
$$\text{If } p(y_i \mid x_i, w) = \frac{1}{2} \exp(-|w^T x_i - y_i|)$$

$$\text{then MLE is minimum of } f(w) = \|Xw - y\|_1$$

- Other likelihoods lead to different errors ("sigmoid" -> logistic loss).

“Heavy” Tails vs. “Light” Tails

- We know that L1-norm is more robust than L2-norm.
  - What does this mean in terms of probabilities?



Here "tail" means "mass of the distribution away from the mean!"

  - Gaussian has “light tails”: assumes everything is close to mean.
  - Laplace has “heavy tails”: assumes some data is far from mean.
  - Student ‘t’ is even more heavy-tailed/robust, but NLL is non-convex.

http://austinrochford.com/posts/2013-09-02-prior-distributions-for-bayesian-regression-using-pymc.html

Summary

- SAG and other methods fix SG convergence for finite datasets.
- Infinite datasets can be used with SG and do not overfit.
- Maximum likelihood estimate viewpoint of common models.
  - Objective functions are equivalent to maximizing $p(y, X \mid w)$ or $p(y \mid X, w)$.

- Next time:
  - How does regularization fit it?