Introduction to Machine Learning

Optimization

Mingchen Gao

Computer Science & Engineering
State University of New York at Buffalo
Buffalo, NY, USA
mgao8@buffalo.edu
Slides Adapted from Varun Chandola

University at Buffalo
Department of Computer Science and Engineering
School of Engeering and Applied Sciences

Outline

Machine Learning as Optimization
Convex Optimization
Gradient Descent
Issues with Gradient Descent
Stochastic Gradient Descent

Machine Learning as Optimization Problem[1]

- Learning is optimization
- Faster optimization methods for faster learning
- Let $w \in \mathbb{R}^d$ and $f_0(w), f_1(w), \ldots, f_m(w)$ be real-valued functions.
- Standard optimization formulation is:

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & f_0(w) \\ \text{subject to} \quad & f_i(w) \leq 0, \ i = 1, \ldots, m. \end{aligned}$$

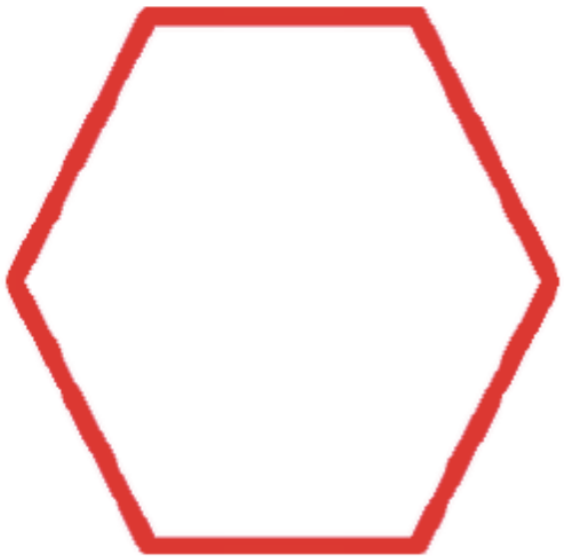
[1] Adapted from `http://ttic.uchicago.edu/~gregory/courses/ml2012/lectures/tutorial_optimization.pdf`. Also see, `http://www.stanford.edu/~boyd/cvxbook/` and `http://scipy-lectures.github.io/advanced/mathematical_optimization/`.

Solving Optimization Problems

- Methods for **general optimization problems**
  - Simulated annealing, genetic algorithms
- Exploiting *structure* in the optimization problem
  - **Convexity**, Lipschitz continuity, smoothness

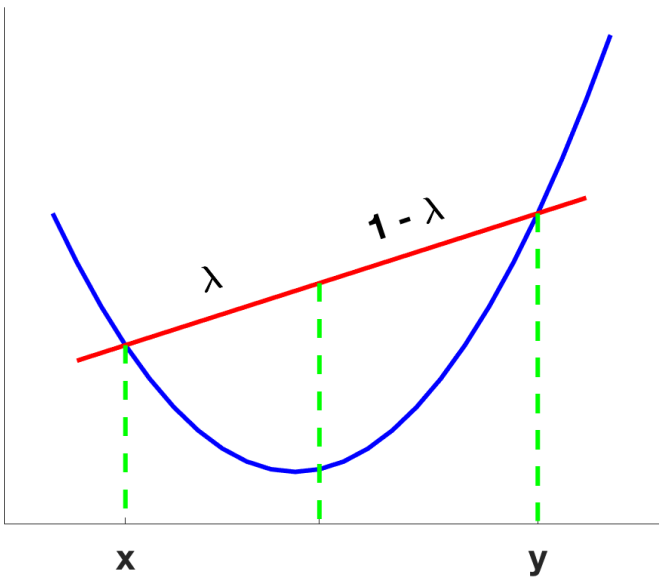
Convexity

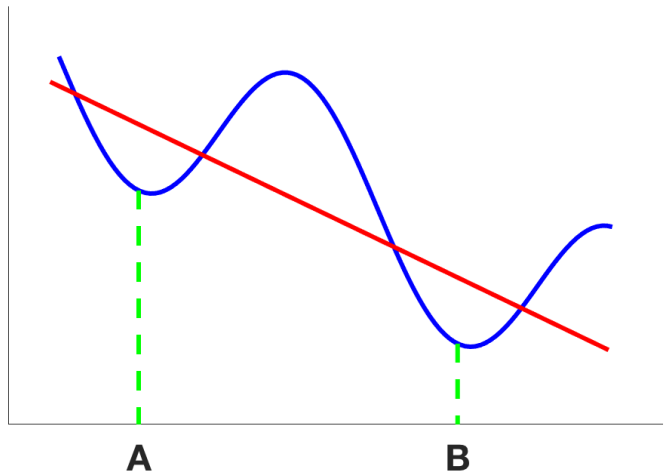
- Convex Sets



Convex Function

- Convex function





Convex Optimization

- Optimality Criterion

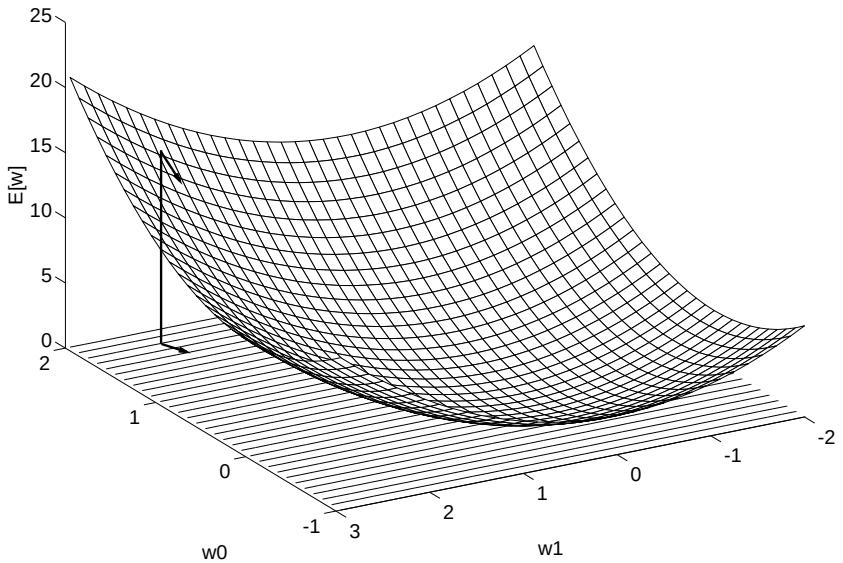
$$\begin{aligned} \underset{w}{\text{minimize}} \quad & f_0(w) \\ \text{subject to} \quad & f_i(w) \leq 0,\ i = 1, \ldots, m. \end{aligned}$$

  where all $f_i(w)$ are **convex functions**.
- w_0 is feasible if $w_0 \in Dom\ f_0$ and all constraints are satisfied
- A feasible w^* is optimal if $f_0(w^*) \leq f_0(w)$ for all w satisfying the constraints

Gradient of a Function

- Denotes the direction of steepest ascent

$$\nabla E(\mathbf{w}) = \begin{bmatrix} \frac{\partial E}{\partial w_0} \\ \frac{\partial E}{\partial w_1} \\ \vdots \\ \frac{\partial E}{\partial w_d} \end{bmatrix}$$



Finding Extremes of a Single Variable Function

- Set derivative to 0
- Second derivative for minima or maxima

Finding Extremes of a Multiple Variable Function - Gradient Descent

1. Start from any point in variable space
2. Move along the direction of the steepest descent (or ascent)
   - By how much?
   - A learning rate (η)
   - What is the direction of steepest descent?
     - Gradient of E at $\mathbf{w}$

Training Rule for Gradient Descent

$$\mathbf{w} = \mathbf{w} - \eta \nabla E(\mathbf{w})$$

For each weight component:

$$w_j = w_j - \eta \frac{\partial E}{\partial w_j}$$

Convergence Guaranteed?

- Error surface contains only one global minimum
- Algorithm *will* converge
  - Examples need not be linearly separable
- η should be *small enough*
- Impact of too large η?
- Too small η?

Issues with Gradient Descent

- Slow convergence
- Stuck in local minima

Stochastic Gradient Descent [1]

- **Update weights after every training example or a batch of examples.**
- For sufficiently small η, closely approximates Gradient Descent.

Gradient Descent	**Stochastic Gradient Descent**
Weights updated after summing error over all examples	Weights updated after examining a batch of examples
More computations per weight update step	Significantly lesser computations
Risk of local minima	Reduce the risk of local minima but can't avoid it

References


Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel.
Backpropagation applied to handwritten zip code recognition.
Neural Comput., 1(4):541–551, Dec. 1989.