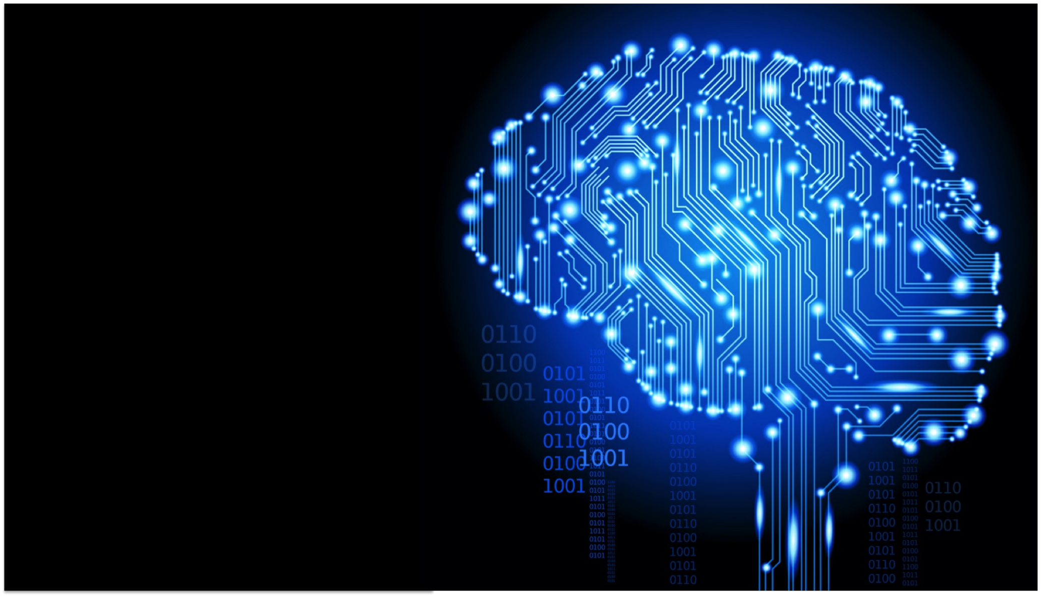
Convex Optimization for Machine Learning



Sunghee Yun (sunyun@amazon.com)

Mobile Shopping Team
Amazon

Today

- Convex optimization
  - why convex optimization?
  - general optimization
  - machine learning as an optimization
- Machine learning
  - statistics perspective
  - computer science perspective
  - numerical algorithms perspectives

Prerequisite for the talk

This talk will assume the audience

- has been exposed to basic linear algebra
- can distinguish componentwise inequality from that for positive semidefiniteness, *i.e.*,

$$Ax \preceq b \Leftrightarrow \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix} x \preceq \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \Leftrightarrow a_i^T x \leq b_i \text{ for } i = 1, \ldots, m,$$

but,

$$A \succeq 0 \Leftrightarrow A = A^T \text{ and } x^T A x \geq 0 \text{ for all } x \in \mathbf{R}^n$$
$$A \succ 0 \Leftrightarrow A = A^T \text{ and } x^T A x > 0 \text{ for all nonzero } x \in \mathbf{R}^n$$

Why convex optimization?

- many machine learning algorithms (inherently) depend on convex optimization
- one of few optization class that can be actually solved
- a number of engineering and scientific problems can be cast into convex optimization problems
- many more can be approximated to convex optimization
- convex optimization sheds lights on intrinsic property and structure of many optimization, hence, machine learning algorithms

Why convex optimization?

- many machine learning algorithms (inherently) depend on convex optimization
- one of few optization class that can be actually solved
- a number of engineering and scientific problems can be cast into convex optimization problems
- many more can be approximated to convex optimization
- convex optimization sheds lights on intrinsic property and structure of many optimization, hence, machine learning algorithms

Why convex optimization?

- many machine learning algorithms (inherently) depend on convex optimization
- one of few optization class that can be actually solved
- a number of engineering and scientific problems can be cast into convex optimization problems
- many more can be approximated to convex optimization
- convex optimization sheds lights on intrinsic property and structure of many optimization, hence, machine learning algorithms

Why convex optimization?

- many machine learning algorithms (inherently) depend on convex optimization
- one of few optization class that can be actually solved
- a number of engineering and scientific problems can be cast into convex optimization problems
- many more can be approximated to convex optimization
- convex optimization sheds lights on intrinsic property and structure of many optimization, hence, machine learning algorithms

Why convex optimization?

- many machine learning algorithms (inherently) depend on convex optimization
- one of few optization class that can be actually solved
- a number of engineering and scientific problems can be cast into convex optimization problems
- many more can be approximated to convex optimization
- convex optimization sheds lights on intrinsic property and structure of many optimization, hence, machine learning algorithms

Why convex optimization?

- many machine learning algorithms (inherently) depend on convex optimization
- one of few optization class that can be actually solved
- a number of engineering and scientific problems can be cast into convex optimization problems
- many more can be approximated to convex optimization
- convex optimization sheds lights on intrinsic property and structure of many optimization, hence, machine learning algorithms

Mathematical optimization

- mathematical optimization problem:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \;\; i = 1, \ldots, m \\ & h_i(x) = 0, \;\; i = 1, \ldots, p \end{array}$$

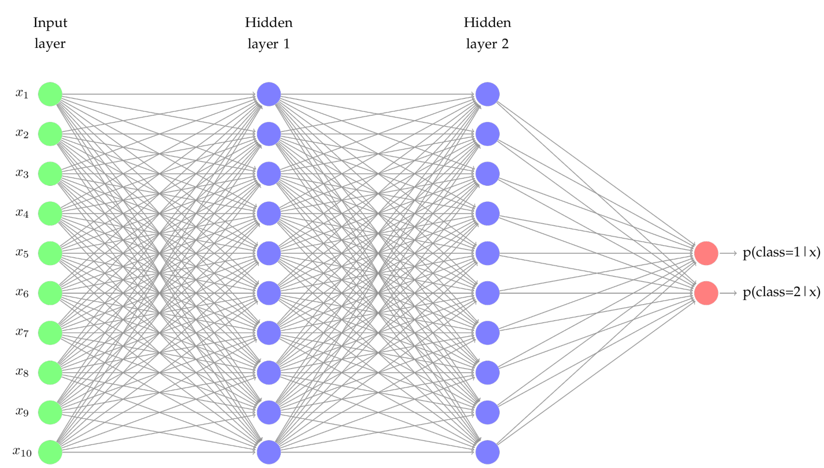
  - $x = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}^T \in \mathbf{R}^n$ is the (vector) optimization variable
  - $f_0 : \mathbf{R}^n \to \mathbf{R}$ is the objective function
  - $f_i : \mathbf{R}^n \to \mathbf{R}$ are the inequality constraint functions
  - $h_i : \mathbf{R}^n \to \mathbf{R}$ are the equality constraint functions

Optimization examples

- circuit optimization
  - optimization variables: transistor widths, resistances, capacitances, inductances
  - objective: operating speed (or equivalently, maximum delay)
  - constraints: area, power consumption
- portfolio optimization
  - optimization variables: amounts invested in different assets
  - objective: expected return
  - constraints: budget, overall risk, return variance

Optimization examples

- machine learning
  - optimization variables: model parameters ($e.g.$, connection weights)
  - objective: squared error (or loss function)
  - constraints: network architecture



Solution methods

- for general optimization problems
  - extremly difficult to solve (practically impossible to solve)
  - most methods try to find (good) suboptimal solutions, $e.g.$, using heuristics
- some exceptions
  - least-squares (LS)
  - liner programming (LP)
  - semidefinite programming (SDP)

Least-squares (LS)

- least-squares (LS) problem:

$$\text{minimize} \quad \|Ax - b\|_2^2 = \sum_{i=1}^m (a_i^T x - b_i)^2$$

  - analytic solution: any solution satisfying $(A^T A)x^* = A^T b$
  - extremely reliable and efficient algorithms
  - has been there at least since Gauss
- applications
  - LS problems are easy to recognize
  - has huge number of applications, $e.g.$, line fitting

Linear programming (LP)

- linear program (LP):

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b \end{array}$$

  - no analytic solution
  - reliable and efficient algorithms exist, $e.g.$, simplex method, interiorpoint method
  - has been there at least since Fourier
  - systematical algorithm existed since World War II
- applications
  - less obvious to recognize (than LS)
  - lots of problems can be cast into LP, $e.g.$, network flow problem

Semidefinite programming (SDP)

- semidefinite program (SDP):

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & F_0 + x_1 F_1 + \cdots + x_n F_n \succeq 0 \end{array}$$

  - no analytic solution
  - but, reliable and efficient algorithms exist, *e.g.*, interior-point method
  - recent technology
- applications
  - never easy to recognize
  - lots of problems, *e.g.*, optimal control theory, can be cast into SDP
  - extremely non-obvious, but convex, hence global optimality easily achieved!

Max-det problem (extension of SDP)

- max-det program:

$$\begin{array}{ll} \text{minimize} & c^T x + \log\det(F_0 + x_1 F_1 + \cdots + x_n F_n) \\ \text{subject to} & G_0 + x_1 G_1 + \cdots + x_n G_n \succeq 0 \end{array}$$

  - no analytic solution
  - but, reliable and efficient algorithms exist, $e.g.$, interior-point method
  - recent technology
- applications
  - never easy to recognize
  - lots of stochastic optimization problems, $e.g.$, every covariance matrix is positive semidefinite
  - again convex, hence global optimality (relatively) easily achieved!

Common features in these Exceptions?

- they are convex optimization problems!
- convex optimization:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \preceq_{K_i} 0, \;\; i = 1, \ldots, m \\ & Ax = b \end{array}$$

  where

  - $f_0(\lambda x + (1-\lambda)y) \leq \lambda f_0(x) + (1-\lambda) f_0(y)$ for all $x, y \in \mathbf{R}^n$ and $0 \leq \lambda \leq 1$
  - $f_i : \mathbf{R}^n \to \mathbf{R}^{k_i}$ are K_i-convex w.r.t. proper cone $K_i \subseteq \mathbf{R}^{k_i}$
  - all equality constraints are linear

Convex optimization

- algorithms
  - classical algorithms like simplex method still work well for many LPs
  - many state-of-the-art algorithms develoled for (even) large-scale convex optimization problems
    - barrier methods
    - primal-dual interior-point methods
- applications
  - huge number of engineering and scientific problems are (or can be cast into) convex optimization problems
  - convex relaxation

What's fuss about convex optimization?

- which one of these problems are easier to solve?
  - (generalized) geometric program with $n = 3,000$ variables and $m = 1,000$ constraints

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{p_0} \alpha_{0,i} x_1^{\beta_{0,i,1}} \cdots x_n^{\beta_{0,i,n}} \\ \text{subject to} & \sum_{i=1}^{p_j} \alpha_{j,i} x_1^{\beta_{j,i,1}} \cdots x_n^{\beta_{j,i,n}} \leq 1, \; j = 1, \ldots, m \end{array}$$

    with $\alpha_{j,i} \geq 0$ and $\beta_{j,i,k} \in \mathbf{R}$
    $\Rightarrow$ can be solved within 1 minute *globally* in your laptop computer
  - minimization of 10th order polynomial of $n = 20$ variables with no constraint

$$\text{minimize} \quad \sum_{i_1=1}^{10} \cdots \sum_{i_n=1}^{10} c_{i_1,\ldots,i_n} x_1^{i_1} \cdots x_n^{i_n}$$

    with $c_{i_1,\ldots,i_n} \in \mathbf{R}$
    $\Rightarrow$ you *cannot* solve!

What's fuss about convex optimization?

- which one of these problems are easier to solve?
  - (generalized) geometric program with $n = 3,000$ variables and $m = 1,000$ constraints

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{p_0} \alpha_{0,i} x_1^{\beta_{0,i,1}} \cdots x_n^{\beta_{0,i,n}} \\ \text{subject to} & \sum_{i=1}^{p_j} \alpha_{j,i} x_1^{\beta_{j,i,1}} \cdots x_n^{\beta_{j,i,n}} \leq 1, \; j = 1, \ldots, m \end{array}$$

    with $\alpha_{j,i} \geq 0$ and $\beta_{j,i,k} \in \mathbf{R}$
    $\Rightarrow$ can be solved within 1 minute *globally* in your laptop computer
  - minimization of 10th order polynomial of $n = 20$ variables with no constraint

$$\text{minimize} \quad \sum_{i_1=1}^{10} \cdots \sum_{i_n=1}^{10} c_{i_1,\ldots,i_n} x_1^{i_1} \cdots x_n^{i_n}$$

    with $c_{i_1,\ldots,i_n} \in \mathbf{R}$
    $\Rightarrow$ you *cannot* solve!

What's fuss about convex optimization?

- which one of these problems are easier to solve?
  - (generalized) geometric program with $n = 3,000$ variables and $m = 1,000$ constraints

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{p_0} \alpha_{0,i} x_1^{\beta_{0,i,1}} \cdots x_n^{\beta_{0,i,n}} \\ \text{subject to} & \sum_{i=1}^{p_j} \alpha_{j,i} x_1^{\beta_{j,i,1}} \cdots x_n^{\beta_{j,i,n}} \leq 1, \ j = 1, \ldots, m \end{array}$$

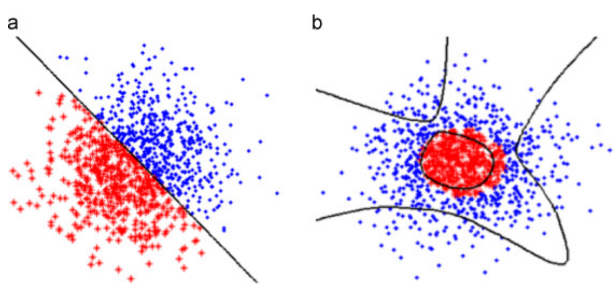
    with $\alpha_{j,i} \geq 0$ and $\beta_{j,i,k} \in \mathbf{R}$
    $\Rightarrow$ can be solved within 1 minute *globally* in your laptop computer
  - minimization of 10th order polynomial of $n = 20$ variables with no constraint

$$\text{minimize} \quad \sum_{i_1=1}^{10} \cdots \sum_{i_n=1}^{10} c_{i_1,\ldots,i_n} x_1^{i_1} \cdots x_n^{i_n}$$

    with $c_{i_1,\ldots,i_n} \in \mathbf{R}$
    $\Rightarrow$ you *cannot* solve!

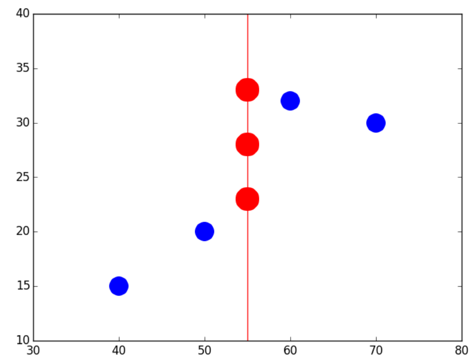
What is machine learning?

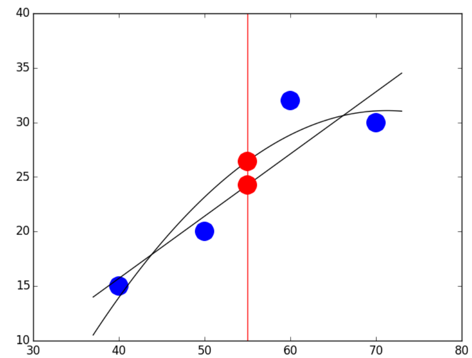
- machine learning
  - is the subfield of computer science that "gives computers the ability to learn without being explicitly programmed." (Arthur Samuel, 1959)
  - learns from data and predicts on data
- applications
  - spam fitering, search engine
  - detection of network intruders (or malicious insiders)
  - computer vision, speach recognition, natural language processing



ML example: regression

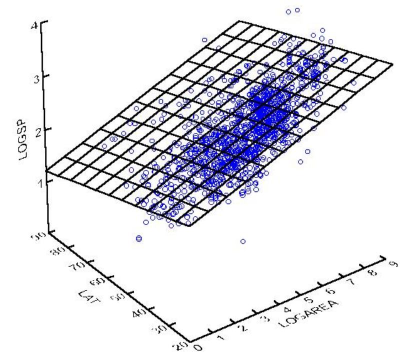
- problem: what is a reasonable price for a house?
  - what would a rational (or rather normal) human being do?
  - ML approach:
    - $*$ collect data: x: size, y: price
    - $*$ train model: draw a line to represent (typical) trend
    - $*$ predict a price from the line



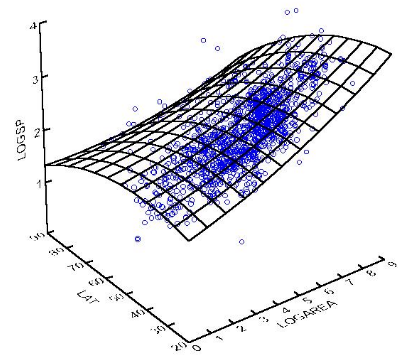


ML example: multi-variate regression

- what if we have more than one x? or rather more than two x's?



- what if highly nonlinera and nonconvex fitting function is needed?



Mathematical formulation for (supervised) ML

- given training set, $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, where $x^{(i)} \in \mathbf{R}^p$ and $y^{(i)} \in \mathbf{R}^q$
- want to find function $g_\theta : \mathbf{R}^p \to \mathbf{R}^q$ with learning parameter, $\theta \in \mathbf{R}^n$
  - $g_\theta(x)$ desired to be as close as possible to y for future $(x, y) \in \mathbf{R}^p \times \mathbf{R}^q$
  - *i.e.*, $g_\theta(x) \sim y$
- define a loss function $l : \mathbf{R}^q \times \mathbf{R}^q \to \mathbf{R}_+$
- solve the optimization problem:

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m} \sum_{i=1}^m l(g_\theta(x^{(i)}), y^{(i)}) \\ \text{subject to} & \theta \in \Theta \end{array}$$

Linear regression

- (simple) linear regression is a ML method when
  - $q = 1$, $i.e.$, the output is scalar
  - $g_\theta(x) = \theta^T \begin{bmatrix} 1 \\ x \end{bmatrix} = \theta_0 + \theta_1 x_1 + \cdots + \theta_p x_p$, $i.e.$, $n = p + 1$
  - $l : \mathbf{R} \times \mathbf{R} \to \mathbf{R}_+$ is defined by $l(y_1, y_2) = (y_1 - y_2)^2$
  - $\Theta = \mathbf{R}^{p+1}$, $i.e.$, parameter domain is all the real numbers
- formulation

$$\text{minimize} \quad f(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)} \right)^2$$

Solution method for linear regression

- linear regression is nothing but LS since

$$
\begin{aligned}
mf(\theta) &= \sum_{i=1}^{m} \left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)} \right)^2 = \left\| \begin{bmatrix} 1 & {x^{(1)}}^T \\ \vdots & \vdots \\ 1 & {x^{(m)}}^T \end{bmatrix} \theta - \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \right\|_2^2 \\
&= \|X\theta - y\|_2^2
\end{aligned}
$$

- convex in θ, hence obtains its global optimality when the gradient vanishes, $i.e.$,

$$
m\nabla f(\theta) = 2X^T(X\theta - y) = 2((X^TX)\theta - X^Ty) = 0
$$

- analytic solution exists and in practice,
  - QR decomposition or single value decomposition (SVD) can be used

Multiple output linear regression

- multiple output linear regression is a ML method when
  - $g_\theta(x) = \theta^T \begin{bmatrix} 1 \\ x \end{bmatrix} = \begin{bmatrix} \theta_{1,0} + \theta_{1,1} x_1 + \cdots + \theta_{1,p} x_p \\ \vdots \\ \theta_{q,0} + \theta_{q,1} x_1 + \cdots + \theta_{q,p} x_p \end{bmatrix}$
  - $l : \mathbf{R}^q \times \mathbf{R}^q \to \mathbf{R}_+$ is defined by $l(y_1, y_2) = \|y_1 - y_2\|_2^2$
  - $\Theta = \mathbf{R}^{(p+1)\times q}$, *i.e.*, parameter domain is all the real numbers
- formulation

$$\text{minimize} \quad f(\theta) = \frac{1}{m} \sum_{i=1}^m \left\| \theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)} \right\|_2^2$$

Solution method for multiple output linear regression

- linear regression is nothing but LS since

$$
\begin{aligned}
mf(\theta) &= \sum_{i=1}^{m} \left\| \theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)} \right\|_2^2 \\
&= \left\| \begin{bmatrix} 1 & {x^{(1)}}^T & \cdots & 1 & {x^{(1)}}^T \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & {x^{(m)}}^T & \cdots & 1 & {x^{(m)}}^T \end{bmatrix} \tilde{\theta} - \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \right\|_2^2 \\
&= \|\tilde{X}\tilde{\theta} - y\|_2^2
\end{aligned}
$$

where $\tilde{X} \in \mathbf{R}^{m \times q(p+1)}$ and $\tilde{\theta} \in \mathbf{R}^{q(p+1)}$

- hence, the same method applies

Linear regression with constraints

- what if we have one constraint?

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)}\right)^2 \\ \text{subject to} & \theta_1 \geq 0 \end{array}$$

- no analytic solution exists (with only one constraint) in general
- however, convex optimization algorithms solve it (almost) as easily as original problem
- but, now with *any* number of convex constraints

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)}\right)^2 \\ \text{subject to} & g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, l \\ & A\theta = b \end{array}$$

Linear regression with constraints

- what if we have one constraint?

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)}\right)^2 \\ \text{subject to} & \theta_1 \geq 0 \end{array}$$

- no analytic solution exists (with only one constraint) in general
- however, convex optimization algorithms solve it (almost) as easily as original problem
- but, now with *any* number of convex constraints

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)}\right)^2 \\ \text{subject to} & g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, l \\ & A\theta = b \end{array}$$

Linear regression with constraints

- what if we have one constraint?

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)}\right)^2 \\ \text{subject to} & \theta_1 \geq 0 \end{array}$$

- no analytic solution exists (with only one constraint) in general
- however, convex optimization algorithms solve it (almost) as easily as original problem
- but, now with *any* number of convex constraints

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \begin{bmatrix} 1 \\ x^{(i)} \end{bmatrix} - y^{(i)}\right)^2 \\ \text{subject to} & g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, l \\ & A\theta = b \end{array}$$

Linear regression with constraints

- what if we have one constraint?

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \left[\begin{array}{c} 1 \\ x^{(i)} \end{array}\right] - y^{(i)}\right)^2 \\ \text{subject to} & \theta_1 \geq 0 \end{array}$$

- no analytic solution exists (with only one constraint) in general
- however, convex optimization algorithms solve it (almost) as easily as original problem
- but, now with *any* number of convex constraints

$$\begin{array}{ll} \text{minimize} & f(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta^T \left[\begin{array}{c} 1 \\ x^{(i)} \end{array}\right] - y^{(i)}\right)^2 \\ \text{subject to} & g_i(\theta) \leq 0 \text{ for } i = 1, \ldots, l \\ & A\theta = b \end{array}$$

Support vector machine

- problem definition:
  - given $x^{(i)} \in \mathbf{R}^p$: input data, and $y^{(i)} \in \{-1, 1\}$: output labels
  - find hyperplane which separates two different classes as distinctively as possible (in some measure)
- (typical) formulation:

$$\begin{array}{ll} \text{minimize} & \|a\|_2^2 + \gamma \sum_{i=1}^m u_i \\ \text{subject to} & y^{(i)}(a^T x^{(i)} + b) \geq 1 - u_i, \; i = 1, \ldots, m \\ & u \succeq 0 \end{array}$$

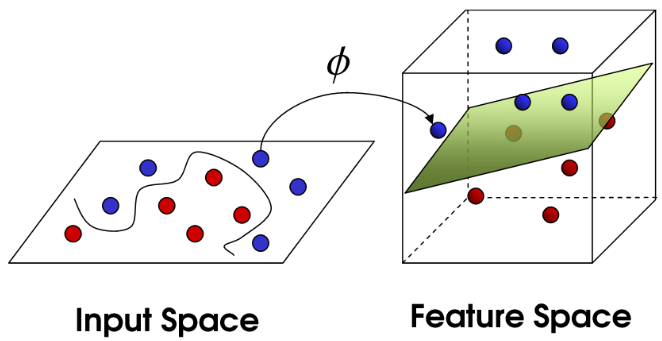
  - convex optimization problem, hence stable and efficient algorithms exist even for very large problems
  - has worked extremely well in practice (until... deep learning boom)

Support vector machine with kernels

- use feature transformation $\phi : \mathbf{R}^p \to \mathbf{R}^q$ (with $q > p$)
- formulation:

$$\begin{array}{ll} \text{minimize} & \|\tilde{a}\|_2^2 + \gamma \sum_{i=1}^m \tilde{u}_i \\ \text{subject to} & y^{(i)}(\tilde{a}^T \phi(x^{(i)}) + \tilde{b}) \geq 1 - \tilde{u}_i, \;\; i = 1, \ldots, m \\ & \tilde{u} \succeq 0 \end{array}$$

- still convex optimization problem



Different perspectives on machine learning

- statistical view
- computer scientific perspective
- numerical algorithmic perspective
- performance acceleration using hardware parallelism with GPGPUs

Statistical perspective

- suppose data set $X_m = \{x^{(1)}, \ldots, x^{(m)}\}$
  - drawn independently from (true, but unknown) data generating distribution $p_{\text{data}}(x)$
- Maximum Likelihood Estimation (MLE) is to solve

$$\text{maximize} \quad p_{\text{data}}(X;\theta) = \prod_{i=1}^{m} p_{\text{data}}(x^{(i)};\theta)$$

- equivalent, but numerically friendly formulation:

$$\text{maximize} \quad \log p_{\text{data}}(X;\theta) = \sum_{i=1}^{m} \log p_{\text{data}}(x^{(i)};\theta)$$

Equivalence of MLE to KL divergence

- in information theory, Kullback-Leibler (KL) divergence defines distance between two probability distributions, p and q:

$$D_{\mathrm{KL}}(p\|q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)}\, dx$$

- KL divergence between data distribution, p_{data}, and model distribution, p_{model}, can be approximated by Monte Carlo method as

$$D_{\mathrm{KL}}(p_{\mathrm{data}}\|p_{\mathrm{model}}) \simeq \frac{1}{m}\sum_{i=1}^{m}(\log p_{\mathrm{data}}(x^{(i)}) - \log p_{\mathrm{model}}(x^{(i)};\theta))$$

- hence, *minimizing the KL divergence is equivalent to maximizing the log-likelihood!*

Equivalence of MLE to MSE

- assume the model is Gaussian, *i.e.*, $y \sim \mathcal{N}(g_\theta(x), \Sigma)$:

$$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sqrt{2\pi}^p |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}\left(y^{(i)} - g_\theta(x^{(i)})\right)^T \Sigma^{-1} \left(y^{(i)} - g_\theta(x^{(i)})\right)\right)$$

- assuming that $\Sigma = I_p$, the log-likelihood becomes

$$\sum_{i=1}^{m} \log p(y^{(i)}|x^{(i)};\theta) = -\sum_{i=1}^{m} \|y^{(i)} - g_\theta(x^{(i)})\|_2^2/2 - \frac{pm}{2}\log(2\pi)$$

- hence, *maximizing log-likelihood is equivalent to minimizing mean-square-error (MSE)!*

Other statistical factors

- overfitting problems
- training and test
- cross-validation
- regularization
- drop-out

Computer scientific perspectives

- neural network architectures
- hyper parameter optimization
- double/single precision representation
- low-power machine learning (especially for inference)

Numerical algorithmic perspectives

- basic formulation:

$$\text{minimize} \quad f(\theta) = \frac{1}{m}\sum_{i=1}^{m} l(g_\theta(x^{(i)}), y^{(i)})$$

- formulation with regularization:

$$\text{minimize} \quad f(\theta) = \frac{1}{m}\sum_{i=1}^{m} l(g_\theta(x^{(i)}), y^{(i)}) + \gamma r(\theta)$$

- stochastic gradient descent (SGD):

$$\theta^{(k+1)} = \theta^{(k)} - \alpha_k \nabla f(\theta)$$

- some other momentum and adaptive methods: Nesterov's accelerated gradient method, AdaGrad, RMSProp, Adam, *etc.*

Ridge regression

- Ridge regression solves the following problem: (for some $\lambda > 0$)

$$\text{minimize} \quad f_0(x) = \|Ax - y\|_2^2 + \lambda\|x\|_2^2$$

  – regularization, $e.g.$, to preventing overfitting

- can be extended to (without sacraficing solvability!)

$$\begin{array}{ll} \text{minimize} & f_0(x) = \|Ax - y\|_2^2 + \lambda\|x\|_2^2 = \left\| \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} x - \begin{bmatrix} y \\ 0 \end{bmatrix} \right\|_2^2 \\ \text{subject to} & f_i(x) \leq 0, \ i = 1, \ldots, m \\ & h_i(x) = 0, \ i = 1, \ldots, p \end{array}$$

- can be incorporated into gradient descent algorithm, $e.g.$,

$$\nabla f(x) = 2A^T(Ax - y) + 2\lambda x$$

Lasso (least absolute shrinkage & selection operator)

- Lasso solves (a problem equivalent to) the following problem:

$$\text{minimize} \quad f_0(x) = \|Ax - y\|^2 + \lambda \|x\|_1$$

  - 1-norm penalty term for parameter selection
  - similar to drop-out technique for regularization
- However, the objective funtion *not* smooth.
- simple trick would solve this problem

$$\begin{array}{ll} \text{minimize} & f_0(x) = \|Ax - y\|^2 + \lambda \sum_{i=1}^n z_i \\ \text{subject to} & -z_i \leq x_i \leq z_i, \ i = 1, \ldots, n \\ & f_i(x) \leq 0, \ i = 1, \ldots, m \\ & h_i(x) = 0, \ i = 1, \ldots, p \end{array}$$

- Can't we use these kinds of tricks for our ML problems?

Duality

- every (constrained) optimization problem has a *dual problem* (whether or not it's a convex optimization problem)
- every dual problem is a *convex optimization problem* (whether or not it's a convex optimization problem)
- duality provides *optimality certificate*, hence plays *central role* for modern optimization and machine learning algorithm implementation
- (usually) solving one readily solves the other!

Lagrangian

- standard form problem:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \; i = 1, \dots, m \\ & h_i(x) = 0, \; i = 1, \dots, p \end{array}$$

where $x \in \mathbf{R}^n$ is optimization variable, $\mathcal{D}$ is domain, p^* is optimal value

- Lagrangian: $L : \mathbf{R}^n \times \mathbf{R}^m \times \mathbf{R}^p \to \mathbf{R}$ with $\mathbf{dom}\, L = \mathcal{D} \times \mathbf{R}^m \times \mathbf{R}^p$ defined by

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x)$$

  - λ_i: Lagrange multiplier associated with $f_i(x) \leq 0$
  - ν_i: Lagrange multiplier associated with $h_i(x) = 0$

Lagrange dual function

- Lagrange dual function: $g : \mathbf{R}^m \times \mathbf{R}^p \to \mathbf{R}$ defined by

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = \inf_{x \in \mathcal{D}} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \right)$$

  - g is *always* concave
  - $g(\lambda, \nu)$ can be $-\infty$

- lower bound property: if $\lambda \succeq 0$, then $g(\lambda, \nu) \leq p^*$

Dual problem

- Lagrange dual problem:

$$\begin{array}{ll} \text{maximize} & g(\lambda, \nu) \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

  - is a convex optimization problem
  - provides a lower bound on p^*

- let d^* denote the optimal value for the dual problem
  - week duality: $d^* \leq p^*$
  - strong duality: $d^* = p^*$

Dual problem provides optimality certificate!

- (almost) all algorithms solves the dual problem simultaneously
- Lagrangian dual variables obtained with no additional cost
- if iterative algorithm generates solution sequence,

$$(x^{(1)}, \lambda^{(1)}, \nu^{(1)}) \to (x^{(2)}, \lambda^{(2)}, \nu^{(2)}) \to (x^{(3)}, \lambda^{(3)}, \nu^{(3)}) \to \cdots$$

then, we have an optimality certificate:

$$f(x^{(k)}) - p^* \leq f(x^{(k)}) - g(\lambda^{(k)}, \nu^{(k)})$$

In summary

- convex optimization problems are one of few optimization problems that can actually be solved

- many ML problems can be cast into convex optimizations

- convex optimization could inspire new methods for MLs

Thank you!

Sunghee Yun (sunyun@amazon.com)

Weak duality

- weak duality implies $d^* \leq p^*$
  - always true (by construction of dual problem)
  - provides *nontrivial* lower bounds, especially, for difficult problems, $e.g.$, solving the following SDP:

$$\begin{array}{ll} \text{maximize} & -\mathbf{1}^T \nu \\ \text{subject to} & W + \mathbf{diag}(\nu) \succeq 0 \end{array}$$

  gives a lower bound for max-cut problem

$$\begin{array}{ll} \text{minimize} & x^T W x \\ \text{subject to} & x_i^2 = 1, \ i = 1, \ldots, n \end{array}$$

Strong duality

- strong duality implies $d^* = p^*$
  - not necessarily hold; does not hold in general
  - *usually* holds for convex optimization problems
  - conditions which guarantee strong duality in convex problems called *constraint qualifications*

Duality example: LP

- primal problem:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b \end{array}$$

- dual function:

$$g(\lambda) = \inf_x \left(\left(c + A^T\lambda \right)^T x - b^T\lambda \right) = \begin{cases} -b^T\lambda & \text{if } A^T\lambda + c = 0 \\ -\infty & \text{otherwise} \end{cases}$$

- dual problem:

$$\begin{array}{ll} \text{maximize} & -b^T\lambda \\ \text{subject to} & A^T\lambda + c = 0 \\ & \lambda \succeq 0 \end{array}$$

  - Slater's condition implies that $p^* = d^*$ if $A\tilde{x} \prec b$ for some $\tilde{x}$
  - truth is, $p^* = d^*$ except when both primal and dual are infeasible

Duality example: QP

- primal problem (assuming $P \in \mathbf{S}^n_{++}$):

$$\begin{array}{ll} \text{minimize} & x^T P x \\ \text{subject to} & Ax \preceq b \end{array}$$

- dual function:

$$g(\lambda) = \inf_x \left(x^T P x + \lambda^T (Ax - b) \right) = -\frac{1}{4} \lambda^T A P^{-1} A^T \lambda - b^T \lambda$$

- dual problem:

$$\begin{array}{ll} \text{maximize} & -\lambda^T A P^{-1} A^T \lambda / 4 - b^T \lambda \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

  - Slater's condition implies that $p^* = d^*$ if $A\tilde{x} \prec b$ for some $\tilde{x}$
  - truth is, $p^* = d^*$ always!

Complementary slackness

- assume strong dualtiy holds, x^* is primal optimal, and (λ^*, ν^*) is dual optimal

$$\begin{aligned} f_0(x^*) &= g(\lambda^*, \nu^*) = \inf_x \left(f_0(x) + \sum_{i=1}^m \lambda_i^* f_i(x) + \sum_{i=1}^p \nu_i^* h_i(x) \right) \\ &\le f_0(x^*) + \sum_{i=1}^m \lambda_i^* f_i(x^*) + \sum_{i=1}^p \nu_i^* h_i(x^*) \\ &\le f_0(x^*) \end{aligned}$$

- thus, all inequalities are tight, $i.e.$, they hold with equalities
  - x^* minimizes $L(x, \lambda^*, \nu^*)$
  - $\lambda_i^* f_i(x^*) = 0$ for all i, known as *complementary slackness*

$$\lambda_i^* > 0 \Rightarrow f_i(x^*) = 0, \qquad f_i(x^*) < 0 \Rightarrow \lambda_i^* = 0$$

Karush-Kuhn-Tucker (KKT) conditions

- KKT (optimality) conditions consist of
  - primal feasibility: $f_i(x) \leq 0$ for all $1 \leq i \leq m$, $h_i(x) = 0$ for all $1 \leq i \leq p$
  - dual feasibility: $\lambda \succeq 0$
  - complementary slackness: $\lambda_i f_i(x) = 0$
  - zero gradient of Lagrangian: $\nabla f_0(x) + \sum_{i=1}^m \lambda_i \nabla f_i(x) + \sum_{i=1}^p \nu_i \nabla h_i(x) = 0$

- if strong daulity holds and x^*, λ^*, and ν^* are optimal, they satisfy KKT condtions!

KKT conditions for convex optimization problem

- if $\tilde{x}$, $\tilde{\lambda}$, and $\tilde{\nu}$ satisfy KKT for convex optimization problem, then they are optimal!
  - complementary slackness implies $f_0(\tilde{x}) = L(\tilde{x}, \tilde{\lambda}, \tilde{\nu})$
  - last conidtion together with convexity implies $g(\tilde{\lambda}, \tilde{\nu}) = L(\tilde{x}, \tilde{\lambda}, \tilde{\nu})$
- thus, for example, if Slater's condition is satisfied, x is optimal if and only if there exist λ, ν that satisfy KKT conditions
  - Slater's condition implies strong dualtiy, hence dual optimum is attained
  - this generalizes optimality condition $\nabla f_0(x) = 0$ for unconstrained problem