Non-Quadratic Regularizers

Jong-Han Kim

EE787 Fundamentals of machine learning
Kyung Hee University

Regularizers

Regularizers

- motivation:
  - large θ_i makes prediction $\theta^T x$ sensitive to value of x_i
  - so we want θ (or $\theta_{2:d}$ if $x_1 = 1$) small
- regularizer $r : \mathbf{R}^d \to \mathbf{R}$ measures the size of θ
- usually regularizer is *separable*,

$$r(\theta) = q(\theta_1) + \cdots + q(\theta_d)$$

where $q : \mathbf{R} \to \mathbf{R}$ is a penalty function for the predictor coefficients

Sum squares regularizer

- *sum squares* regularizer uses square penalty $q^{\mathsf{sqr}}(a) = a^2$

$$r(\theta) = ||\theta||^2 = \theta_1^2 + \cdots + \theta_d^2$$

- also called *quadratic*, *Tychonov*, or ℓ_2 regularizer

Sensitivity interpretation

- suppose the feature vector x changes to $\tilde{x} = x + \delta$
- δ is the *perturbation* or change in x
- the change in prediction is $|\theta^\mathsf{T}\tilde{x} - \theta^\mathsf{T}x| = |\theta^\mathsf{T}\delta|$
- how big can this be, if δ is small, *i.e.*, $||\delta|| \leq \epsilon$?
- by Cauchy-Schwarz inequality, $|\theta^\mathsf{T}\delta| \leq ||\theta|| ||\delta|| \leq \epsilon||\theta||$
- and the choice $\delta = \frac{\epsilon}{||\theta||}\theta$ achieves this maximum change in prediction
- so $||\theta||$ is a measure of the worst-case change in prediction when x is perturbed by δ, with $||\delta|| \leq 1$

ℓ_1 regularizer

- *sum absolute* or ℓ_1 regularizer uses absolute value penalty $q^{\text{abs}}(a) = |a|$

$$r(\theta) = ||\theta||_1 = |\theta_1| + \cdots + |\theta_d|$$

- $||\theta||_1$ is *ℓ_1 norm* of θ
- like the Euclidean or ℓ_2 norm $||\theta||$, it is a norm, *i.e.*, a measure of the size of the vector θ
- Euclidean norm is often written as $||\theta||_2$ to distinguish it from the ℓ_1 norm
- they are both members of the *p-norm family*, defined as

$$||\theta||_p = (|\theta_1|^p + \cdots + |\theta_d|^p)^{1/p}$$

for $p \geq 1$

Sensitivity interpretation

- suppose the feature vector x changes to $\tilde{x} = x + \delta$
- now we assume $|\delta_i| \leq \epsilon$, *i.e.*, *each feature can change by* $\pm\epsilon$
- how big can the change in prediction $|\theta^T \tilde{x} - \theta^T x| = |\theta^T \delta|$ be?
- the choice $\delta_i = \epsilon \, \mathrm{sign}(\theta_i)$ maximizes the change in prediction, *i.e.*,
  - $\delta_i = \epsilon$ if $\theta_i \geq 0$
  - $\delta_i = -\epsilon$ if $\theta_i < 0$
- with this choice the change in prediction is

$$\epsilon|\theta^T \mathrm{sign}(\theta)| = \epsilon(|\theta_1| + \cdots + |\theta_d|) = \epsilon||\theta||_1$$

- so $||\theta||_1$ is a measure of the worst-case change in prediction when x is perturbed entrywise by 1

Lasso regression

- use square loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$
- choosing θ to minimize $\mathcal{L}(\theta) + \lambda||\theta||_2^2$ is called *ridge regression*
- choosing θ to minimize $\mathcal{L}(\theta) + \lambda||\theta||_1$ is called *lasso regression*
- invented by (Stanford's) Rob Tibshirani, 1994
- widely used in advanced machine learning
- unlike ridge regression, there is no formula for the lasso parameter vector
- but we can efficiently compute it anyway (since it's convex)
- the lasso regression model has some interesting properties

Sparsifying regularizers

Sparse coefficient vector

- suppose θ is sparse, *i.e.*, many of its entries are zero
- prediction $\theta^\mathsf{T} x$ does not depend on features x_i for which $\theta_i = 0$
- this means we select *some* features to use (*i.e.*, those with $\theta_i \neq 0$)
- (possible) practical benefits of sparse θ:
  - can improve performance when many features are actually irrelevant
  - makes predictor *simpler to interpret*

Sparse coefficient vectors via ℓ_1 regularization

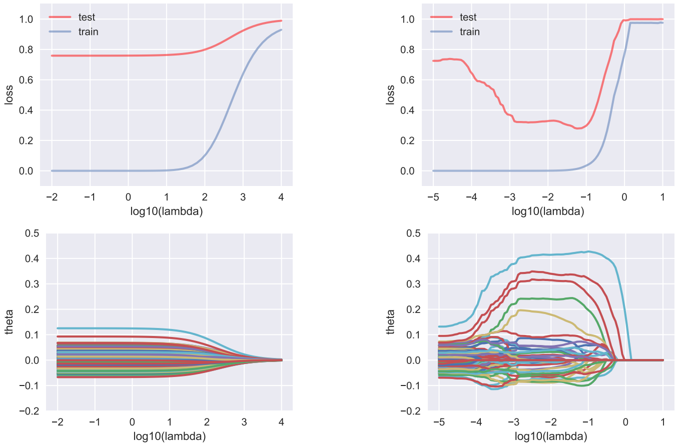
using ℓ_1 regularization leads to sparse coefficient vectors

$r(\theta) = ||\theta||_1$ is called a *sparsifying regularizer*

rough explanation:

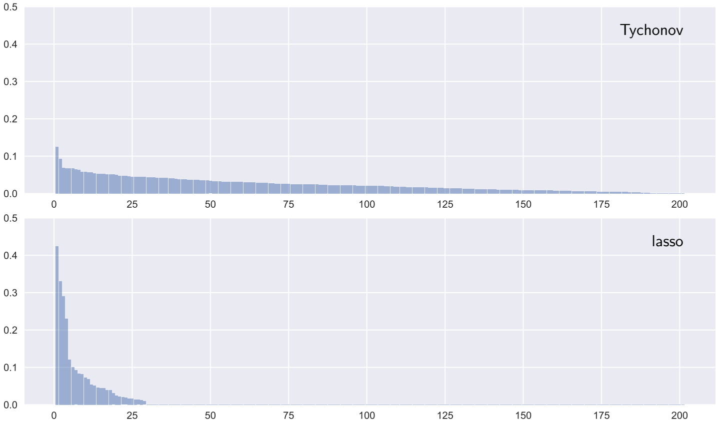
- for square penalty, once θ_i is small, θ_i^2 is very small
- so incentive for sum squares regularizer to make a coefficient smaller decreases once it is small
- for absolute penalty, incentive to make θ_i smaller keeps up all the way until it's zero

Example



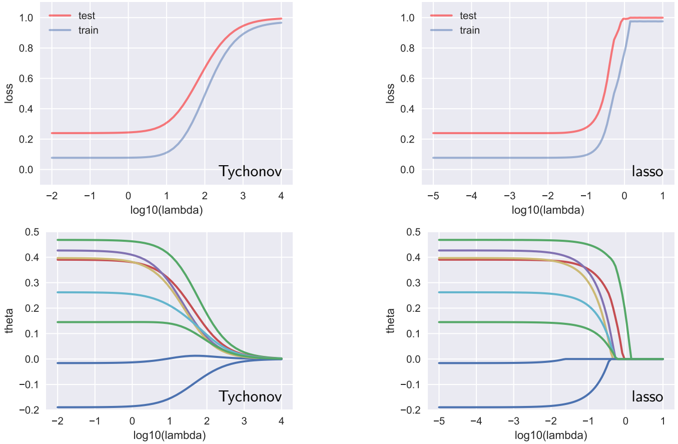
- artificially generated 50 data points, 200 features
- only a few features are relevant
- left hand plots use Tychonov, right hand use lasso

Example



- sorted $|\theta_i|$ at optimal λ
- lasso solution has only 35 non-zero components

Example



- choose λ based on regularization path with test data
- keep features corresponding to largest components of θ and *retrain*
- plots above use most important 7 features identified by lasso

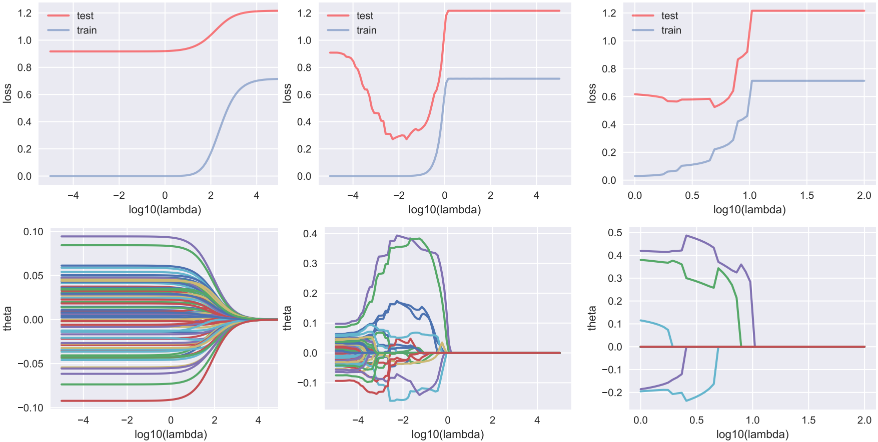
Even stronger sparsifiers

- $q(a) = |a|^{1/2}$
- called $\ell_{0.5}$ regularizer
- but you shouldn't use this term since

$$\left(|\theta_1|^{0.5} + \cdots + |\theta_d|^{0.5}\right)^2$$

  is not a norm (see VMLS)
- 'stronger' sparsifier than ℓ_1
- but not convex so computing θ is heuristic

Example



- ℓ_2, ℓ_1, and square root regularization

Nonnegative regularizer

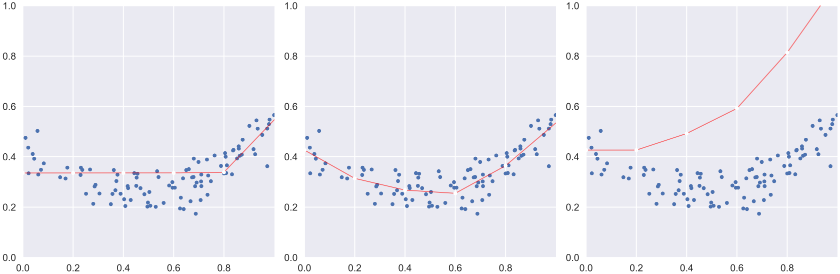
Nonegative coefficients

- in some cases we know or require that $\theta_i \geq 0$
- this means that when x_i increases, so must our prediction
- we can think of this constraint as regularization with penalty function

$$q(a) = \begin{cases} 0 & a \geq 0 \\ \infty & a < 0 \end{cases}$$

- example: y is lifespan, x_i measures healthy behavior i
- with quadratic loss, called *nonnegative least squares* (NNLS)
- common heuristic for nonnegative least squares: use $(\theta^{\mathsf{ls}})_+$ (works poorly)

Example



- feature vector $x = (1, u, (u - 0.2)_+, \ldots, (u - 0.8)_+)$
- nonnegative θ_i means predictor function is convex (curves up)
- NNLS loss 0.59, LS loss 0.30, heuristic loss 15.05

How to choose a regularizer

use out-of-sample or cross-validation to choose among regularizers

- for each candidate regularizer, choose λ to minimize test error
  (and maybe a little larger ...)
- use the regularizer that gives the best test error
- then make up a story about why you knew that would be the best