Non-Quadratic Regularizers

Sanjay Lall and Stephen Boyd

EE104
Stanford University

Regularizers

Regularizers and sensitivity

- we want to choose θ to achieve low empirical risk $\mathcal{L}(\theta)$
- but also, we'd like the predictor g_θ to not be too sensitive
- roughly: for x near $\tilde{x}$, $g_\theta(x)$ should be near $g_\theta(\tilde{x})$
  - sensitive predictors sometimes don't generalize well
  - insensitive predictors often generalize well
- a regularizer $r : \mathbf{R}^p \to \mathbf{R}$ is a function that measures the sensitivity of g_θ
- often predictor sensitivity corresponds to the size of θ
- another interpretation:
  - the regularizer encodes *prior information* we have about θ
  - specifically, that $r(\theta)$ is small
- with either interpretation, we want both $\mathcal{L}(\theta)$ and $r(\theta)$ small

Regularized empirical risk minimization

- in RERM we choose θ to minimize $\mathcal{L}(\theta) + \lambda r(\theta)$
- $\lambda > 0$ is the regularization hyper-parameter, used to trade off $\mathcal{L}(\theta)$ and $r(\theta)$
- we choose λ (and r) by validation on a test set
- we use a regularizer to achieve better test set performance

Penalty based regularizers

- many common regularizers are given by a penalty function $q : \mathbf{R} \to \mathbf{R}$

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

- usually $q(a) \geq 0$ for all a, and $q(0) = 0$
- $q(\theta_i)$ expresses our displeasure in choosing predictor coefficient θ_i
- common examples:
  - sum square, quadratic, Tychonov, ℓ_2, or ridge regularizer: $q^{\mathsf{sqr}}(a) = a^2$, so $r(\theta) = ||\theta||_2^2$
  - sum absolute, ℓ_1, or lasso regularizer: $q^{\mathsf{abs}}(a) = |a|$, so $r(\theta) = ||\theta||_1$
  - nonnegative regularizer: $q^{\mathsf{nn}}(a) = \begin{cases} 0 & a \geq 0 \\ \infty & a < 0 \end{cases}$ (requires predictor coefficients to be nonnegative)

Sensitivity of linear predictors

Feature perturbation

- consider a linear predictor $g_\theta(x) = \theta^T x$
- suppose the feature vector x changes to $\tilde{x} = x + \delta$
- δ is the *perturbation* or change in x
- we'll assume that any $\delta \in \Delta$ is possible
- Δ is called the *feature perturbation set*
- the change in prediction if x changes to $\tilde{x} = x + \delta$ is $|\theta^T \tilde{x} - \theta^T x| = |\theta^T \delta|$
- how big can this be, over all $\delta \in \Delta$?
- we define the *worst case sensitivity* as $\max_{\delta \in \Delta} |\theta^T \delta|$
- it is evidently a measure of sensitivity

Worst case sensitivity with ℓ_2 perturbation

- let's take $\Delta = \{\delta \mid ||\delta||_2 \leq \epsilon\}$ (called an *ℓ_2-ball*)
- means the feature vector x can change to any $\tilde{x}$ within ℓ_2 distance ϵ
- by Cauchy-Schwarz inequality, $|\theta^\mathsf{T}\delta| \leq ||\theta||_2||\delta||_2 \leq \epsilon||\theta||_2$
- and the choice $\delta = \frac{\epsilon}{||\theta||_2}\theta$ achieves this maximum change in prediction
- so the worst-case sensitivity is $\epsilon||\theta||_2$
- justifies sum square regularizer $r(\theta) = ||\theta||_2^2 = \theta_1^2 + \cdots + \theta_d^2$

Worst case sensitivity with ℓ_∞ perturbation

- let's take $\Delta = \{\delta \mid |\delta_i| \leq \epsilon,\ i = 1, \ldots, d\}$ (called an *ℓ_∞-ball*)
- also expressed as $\Delta = \{\delta \mid ||\delta||_\infty \leq \epsilon\}$, where $||\delta||_\infty = \max_{i=1,\ldots,d} |\delta_i|$ is the ℓ_∞-norm of δ
- means any component of the feature vector x can change by up to ϵ
- how big can $|\theta^\mathsf{T}\delta|$ be, when $\delta \in \Delta$?
- 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

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

- so the worst case sensitivity is $\epsilon||\theta||_1$
- justifies sum absolute regularizer $r(\theta) = ||\theta||_1 = |\theta_1| + \cdots + |\theta_d|$

Ridge and 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)

Regulization with a constant feature

- suppose we have a constant feature $x_1 = 1$
- associated predictor coefficient θ_1 is the offset
- since x_1 does not change, $\delta_1 = 0$ always
- so θ_1 does not contribute to predictor sensitivity
- for this reason it's common to *not* regularize the associated coefficient θ_1
- we modify sum square regularizer to $r(\theta) = ||\theta_{2:d}||_2^2 = \theta_2^2 + \cdots + \theta_d^2$
- we modify sum absolute regularizer to $r(\theta) = ||\theta_{2:d}||_1 = |\theta_2| + \cdots + |\theta_d|$

Sparsifying regularizers

Sparse coefficient vector

- consider linear predictor $g_\theta(x) = \theta^\mathsf{T} x$
- 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 regressors are actually irrelevant
  - makes predictor *simpler to interpret*
- choosing the sparsity pattern of θ (*i.e.*, which entries are zero) is sometimes called *feature selection*
- there are many ways to carry out feature selection

Sparse coefficient vectors via ℓ_1 regularization

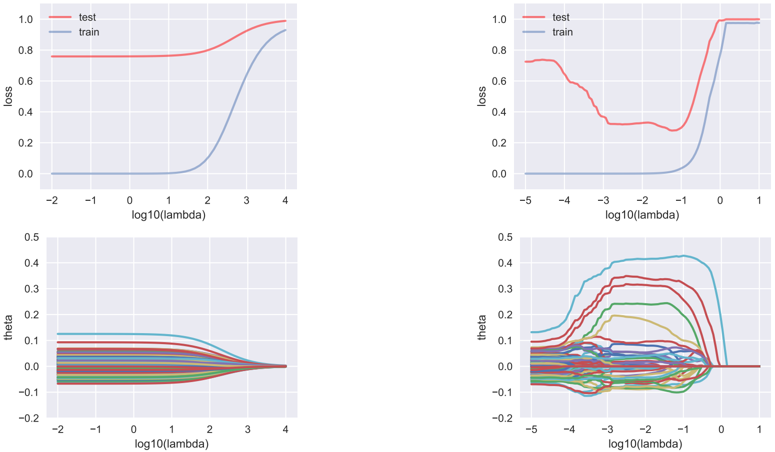
using ℓ_1 regularization leads to sparse coefficient vectors

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

rough explanation:

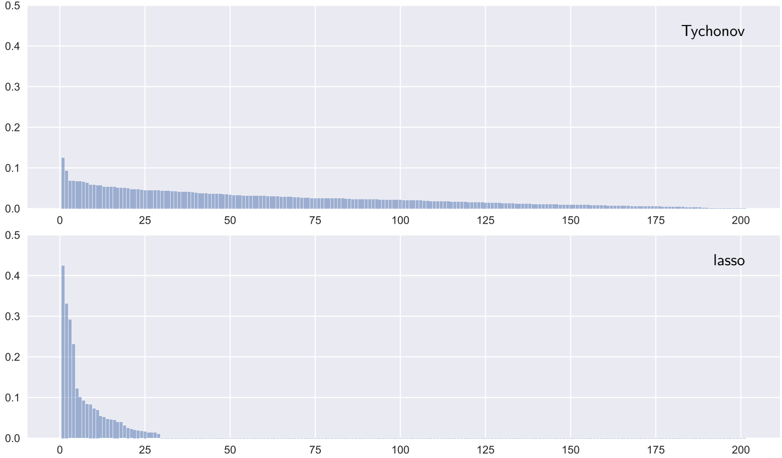
- for square penalty, once θ_i is small, θ_i^2 is very small
- so incentive for sum square regularizer to make a coefficient smaller decreases once it is small
- for absolute penalty, incentive to make θ_i smaller keeps up all the way until it's zero

Example



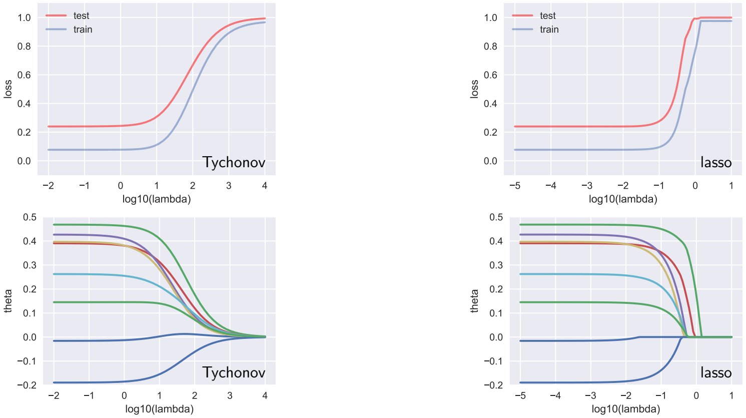
- artificially generated 50 data points, 200 features, only a few of which are relevant
- left hand plots use ridge regression, right hand use lasso

Example



- sorted $|\theta_i|$ at optimal λ
- lasso parameter has only 35 nonzero components; ridge regression has all 200 coefficients nonzero

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

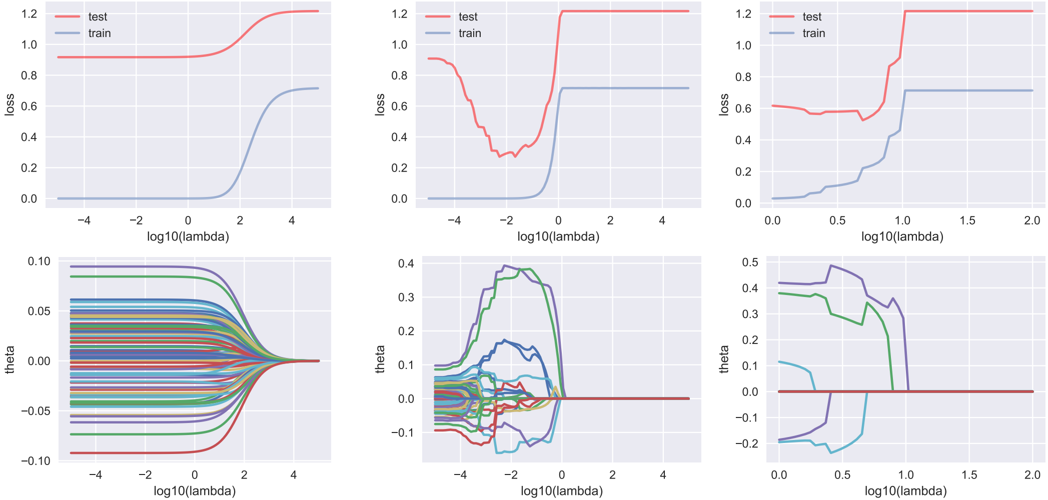
Even stronger sparsifiers

- $q(a) = |a|^{1/2}$
- 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

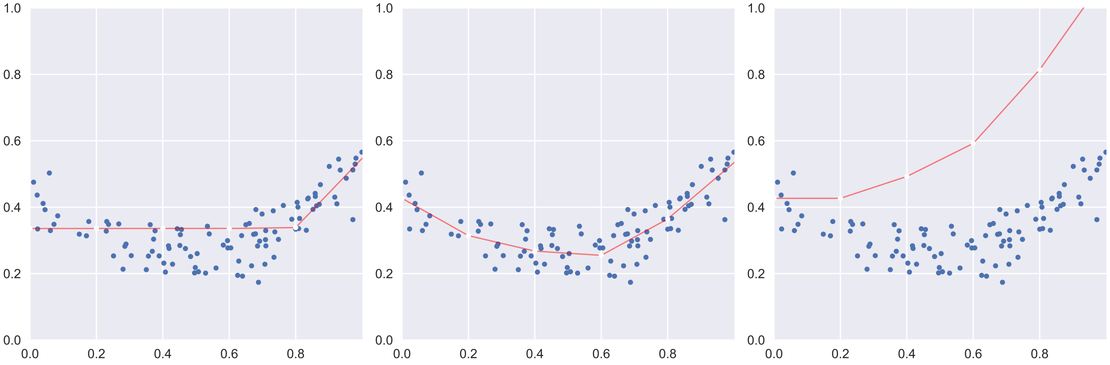
Nonegative coefficients

- 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 both predictor function is convex (curves up) and nondecreasing
- 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