Lecture 6: Regularization; Introduction to Classification

TTIC 31020: Introduction to Machine Learning

Instructor: Kevin Gimpel

TTI-Chicago

October 17, 2019

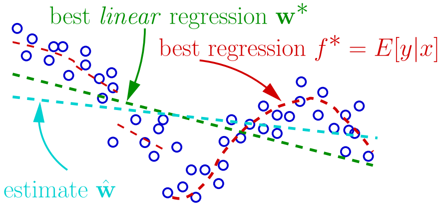
Administrivia

- Problem set 1 due tomorrow (Oct 18) 8:00pm
- Quiz 1 (on regression) on Tuesday, Oct 22
- Recitations:
  - This week: working through problems involving regression and likelihood
  - Next week: working through problems involving optimization and constraints; going over problem set 1

Review: Decomposition of error

- Approximation error

$$\mathbb{E}\left[\left(y - \mathbf{w}^{*\top}\mathbf{x}\right)^2\right]$$

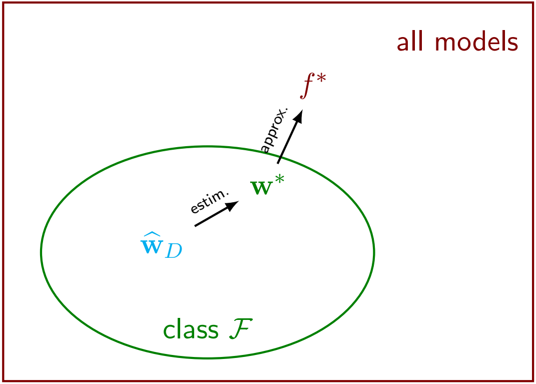


- Estimation error

$$\mathbb{E}\left[\left(\mathbf{w}^{*\top}\mathbf{x} - \hat{\mathbf{w}}^\top\mathbf{x}\right)^2\right]$$

- Approximation error: due to the failure to include optimal predictor in the model class; could be reduced with a more complex model, but limited by inherent uncertainty in $y|\mathbf{x}$
- Estimation error: due to failure to select the best predictor in the chosen model class; could be reduced with more data, or a less complex model

Error decomposition



- f^*: best possible model
- $\mathcal{F}$: parametric class
- $\mathbf{w}^*$: parameters of the best model in class
- $\widehat{\mathbf{w}}_D$: parameters learned on dataset D

- This is a general picture affecting any modeling problem, not just regression.

Review: regularization

- Intuition 1: more complex models get an “unfair advantage” in ERM
- General form of a regularized objective:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \left\{ \underbrace{L(\mathbf{X}, \mathbf{y}; \mathbf{w})}_{\text{empirical risk}} + \underbrace{R(\mathbf{w})}_{\text{regularizer}} \right\}$$

- Intuition 2: advantage of complex models due to freedom to set parameters to large values; limiting parameter values will restrict their ability to fit training data
- Ridge regression:

$$\mathbf{w}^*_{\text{ridge}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2 + \lambda \sum_{j=1}^{m} w_j^2 \right\}$$

convex, closed form solution

Lasso regression

- Use a penalty based on L_1 norm of parameters (aside from bias weight):

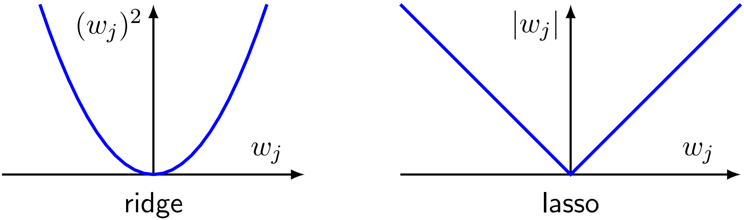
absolute value

$$\mathbf{w}^*_{\text{lasso}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2 + \lambda \sum_{j=1}^{m} |w_j| \right\}$$

- This is still convex, but not “smooth” (differentiable)
- Can solve it efficiently using convex programming methods or first-order numerical optimization (subgradient descent)
- Why is it called “lasso”?
  least absolute shrinkage and selection operator

Ridge vs. lasso regression

- Compare shape of the penalty as a function of w_j:



Optimization of ridge regression

- Can rewrite the optimization problem

$$\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2 + \lambda \sum_{j=1}^{m} w_j^2$$

in the objective/constraint form:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2$$

$$\text{subject to } \sum_{j=1}^{m} w_j^2 \leq t$$

- Correspondence $\lambda \Rightarrow t$ can be shown using Lagrange multipliers.

Optimization for Lasso

- Similarly, for Lasso:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2$$

$$\text{subject to} \sum_{j=1}^{m} |w_j| \leq t$$

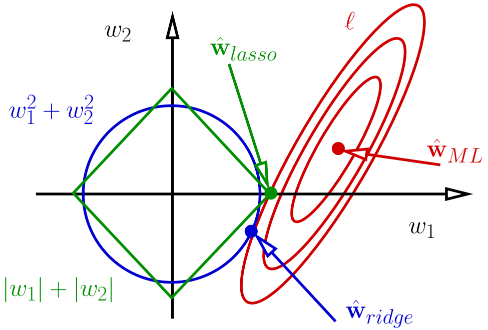
Lasso vs. ridge: geometry of error surfaces

- Constrained maximization formulation (lasso: $p = 1$, ridge: $p = 2$):

$$\hat{\mathbf{w}} = \underset{\mathbf{w}:\, \sum_{j=1}^{m} |w_j|^p \leq t}{\operatorname{argmax}} -\sum_{i=1}^{n} (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2$$

some of coefficient become 0

contour of loss



- With sufficiently large λ (=sufficiently small t) lasso leads to **sparsity**
- For sparsity, have to solve above optimization problem; getting close to the optimum with subgradient descent may not produce a sparse solution

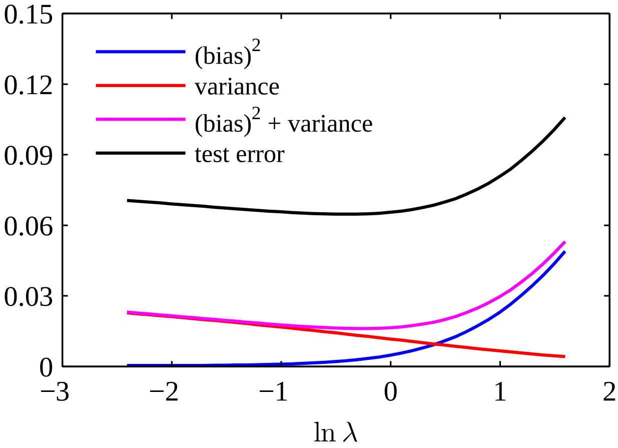
Regularization and bias/variance tradeoff

- Recall: $\mathbb{E}\left[\text{squared loss}\right] = \text{bias}^2 + \text{var} + \text{noise}$.
- How does increasing λ (stronger regularization) affect these?

Increase bias and variance go down

Regularization and bias/variance tradeoff

- Recall: $\mathbb{E}\,[\text{squared loss}] = \text{bias}^2 + \text{var} + \text{noise}$.
- In hindsight:

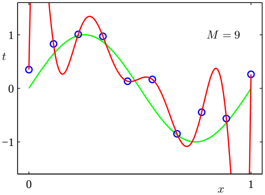


- In reality: often need to rely on procedure like (cross) validation

Choice of λ

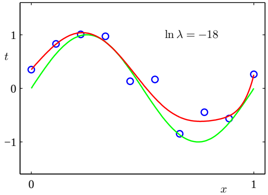
- Example [from Bishop, Ch. 1]: 9th degree polynomial, varying λ:

$\lambda = 0$



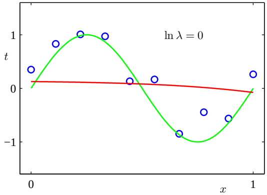
$$\|\mathbf{w}^*\|^2 > 10^{12}$$

$\lambda = e^{-18}$



$$\|\mathbf{w}^*\|^2 \approx 21595$$

$\lambda = 1$

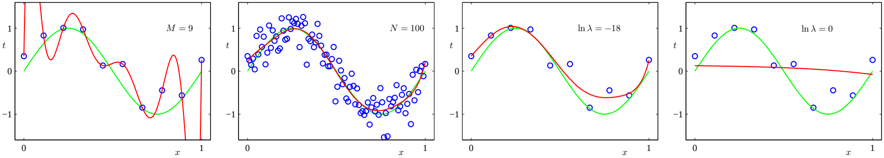


$$\|\mathbf{w}^*\|^2 \approx 0.027$$

- Most often: choose λ by (cross) validation

Regularization and data size

- Recall that regularization is a way to combat variance.
- Variance affects us due to limited training data.
- Adding training data is also a regularization technique!
- Consider fitting an $m = 9$ degree model to n datapoints:



$n = 10$, $\lambda = 0$

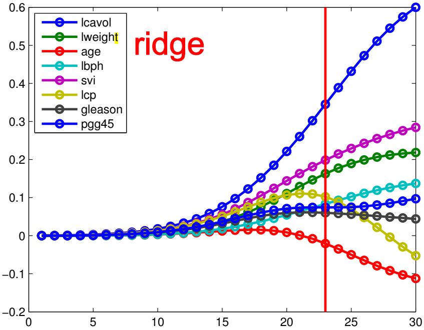
$n = 100$, $\lambda = 0$

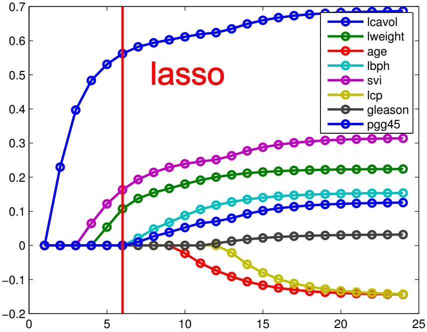
$n = 10$, $\lambda = e^{-18}$

$n = 10$, $\lambda = 1$

Example: lasso vs. ridge regularization paths

- Example: prostate data [Hastie, Tibshirani, and Friedman]
  Red lines: choice of λ by 5-fold cross validation.
  Vertical: value of w_j, horizontal: bound t on $R(\mathbf{w})$



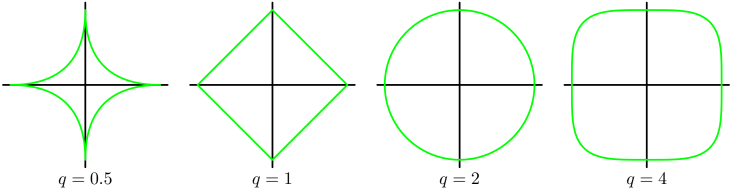


- This is the **regularization path** of the features
- Can we infer "importance" from these paths?
  Not really. Weight magnitude should be considered in the context of all the features.

probably noise for one feature and others are offsetting

General view of L_q penalty

- Can be creative in design of L_q penalty function: $\sum_j |w_j|^q$



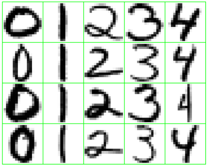
- For $q > 1$, no sparsity is achieved.
- For $q < 1$, non-convex
- What about L_0?

$$\min_{\mathbf{w}} \sum (y_i - \mathbf{w} \cdot \mathbf{x}_i)^2 \quad \text{s.t. } |\{w_j : w_j > 0\}| \leq M$$

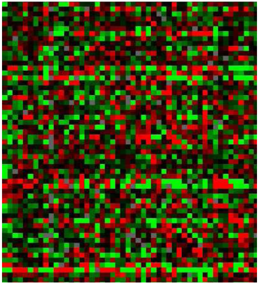
is NP-hard

Classification

- Shifting gears: classification. Many successful applications of ML: vision, speech, medicine, etc.
- Setup: mapping $\mathbf{x} \in \mathcal{X}$ to a **label** $y \in \mathcal{Y}$.
- Examples:



digit recognition
$\mathcal{Y} = \{0, \dots, 9\}$
multiclass classification (10 categories)



prediction from microarray data
$\mathcal{Y} = \{\text{disease present/absent}\}$
binary classification (2 categories)

Decision boundary

- Regression: once learned, maps any point in input space (image, document, feature vector) to a number; that's our prediction.
- Classification: once learned, maps any point in input space to a class (decision on what label to assign). A set of inputs mapped to a particular class is a **decision region**.
- Decision boundary is the boundary between decision regions; a boundary across which decisions flip.

Classification as regression

- Suppose we have a binary problem, $y \in \{-1, 1\}$
  note: sometimes will prefer $y \in \{0, 1\}$
- Idea: treat it as regression, with squared loss
- Assuming the standard model $y = f(\mathbf{x}; \mathbf{w}) + \nu$, and solving with least squares, we get $\widehat{\mathbf{w}}$.
- This corresponds to squared loss as a measure of classification performance! Does this make sense?
- How do we decide on the label based on $f(\mathbf{x}; \widehat{\mathbf{w}})$?

Classification as regression

$$f(\mathbf{x}; \widehat{\mathbf{w}}) \;=\; b + \widehat{\mathbf{w}} \cdot \mathbf{x}$$

(notation change: b instead of w_0)

- Can't just take $\widehat{y} = f(\mathbf{x}; \widehat{\mathbf{w}})$ since it won't be a valid label.
- A reasonable **decision rule**:

decide on $\widehat{y} = 1$ if $f(\mathbf{x}; \widehat{\mathbf{w}}) \;\geq\; 0$, otherwise $\widehat{y} = -1$.

$$\widehat{y} \;=\; \operatorname{sign}(b + \widehat{\mathbf{w}} \cdot \mathbf{x})$$

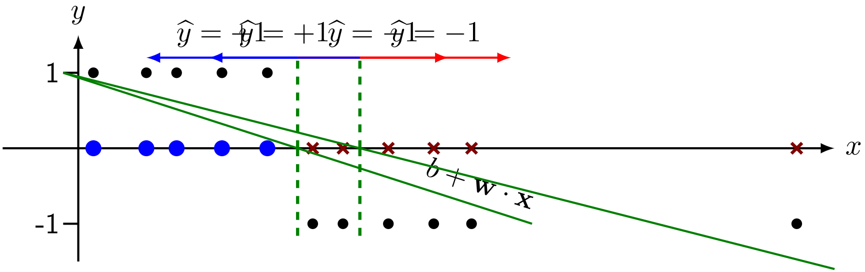
- This specifies a **linear classifier**:
  - The linear **decision boundary** (hyperplane) given by the equation $b + \widehat{\mathbf{w}} \cdot \mathbf{x} = 0$ separates the space into two "half-spaces".

Classification as regression: example

- Linear classifier:

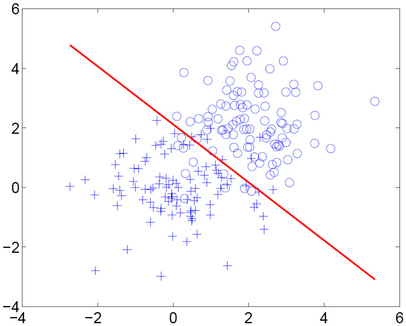
$$\widehat{y} \;=\; \operatorname{sign}\left(b + \widehat{\mathbf{w}} \cdot \mathbf{x}\right)$$

- A 1D example:

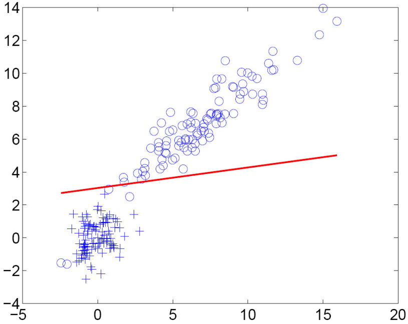


Classification as regression

- Same effect in 2D:



Seems to work well here



but not so well here