

Machine Learning & Data Mining

CS/CNS/EE 155

Lecture 4:
Regularization, Sparsity & Lasso

Recap: Complete Pipeline

$$S = \{(x_i, y_i)\}_{i=1}^N$$

Training Data

$$f(x | w, b) = w^T x - b$$

Model Class(es)

$$L(a, b) = (a - b)^2$$

Loss Function



$$\operatorname{argmin}_{w,b} \sum_{i=1}^N L(y_i, f(x_i | w, b)) \quad \text{SGD!}$$

Cross Validation & Model Selection



Profit!

Different Model Classes?

- Option 1: SVMs vs ANNs vs LR vs LS
- Option 2: Regularization



$$\operatorname{argmin}_{w,b} \sum_{i=1}^N L(y_i, f(x_i | w, b)) \quad \text{SGD!}$$

Cross Validation & Model Selection

Notation Part 1

- L0 Norm (not actually a norm)
 - # of non-zero entries

$$\|w\|_0 = \sum_d 1_{[w_d \neq 0]}$$

- L1 Norm
 - Sum of absolute values

$$|w| = \|w\|_1 = \sum_d |w_d|$$

- L2 Norm & Squared L2 Norm
 - Sum of squares
 - Sqrt(sum of squares)

$$\|w\| = \sqrt{\sum_d w_d^2} \equiv \sqrt{w^T w}$$

$$\|w\|^2 = \sum_d w_d^2 \equiv w^T w$$

- L-infinity Norm
 - Max absolute value

$$\|w\|_\infty = \lim_{p \rightarrow \infty} \sqrt[p]{\sum_d |w_d|^p} = \max_d |w_d|$$

Notation Part 2

- Minimizing Squared Loss

- Regression
 - Least-Squares

$$\operatorname{argmin}_w \sum_i (y_i - w^T x_i + b)^2$$

- (Unless Otherwise Stated)

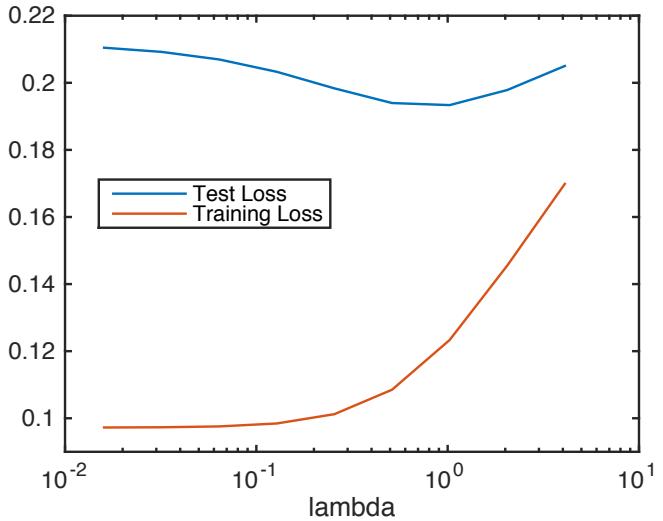
- E.g., Logistic Regression = Log Loss

Ridge Regression

$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_i (y_i - w^T x_i + b)^2$$

The diagram shows the cost function for Ridge Regression. It consists of two main components: a regularization term and a training loss term. The regularization term is represented by a blue bracket under the term $\lambda w^T w$, which is labeled "Regularization". The training loss term is represented by a blue bracket under the sum $\sum_i (y_i - w^T x_i + b)^2$, which is labeled "Training Loss".

- aka L2-Regularized Regression
- Trades off model complexity vs training loss
- Each choice of λ a “model class”
 - Will discuss the further later



$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_i (y_i - w^T x_i + b)^2$$

	w		b
0.7401	0.2441	-0.1745	
0.7122	0.2277	-0.1967	
0.6197	0.1765	-0.2686	
0.4124	0.0817	-0.4196	
0.1801	0.0161	-0.5686	
	⋮		
0.0001	0.0000	-0.6666	

Larger Lambda

Person	Age>10	Male?	Height > 55"
Alice	1	0	1
Bob	0	1	0
Carol	0	0	0
Dave	1	1	1
Erin	1	0	1
Frank	0	1	1
Gena	0	0	0
Harold	1	1	1
Irene	1	0	0
John	0	1	1
Kelly	1	0	1
Larry	1	1	1

Train Test

Updated Pipeline

$$S = \{(x_i, y_i)\}_{i=1}^N$$

Training Data

$$f(x | w, b) = w^T x - b$$

Model Class

$$L(a, b) = (a - b)^2$$

Loss Function



$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_{i=1}^N L(y_i, f(x_i | w, b))$$

Choosing $\lambda!$

Cross Validation & Model Selection

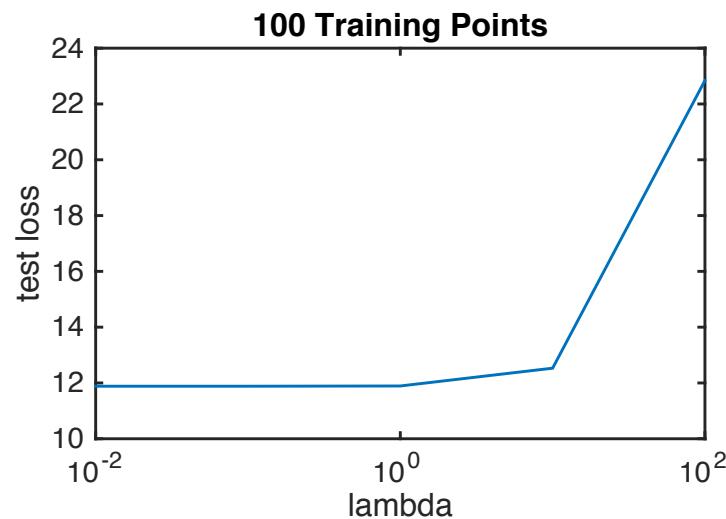
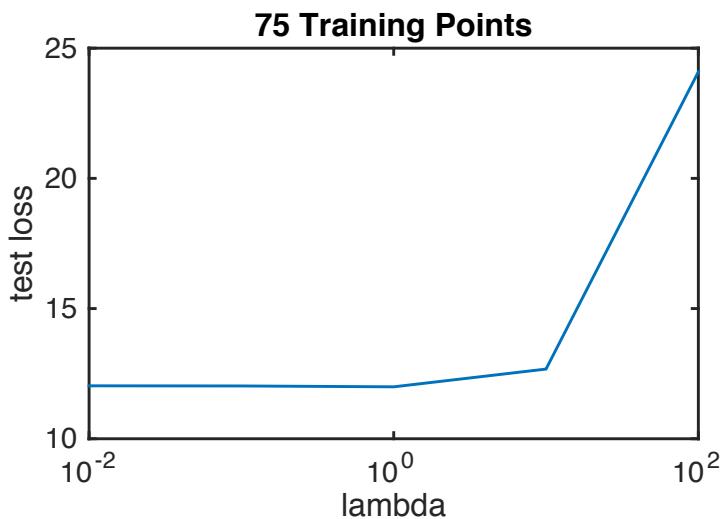
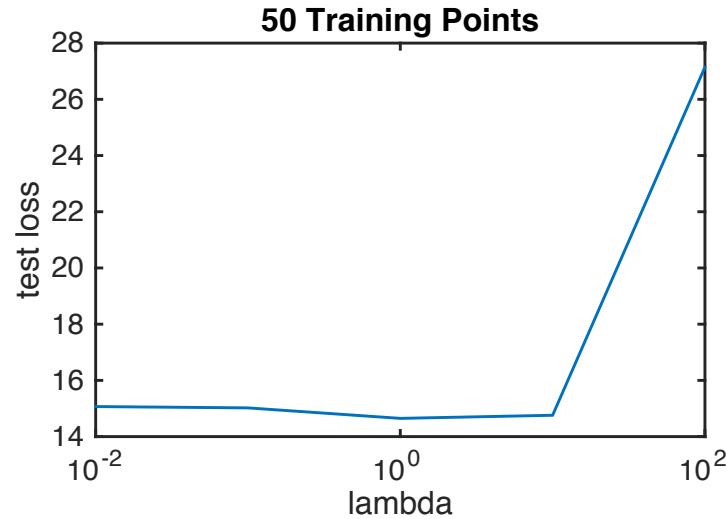
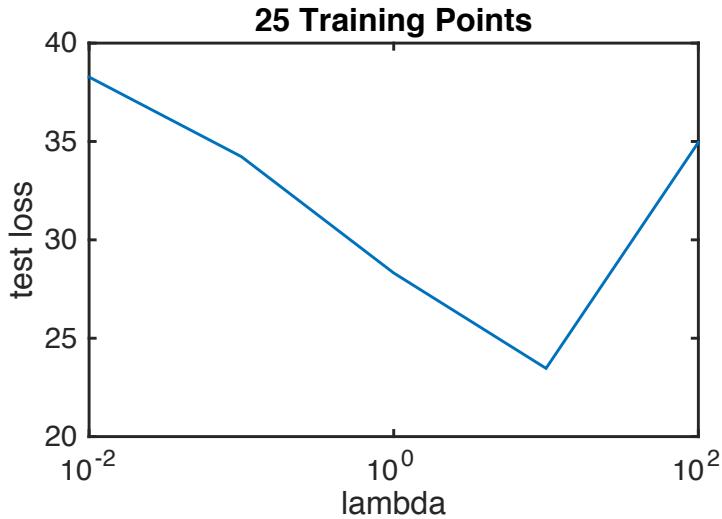


Profit!

Person	Age > 10	Male ?	Height > 55"	Model Score w/ Increasing Lambda					
				0.91	0.89	0.83	0.75	0.67	
Train	Alice	1	0	1	0.91	0.89	0.83	0.75	0.67
	Bob	0	1	0	0.42	0.45	0.50	0.58	0.67
	Carol	0	0	0	0.17	0.26	0.42	0.50	0.67
	Dave	1	1	1	1.16	1.06	0.91	0.83	0.67
	Erin	1	0	1	0.91	0.89	0.83	0.79	0.67
	Frank	0	1	1	0.42	0.45	0.50	0.54	0.67
Test	Gena	0	0	0	0.17	0.27	0.42	0.50	0.67
	Harold	1	1	1	1.16	1.06	0.91	0.83	0.67
	Irene	1	0	0	0.91	0.89	0.83	0.79	0.67
	John	0	1	1	0.42	0.45	0.50	0.54	0.67
	Kelly	1	0	1	0.91	0.89	0.83	0.79	0.67
	Larry	1	1	1	1.16	1.06	0.91	0.83	0.67

Best test error

Choice of Lambda Depends on Training Size



25 dimensional space

Randomly generated linear response function + noise

Recap: Ridge Regularization

- Ridge Regression:
 - L2 Regularized Least-Squares

$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_i (y_i - w^T x_i + b)^2$$

- Large $\lambda \rightarrow$ more stable predictions
 - Less likely to overfit to training data
 - Too large $\lambda \rightarrow$ underfit
- Works with other loss
 - Hinge Loss, Log Loss, etc.

Aside: Stochastic Gradient Descent

$$\operatorname{argmin}_{w,b} \lambda \|w\|^2 + \sum_{i=1}^N L(y_i, f(x_i | w, b))$$

$$\tilde{L}(w, b) = \sum_{i=1}^N \left[\frac{1}{N} \lambda \|w\|^2 + L_i(w, b) \right]$$



$$\frac{1}{N} \tilde{L}(w, b) = E_i \left[\tilde{L}_i(w, b) \right]$$



Do SGD on this

Model Class Interpretation

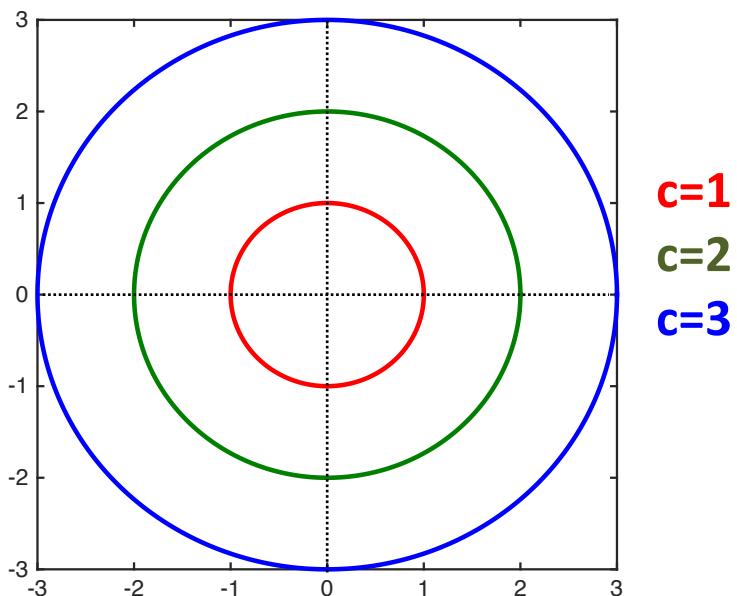
$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_{i=1}^N L(y_i, f(x_i | w, b))$$

- This is not a model class!
 - At least not what we've discussed...
- An optimization procedure
 - Is there a connection?

Norm Constrained Model Class

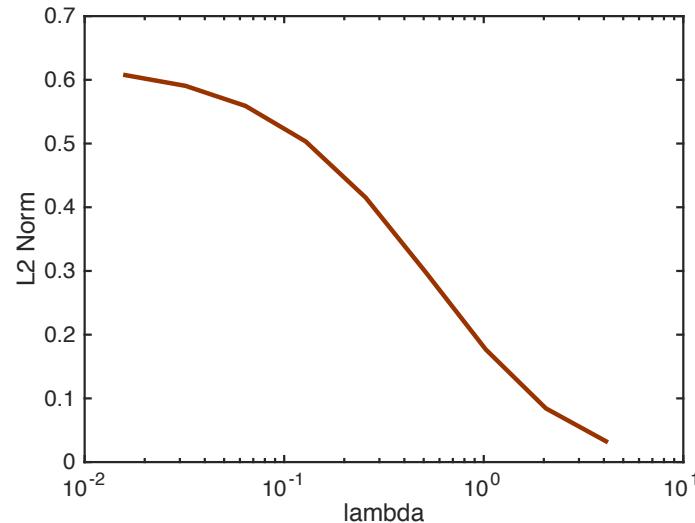
$$f(x \mid w, b) = w^T x - b \quad \text{s.t. } w^T w \leq c \equiv \|w\|^2 \leq c$$

Visualization



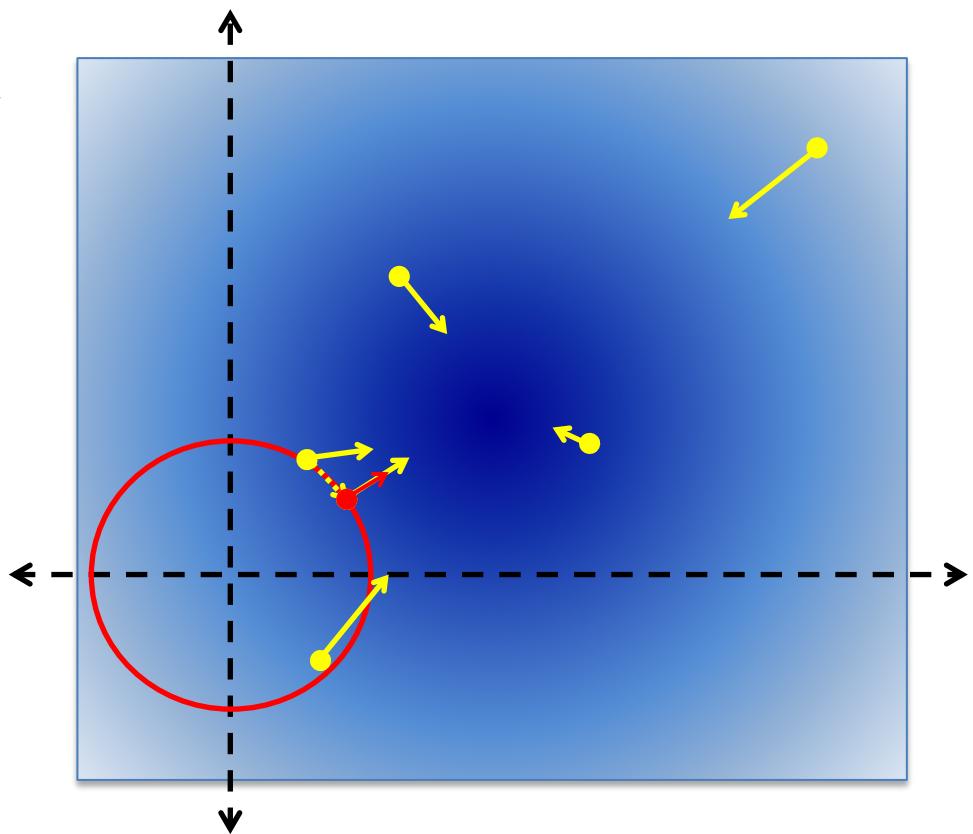
Seems to correspond to lambda...

$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_{i=1}^N L(y_i, f(x_i \mid w, b))$$



Lagrange Multipliers

$$\operatorname{argmin}_w L(y, w) \equiv (y - w^T x)^2$$



- Optimality Condition:
 - Gradients aligned!
 - **Constraint Boundary**
 - **Loss**

$$\exists \lambda \geq 0 : (\partial_w L(y, w) = -\lambda \partial_w w^T w) \wedge (w^T w \leq c)$$

Omitting b &
1 training data
for simplicity

Norm Constrained Model Class Training:

$$\underset{w}{\operatorname{argmin}} L(y, w) \equiv (y - w^T x)^2 \quad \text{s.t. } w^T w \leq c$$

Omitting b &
1 training data
for simplicity

Two Conditions
Must Be Satisfied
At Optimality \Leftrightarrow .

Observation about Optimality:

$$\exists \lambda \geq 0 : (\partial_w L(y, w) = -\lambda \partial_w w^T w) \wedge (w^T w \leq c)$$

Lagrangian:

$$\min_w \max_{\lambda} \Lambda(w, \lambda) = (y - w^T x)^T + \lambda \max\{w^T w - c, 0\}$$

Claim: Solving Lagrangian
Solves Norm-Constrained
Training Problem

Satisfies First Condition!

Optimality Implication of Lagrangian:

$$\partial_w \Lambda(w, \lambda) = -2x(y - w^T x)^T + 2\lambda w \equiv 0$$

$$\rightarrow 2x(y - w^T x)^T = -2\lambda w$$

Norm Constrained Model Class Training:

$$\underset{w}{\operatorname{argmin}} L(y, w) \equiv (y - w^T x)^2 \quad \text{s.t. } w^T w \leq c$$

Omitting b &
1 training data
for simplicity

Two Conditions
Must Be Satisfied
At Optimality \Leftrightarrow .

Observation about Optimality:

$$\exists \lambda \geq 0 : (\partial_w L(y, w) = -\lambda \partial_w w^T w) \wedge (w^T w \leq c)$$

Lagrangian:

$$\min_w \max_{\lambda} \Lambda(w, \lambda) = (y - w^T x)^T + \lambda \max\{w^T w - c, 0\}$$

Claim: Solving Lagrangian
Solves Norm-Constrained
Training Problem

Optimality Implication of Lagrangian:

Satisfies 2nd
Condition!

$$\partial_{\lambda} \Lambda(w, \lambda) = \begin{cases} 0 & \text{if } w^T w < c \\ w^T w - c & \text{if } w^T w \geq c \end{cases} \equiv 0 \rightarrow w^T w \leq c$$

Norm Constrained Model Class Training:

$$\operatorname{argmin}_w L(y, w) \equiv (y - w^T x)^2 \quad \text{s.t. } w^T w \leq c$$

L2 Regularized Training:

$$\operatorname{argmin}_w \lambda w^T w + (y - w^T x)^2$$

Lagrangian:

$$\min_w \max_\lambda \Lambda(w, \lambda) = (y - w^T x)^T + \lambda \max\{w^T w - c, 0\}$$

- Lagrangian = Norm Constrained Training:

$$\exists \lambda \geq 0 : (\partial_w L(y, w) = -\lambda \partial_w w^T w) \wedge (w^T w \leq c)$$

- Lagrangian = L2 Regularized Training:

- Hold λ fixed

- Equivalent to solving Norm Constrained!

Omitting b &
1 training data
for simplicity

- For some c

Recap #2: Ridge Regularization

- Ridge Regression:
 - L2 Regularized Least-Squares = Norm Constrained Model
$$\operatorname{argmin}_{w,b} \lambda w^T w + L(w) \quad \equiv \quad \operatorname{argmin}_{w,b} L(w) \text{ s.t. } w^T w \leq c$$
- Large $\lambda \rightarrow$ more stable predictions
 - Less likely to overfit to training data
 - Too large $\lambda \rightarrow$ underfit
- Works with other loss
 - Hinge Loss, Log Loss, etc.

Hallucinating Data Points

$$\operatorname{argmin}_w \lambda w^T w + \sum_{i=1}^N (y_i - w^T x_i)^2$$

$$\partial_w = 2\lambda w - 2 \sum_{i=1}^N x (y_i - w^T x_i)^T$$

- Instead hallucinate D data points?

$$\operatorname{argmin}_w \sum_{d=1}^D (0 - w^T \sqrt{\lambda} e_d)^2 + \sum_{i=1}^N (y_i - w^T x_i)^2$$

$$\partial_w = 2 \sum_{d=1}^D \sqrt{\lambda} e_d (w^T \sqrt{\lambda} e_d)^T - 2 \sum_{i=1}^N x (y_i - w^T x_i)^T$$



$$= 2 \sum_{d=1}^D \lambda e_d^T w = 2 \sum_{d=1}^D \lambda w_d = 2\lambda w$$

**Identical to
Regularization!**

$$\left\{ (\sqrt{\lambda} e_d, 0) \right\}_{d=1}^D$$

Unit vector
along d-th
Dimension

$$e_d = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Omitting b &
for simplicity

Extension: Multi-task Learning

- 2 prediction tasks:
 - Spam filter for Alice
 - Spam filter for Bob
- Limited training data for both...
 - ... but Alice is similar to Bob

Extension: Multi-task Learning

- Two Training Sets
 - N relatively small
- **Option 1:** Train Separately

$$\underset{w}{\operatorname{argmin}} \lambda w^T w + \sum_{i=1}^N \left(y_i^{(1)} - w^T x_i^{(1)} \right)^2$$

$$S^{(1)} = \left\{ (x_i^{(1)}, y_i^{(1)}) \right\}_{i=1}^N$$

$$\underset{v}{\operatorname{argmin}} \lambda v^T v + \sum_{i=1}^N \left(y_i^{(2)} - v^T x_i^{(2)} \right)^2$$

$$S^{(2)} = \left\{ (x_i^{(2)}, y_i^{(2)}) \right\}_{i=1}^N$$

Both models have high error.

Omitting b &
for simplicity

Extension: Multi-task Learning

- Two Training Sets
 - N relatively small
- **Option 2:** Train Jointly

$$\begin{aligned} \operatorname{argmin}_{w,v} & \lambda w^T w + \sum_{i=1}^N \left(y_i^{(1)} - w^T x_i^{(1)} \right)^2 \\ & + \lambda v^T v + \sum_{i=1}^N \left(y_i^{(2)} - v^T x_i^{(2)} \right)^2 \end{aligned}$$

$$S^{(1)} = \left\{ (x_i^{(1)}, y_i^{(1)}) \right\}_{i=1}^N$$

$$S^{(2)} = \left\{ (x_i^{(2)}, y_i^{(2)}) \right\}_{i=1}^N$$

Doesn't accomplish anything!
(w & v don't depend on each other)

Omitting b &
for simplicity

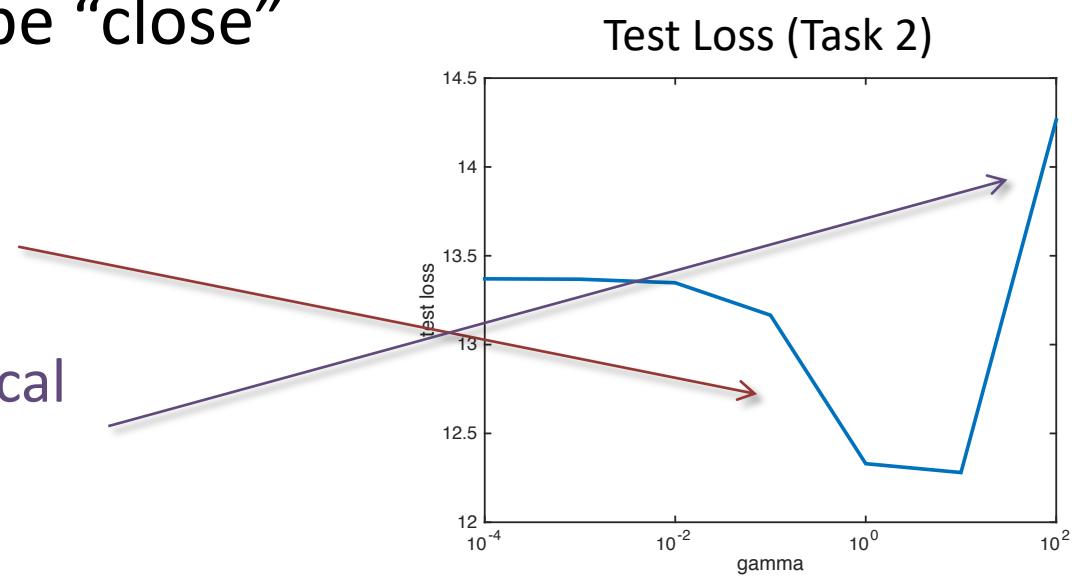
Multi-task Regularization

$$\underset{w,v}{\operatorname{argmin}} \lambda w^T w + \lambda v^T v + \gamma (w - v)^T (w - v) + \sum_{i=1}^N (y_i^{(1)} - w^T x_i^{(1)})^2 + \sum_{i=1}^N (y_i^{(2)} - v^T x_i^{(2)})^2$$

Standard Regularization Multi-task Regularization Training Loss

- Prefer w & v to be “close”

- Controlled by γ
- Tasks similar
 - Larger γ helps!
- Tasks not identical
 - γ not too large



Lasso

L1-Regularized Least-Squares

L1 Regularized Least Squares

$$\operatorname{argmin}_w \lambda |w| + \sum_{i=1}^N (y_i - w^T x_i)^2$$

$$\operatorname{argmin}_w \lambda \|w\|^2 + \sum_{i=1}^N (y_i - w^T x_i)^2$$

- L2:

$$w = \sqrt{2} \quad \text{vs} \quad w = 1$$

||

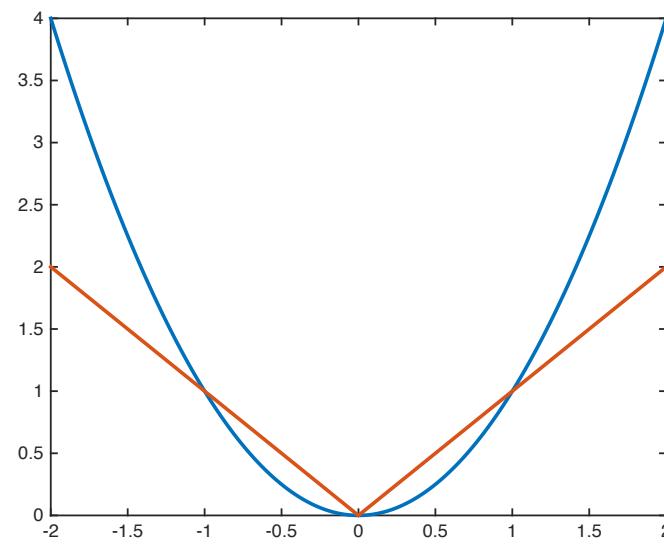
$$w = 1 \quad \text{vs} \quad w = 0$$

- L1:

$$w = 2 \quad \text{vs} \quad w = 1$$

||

$$w = 1 \quad \text{vs} \quad w = 0$$



Omitting b &
for simplicity

Aside: Subgradient (sub-differential)

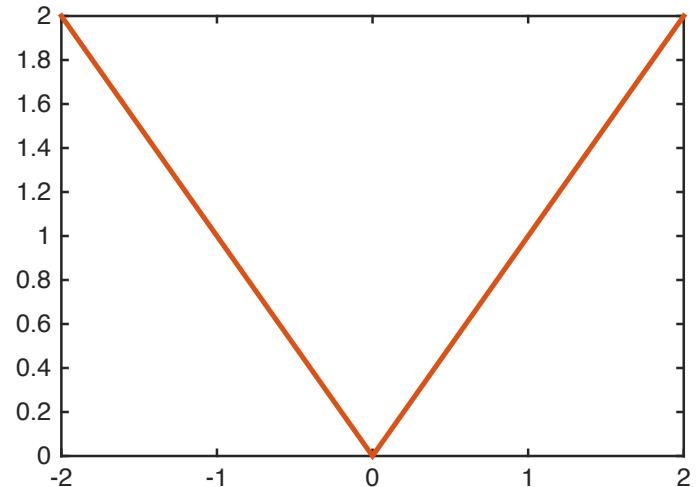
$$\nabla_a R(a) = \{c \mid \forall a' : R(a') - R(a) \geq c(a' - a)\}$$

- Differentiable: $\nabla_a R(a) = \partial_a R(a)$

- L1:

$$\nabla_{w_d} |w| \begin{cases} -1 & \text{if } w_d < 0 \\ +1 & \text{if } w_d > 0 \\ [-1, +1] & \text{if } w_d = 0 \end{cases}$$

Continuous range for $w=0!$



Omitting b &
for simplicity

L1 Regularized Least Squares

$$\operatorname{argmin}_w \lambda |w| + \sum_{i=1}^N (y_i - w^T x_i)^2$$

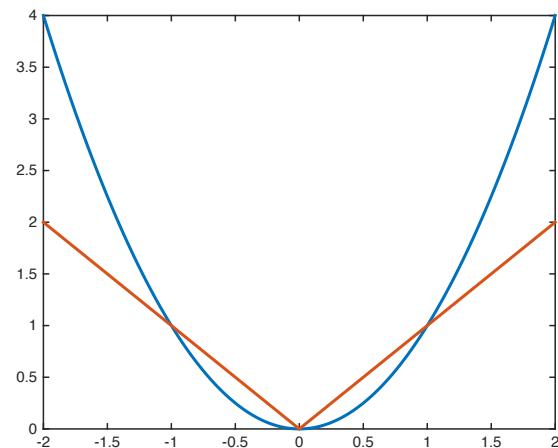
$$\operatorname{argmin}_w \lambda \|w\|^2 + \sum_{i=1}^N (y_i - w^T x_i)^2$$

- L2:

$$\nabla_{w_d} \|w\|^2 = 2w_d$$

- L1:

$$\nabla_{w_d} |w| \begin{cases} -1 & \text{if } w_d < 0 \\ +1 & \text{if } w_d > 0 \\ [-1, +1] & \text{if } w_d = 0 \end{cases}$$



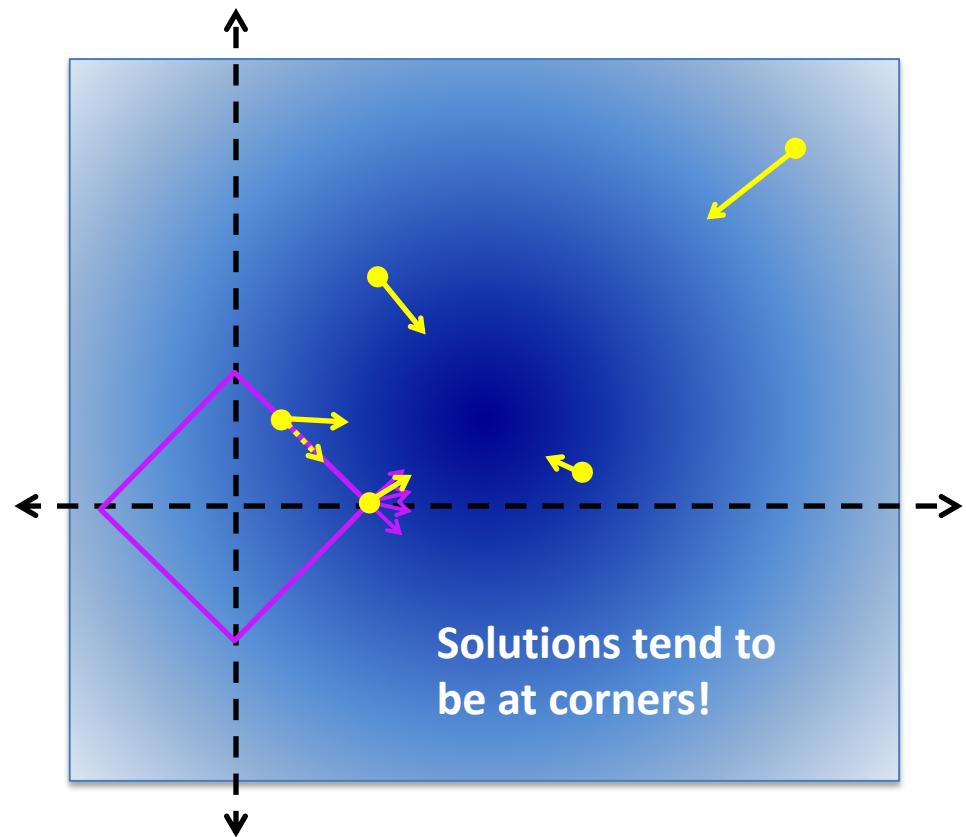
Omitting b &
for simplicity

Lagrange Multipliers

$$\underset{w}{\operatorname{argmin}} L(y, w) \equiv (y - w^T x)^2$$

$$\text{s.t. } |w| \leq c$$

$$\nabla_{w_d} |w| \begin{cases} -1 & \text{if } w_d < 0 \\ +1 & \text{if } w_d > 0 \\ [-1, +1] & \text{if } w_d = 0 \end{cases}$$



$$\exists \lambda \geq 0 : (\partial_w L(y, w) \in -\lambda \nabla_w |w|) \wedge (|w| \leq c)$$

Omitting b &
1 training data
for simplicity

Sparsity

- w is sparse if mostly 0's:
 - Small L0 Norm

$$\|w\|_0 = \sum_d 1_{[w_d \neq 0]}$$

- Why not L0 Regularization?

- **Not continuous!**

$$\operatorname{argmin}_w \lambda \|w\|_0 + \sum_{i=1}^N (y_i - w^T x_i)^2$$

- L1 induces sparsity
 - And is continuous!

$$\operatorname{argmin}_w \lambda |w| + \sum_{i=1}^N (y_i - w^T x_i)^2$$

Omitting b &
for simplicity

Why is Sparsity Important?

- Computational / Memory Efficiency
 - Store 1M numbers in array
 - Store 2 numbers per non-zero
 - (Index, Value) pairs
 - E.g., [(50,1), (51,1)]
 - Dot product more efficient: $w^T x$
- Sometimes true w is sparse
 - Want to recover non-zero dimensions

$$\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Lasso Guarantee

$$\underset{w}{\operatorname{argmin}} \lambda|w| + \sum_{i=1}^N (y_i - w^T x_i + b)^2$$

- Suppose data generated as: $y_i \sim Normal(w_*^T x_i, \sigma^2)$
- Then if: $\lambda > \frac{2}{\kappa} \sqrt{\frac{2\sigma^2 \log D}{N}}$
- With high probability (increasing with N):

$$Supp(w) \subseteq Supp(w_*)$$

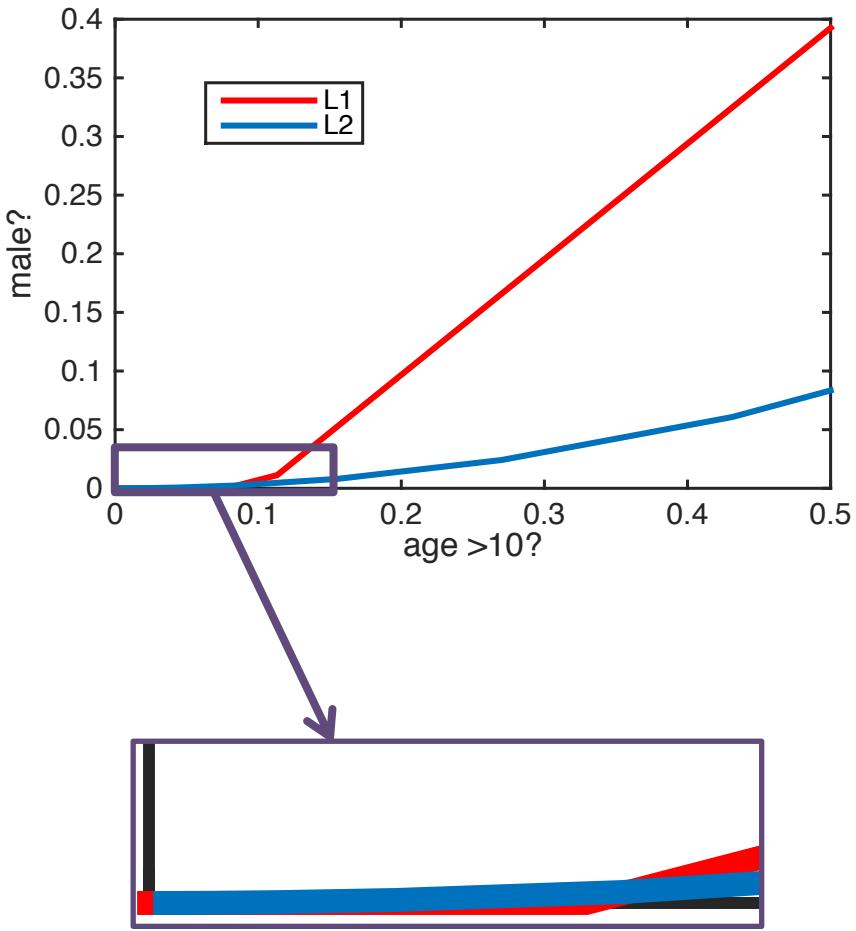
High Precision
Parameter Recovery

$$\forall d : |w_d| \geq \lambda c \rightarrow Supp(w) = Supp(w_*)$$

Sometimes High Recall

$$Supp(w_*) = \{d | w_{*,d} \neq 0\}$$

See also: <https://www.cs.utexas.edu/~pradeepr/courses/395T-LT/filez/highdimII.pdf>
http://www.eecs.berkeley.edu/~wainwrig/Papers/Wai_SparseInfo09.pdf



Magnitude of the two weights.
(As regularization shrinks)

Person	Age>10	Male?	Height > 55"
Alice	1	0	1
Bob	0	1	0
Carol	0	0	0
Dave	1	1	1
Erin	1	0	1
Frank	0	1	1
Gena	0	0	0
Harold	1	1	1
Irene	1	0	0
John	0	1	1
Kelly	1	0	1
Larry	1	1	1

Aside: Optimizing Lasso

- Solving Lasso gives sparse model
 - Will stochastic gradient descent find it?
- No!
 - Hard to hit exactly 0 with gradient descent
- **Solution:** Iterative Soft Thresholding
 - Intuition: if gradient update passes 0, clamp at 0

Recap: Lasso vs Ridge

- Model Assumptions
 - Lasso learns sparse weight vector
- Predictive Accuracy
 - Lasso often not as accurate
 - **Re-run Least Squares on dimensions selected by Lasso**
- Ease of Inspection
 - Sparse w's easier to inspect
- Ease of Optimization
 - Lasso somewhat trickier to optimize

Recap: Regularization

- L2

$$\operatorname{argmin}_w \lambda \|w\|^2 + \sum_{i=1}^N (y_i - w^T x_i)^2$$

- L1 (Lasso)

$$\operatorname{argmin}_w \lambda |w| + \sum_{i=1}^N (y_i - w^T x_i)^2$$

- Multi-task

$$\begin{aligned} & \operatorname{argmin}_{w,v} \lambda w^T w + \lambda v^T v + \gamma (w - v)^T (w - v) \\ & + \sum_{i=1}^N (y_i^{(1)} - w^T x_i^{(1)})^2 + \sum_{i=1}^N (y_i^{(2)} - v^T x_i^{(2)})^2 \end{aligned}$$

- [Insert Yours Here!]

Omitting b &
for simplicity

Recap: Updated Pipeline

$$S = \{(x_i, y_i)\}_{i=1}^N$$

Training Data

$$f(x | w, b) = w^T x - b$$

Model Class

$$L(a, b) = (a - b)^2$$

Loss Function



$$\operatorname{argmin}_{w,b} \lambda w^T w + \sum_{i=1}^N L(y_i, f(x_i | w, b))$$

Choosing λ !

Cross Validation & Model Selection



Profit!

Next Lectures

- Decision Trees
- Bagging
- Random Forests
- Boosting
- Ensemble Selection