Lasso Regression: 
Regularization for feature selection

CSE 446: Machine Learning 
Emily Fox 
University of Washington 
January 18, 2017

Feature selection task
Why might you want to perform feature selection?

Efficiency:
- If $\text{size}(w) = 100B$, each prediction is expensive
- If $w$ is sparse, computation only depends on # of non-zeros

\[
\hat{y}_i = \sum_{\hat{w}_j \neq 0} \hat{w}_j h_j(x_i)
\]

Interpretability:
- Which features are relevant for prediction?

Sparsity: Housing application

Lot size, Single Family, Year built, Last sold price, Last sale price/sqft, Finished sqft, Unfinished sqft, Finished basement sqft, # floors, Flooring types, Parking type, Parking amount, Cooling, Heating, Exterior materials, Roof type, Structure style

Dishwasher, Garbage disposal, Microwave, Range / Oven, Refrigerator, Washer, Dryer, Laundry location, Heating type, Jetted Tub, Deck, Fenced Yard, Lawn, Garden, Sprinkler System
Option 1: All subsets or greedy variants

Exhaustive approach: “all subsets”

Consider all possible models, each using a subset of features

How many models were evaluated?

\[
\begin{align*}
y_i &= \epsilon_i \\
y_i &= w_0 h_0(x_i) + \epsilon_i \\
y_i &= w_1 h_1(x_i) + \epsilon_i \\
&\quad \vdots \\
y_i &= w_0 h_0(x_i) + w_1 h_1(x_i) + \epsilon_i \\
&\quad \vdots \\
y_i &= w_0 h_0(x_i) + w_1 h_1(x_i) + \ldots + w_D h_D(x_i) + \epsilon_i
\end{align*}
\]

Each indexed by features included

\[
\begin{bmatrix}
0 & 0 & 0 & \ldots & 0 & 0 \\
1 & 0 & 0 & \ldots & 0 & 0 \\
0 & 1 & 0 & \ldots & 0 & 0 \\
1 & 1 & 0 & \ldots & 0 & 0 \\
1 & 1 & 1 & \ldots & 1 & 1 \\
&\quad \vdots \\
1 & 1 & \ldots & \ldots & \ldots & \ldots
\end{bmatrix}
\]

$2^D = 256$

$2^{30} = 1,073,741,824$

$2^{1000} = 1.071509 \times 10^{301}$

$2^{1008} = \text{HUGE}!!!!!!$

Typically, computationally infeasible
Choosing model complexity?

Option 1: Assess on validation set

Option 2: Cross validation

Option 3+: Other metrics for penalizing model complexity like BIC…

Greedy algorithms

Forward stepwise:
Starting from simple model and iteratively add features most useful to fit

Backward stepwise:
Start with full model and iteratively remove features least useful to fit

Combining forward and backward steps:
In forward algorithm, insert steps to remove features no longer as important

Lots of other variants, too.
Option 2: Regularize

Ridge regression: $L_2$ regularized regression

$$\text{Total cost} = \text{measure of fit} + \lambda \text{measure of magnitude of coefficients}$$

$$\text{RSS}(w) + \lambda \sum \text{magnitude of coefficients}$$

$$||w||_2^2 = w_0^2 + \ldots + w_D^2$$

Encourages small weights but not exactly 0
Using regularization for feature selection

Instead of searching over a discrete set of solutions, can we use regularization?

- Start with full model (all possible features)
- “Shrink” some coefficients exactly to 0
  - i.e., knock out certain features
- Non-zero coefficients indicate “selected” features
Thresholding ridge coefficients?

Why don’t we just set small ridge coefficients to 0?

Selected features for a given threshold value
Thresholding ridge coefficients?

Let's look at two related features...

Nothing measuring bathrooms was included!

Thresholding ridge coefficients?

If only one of the features had been included...
Thresholding ridge coefficients?

Would have included bathrooms in selected model

Can regularization lead directly to sparsity?

Try this cost instead of ridge...

Total cost = \( \text{measure of fit} + \lambda \text{ measure of magnitude of coefficients} \)

\[ \text{RSS}(w) + \lambda \| w \|_1 = |w_0| + \ldots + |w_D| \]

Leads to sparse solutions!

Lasso regression (a.k.a. L_1 regularized regression)
Lasso regression: $L_1$ regularized regression

Just like ridge regression, solution is governed by a continuous parameter $\lambda$

$$\text{RSS}(w) + \lambda ||w||_1$$

tuning parameter = balance of fit and sparsity

If $\lambda = 0$: $\hat{\omega}_{\text{lasso}} = \hat{\omega}_{\text{LS}}$ (unreg. soln.)

If $\lambda = \infty$: $\hat{\omega}_{\text{lasso}} = 0$

If $\lambda$ in between: $0 \leq ||\hat{\omega}_{\text{lasso}}|| \leq ||\hat{\omega}_{\text{LS}}||$

Coefficient path – ridge
Coefficient path – lasso

Fitting the lasso regression model (for given $\lambda$ value)
How we optimized past objectives

To solve for $\hat{w}$, previously took gradient of total cost objective and either:

1) Derived closed-form solution
2) Used in gradient descent algorithm

Optimizing the lasso objective

Lasso total cost: $\text{RSS}(w) + \lambda \sum |w_j|$

Issues:

1) What’s the derivative of $|w_j|$?

2) Even if we could compute derivative, no closed-form solution

---

gradients $\rightarrow$ subgradients

can use subgradient descent
Aside 1: Coordinate descent

Coordinate descent

Goal: Minimize some function \( g \)

\[ g(\mathbf{w}) = g(w_0, w_1, \ldots, w_d) \]

Often, hard to find minimum for all coordinates, but easy for each coordinate

Coordinate descent:

Initialize \( \mathbf{w} = 0 \) (or smartly..)

while not converged

pick a coordinate \( j \)

\[ \mathbf{w}_j \leftarrow \min_{\mathbf{w}} g(\mathbf{w}) \]
Comments on coordinate descent

How do we pick next coordinate?
- At random ("random" or "stochastic" coordinate descent), round robin, ...

**No stepsize** to choose!

Super useful approach for many problems
- Converges to optimum in some cases (e.g., "strongly convex")
- Converges for lasso objective

Aside 2: Normalizing features
Normalizing features

Scale training columns (not rows!) as:

\[ h_j(x_k) = \frac{h_j(x_k)}{\sqrt{\sum_{i=1}^{N} h_j(x_i)^2}} \]

Apply same training scale factors to test data:

\[ h_j(x_k) = \frac{h_j(x_k)}{\sqrt{\sum_{i=1}^{N} h_j(x_i)^2}} \]

Aside 3: Coordinate descent for unregularized regression (for normalized features)
Optimizing least squares objective one coordinate at a time

\[
\text{RSS}(w) = \sum_{i=1}^{N} \left( y_i - \sum_{j=0}^{D} w_j h_j(x_i) \right)^2
\]

Fix all coordinates \( w_{-j} \) and take partial w.r.t. \( w_j \)

\[
\frac{\partial}{\partial w_j} \text{RSS}(w) = -2 \sum_{i=1}^{N} h_j(x_i) (y_i - \sum_{j=0}^{D} w_j h_j(x_i)) \\
= -2 \sum_{i=1}^{N} h_j(x_i) (y_i - \sum_{k=0,k \neq j}^{D} w_k h_k(x_i) - w_j h_j(x_i)) \\
= -2 \sum_{i=1}^{N} h_j(x_i) (y_i - \sum_{k=0,k \neq j}^{D} w_k h_k(x_i)) + 2 w_j \left( \sum_{i=1}^{N} h_j^2(x_i) \right) \\
= -2 \rho_j + 2 w_j
\]

Set partial = 0 and solve

\[
\frac{\partial}{\partial w_j} \text{RSS}(w) = -2 \rho_j + 2 w_j = 0 \\
\hat{w}_j = \rho_j
\]
Coordinate descent for least squares regression

Initialize $\hat{w} = 0$ (or smartly..)
while not converged
for $j=0,1,...,D$

compute: $\rho_j = \sum_{i=1}^{N} h_j(x_i)(y_i - \hat{y}_i(\hat{w}_{-j}))$

set: $\hat{w}_j = \rho_j$

Coordinate descent for lasso
(for normalized features)
Coordinate descent for least squares regression

Initialize $\mathbf{w} = 0$ (or smartly..)
while not converged
for $j=0,1,...,D$
compute: $\rho_j = \sum_{i=1}^{N} h_j(x_i) (y_i - \hat{y}_i(\mathbf{w} - j))$
set: $\mathbf{w}_j = \rho_j$

Coordinate descent for lasso

Initialize $\mathbf{w} = 0$ (or smartly..)
while not converged
for $j=0,1,...,D$
compute: $\rho_j = \sum_{i=1}^{N} h_j(x_i) (y_i - \hat{y}_i(\mathbf{w} - j))$
set: $\mathbf{w}_j = \begin{cases} \rho_j + \lambda/2 & \text{if } \rho_j \leq -\lambda/2 \\ 0 & \text{if } \rho_j \in [-\lambda/2, \lambda/2] \\ \rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 \end{cases}$
Soft thresholding

\[
\hat{w}_j = \begin{cases} 
\rho_j + \lambda/2 & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\
\rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 
\end{cases}
\]

How to assess convergence?

Initialize \( \hat{w} = 0 \) (or smartly..)

while not converged

for \( j = 0,1,...,D \)

compute: \( \rho_j = \sum_{i=1}^{N} h_j(x_i)(y_i - \hat{y}_i(\hat{w} \cdot x_i)) \)

set: \( \hat{w}_j = \begin{cases} 
\rho_j + \lambda/2 & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\
\rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 
\end{cases} \)
Convergence criteria

When to stop?

For convex problems, will start to take smaller and smaller steps

Measure size of steps taken in a full loop over all features
- stop when max step < $\varepsilon$

Other lasso solvers

Classically: Least angle regression (LARS) [Efron et al. ‘04]

Then: Coordinate descent algorithm [Fu ‘98, Friedman, Hastie, & Tibshirani ‘08]

Now:
- Parallel CD (e.g., Shotgun, [Bradley et al. ’11])
- Other parallel learning approaches for linear models
  - Parallel stochastic gradient descent (SGD) (e.g., Hogwild! [Niu et al. ’11])
  - Parallel independent solutions then averaging [Zhang et al. ’12]
- Alternating directions method of multipliers (ADMM) [Boyd et al. ’11]
Coordinate descent for lasso
(for unnormalized features)

Initialize $\hat{w}_0 = 0$ (or smartly...)
while not converged
for $j = 0, 1, \ldots, D$

compute: $\rho_j = \sum_{i=1}^{N} h_j(x_i)(y_i - \hat{y}_i(\hat{w}_j))$

set: $\hat{w}_j = \begin{cases} 
\rho_j + \lambda/2 & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\
\rho_j - \lambda/2 & \text{if } \rho_j > \lambda/2 
\end{cases}$

Coordinate descent for lasso
with normalized features
Coordinate descent for lasso with unnormalized features

Precompute: \( z_j = \sum_{i=1}^{N} h_j(x_i)^2 \)

Initialize \( \hat{w} = 0 \) (or smartly..)

while not converged

for \( j = 0, 1, \ldots, D \)

compute: \( \rho_j = \sum_{i=1}^{N} h_j(x_i)(y_i - \hat{y}_i(\hat{w} \ldots)) \)

set: \( \hat{w}_j = \begin{cases} 
\frac{(\rho_j + \lambda/2)}{z_j} & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\
\frac{(\rho_j - \lambda/2)}{z_j} & \text{if } \rho_j > \lambda/2 
\end{cases} \)

How to choose \( \lambda \)
If sufficient amount of data...

- **Training set**
- **Validation set**
- **Test set**

- fit $\hat{w}_\lambda$
- test performance of $\hat{w}_\lambda$ to select $\lambda^*$
- assess generalization error of $\hat{w}_{\lambda^*}$

Summary for feature selection and lasso regression
Impact of feature selection and lasso

Lasso has changed machine learning, statistics, & electrical engineering

But, for feature selection in general, be careful about interpreting selected features
- selection only considers features included
- sensitive to correlations between features
- result depends on algorithm used
- there are theoretical guarantees for lasso under certain conditions

What you can do now...

- Describe “all subsets” and greedy variants for feature selection
- Analyze computational costs of these algorithms
- Formulate lasso objective
- Describe what happens to estimated lasso coefficients as tuning parameter $\lambda$ is varied
- Interpret lasso coefficient path plot
- Contrast ridge and lasso regression
- Estimate lasso regression parameters using an iterative coordinate descent algorithm
Optimizing lasso objective one coordinate at a time

\[ \text{RSS}(w) + \lambda \|w\|_1 = \sum_{i=1}^{N} (y_i - \sum_{j=0}^{D} w_j h_j(x_i))^2 + \lambda \sum_{j=0}^{D} |w_j| \]

Fix all coordinates \( w_j \) and take partial w.r.t. \( w_j \)

derive without normalizing features
Part 1: Partial of RSS term

\[ \text{RSS}(w) + \lambda \|w\|_1 = \sum_{i=1}^{N} \left( y_i - \sum_{j=0}^{D} w_j h_j(x_i) \right)^2 + \lambda \sum_{j=0}^{D} |w_j| \]

\[ \frac{\partial}{\partial w_j} \text{RSS}(w) = -2 \sum_{i=1}^{N} h_j(x_i) \left( y_i - \sum_{j=0}^{D} w_j h_j(x_i) \right) \]

\[ = -2 \sum_{i=1}^{N} h_j(x_i) \left( y_i - \sum_{j=0}^{D} w_j h_j(x_i) \right) \]

\[ = -2 \sum_{i=1}^{N} h_j(x_i) (y_i - \sum_{j=0}^{D} w_j h_j(x_i)) + 2 w_j \sum_{i=1}^{N} h_j(x_i) \]

\[ = -2 \rho_j^y + 2 w_j \hat{e}_j \]

Part 2: Partial of L_1 penalty term

\[ \text{RSS}(w) + \lambda \|w\|_1 = \sum_{i=1}^{N} \left( y_i - \sum_{j=0}^{D} w_j h_j(x_i) \right)^2 + \lambda \sum_{j=0}^{D} |w_j| \]

\[ \lambda \frac{\partial}{\partial w_j} |w_j| = ??? \]
Subgradients of convex functions

Gradients lower bound convex functions:

\[ g(b) \geq g(a) + \nabla g(a)(b-a) \]

unique at \( x \) if function differentiable at \( x \)

Subgradients: Generalize gradients to non-differentiable points:
- Any plane that lower bounds function

\[ \forall x, \exists \partial g(x) \text{ subgradient of } g \text{ at } x \]

\[ g(b) \geq g(x) + \nabla (b-a) \]

Part 2: Subgradient of \( L_1 \) term

\[
\text{RSS}(w) + \lambda \|w\|_1 = \sum_{i=1}^{N} (y_i - \sum_{j=0}^{D} w_j (x_i))^2 + \lambda \sum_{j=0}^{D} |w_j|
\]

\[
\lambda \partial_{w_j} |w_j| = \begin{cases} 
-\lambda^{-1} & \text{ when } w_j < 0 \\
[-\lambda, \lambda] & \text{ when } w_j = 0 \\
\lambda^{-1} & \text{ when } w_j > 0 
\end{cases}
\]
Putting it all together...

\[
\text{RSS}(w) + \lambda |w|_1 = \sum_{i=1}^{N} (y_i - \sum_{j=0}^{D} w_j h_j(x_i))^2 + \lambda \sum_{j=0}^{D} |w_j|
\]

\[
\partial_{w_j}[\text{lasso cost}] = 2z_j w_j - 2\rho_j + \begin{cases} 
-\lambda & \text{when } w_j < 0 \\
[-\lambda, \lambda] & \text{when } w_j = 0 \\
\lambda & \text{when } w_j > 0 
\end{cases}
\]

Optimal solution:
Set subgradient = 0

\[
\partial_{w_j}[\text{lasso cost}] = \begin{cases} 
2z_j w_j - 2\rho_j - \lambda & \text{when } w_j < 0 \\
[-2\rho_j - \lambda, -2\rho_j + \lambda] & \text{when } w_j = 0 \\
2z_j w_j - 2\rho_j + \lambda & \text{when } w_j > 0 
\end{cases}
= 0
\]

Case 1 \((w_j < 0)\):

\[
2z_j \hat{w}_j - 2\rho_j - \lambda = 0
\]

\[
\hat{w}_j = \frac{2z_j + \lambda}{2\rho_j} = \frac{\hat{w}_j + \frac{\lambda}{2}}{\rho_j}
\]

For \(\hat{w}_j < 0\), need \(\rho_j > -\frac{\lambda}{2}\)

Case 2 \((w_j = 0)\):

\[
\hat{w}_j = 0
\]

For \(\hat{w}_j = 0\), need \([-2\rho_j - \lambda, -2\rho_j + \lambda]\) to contain 0:

\[-2\rho_j - \lambda > 0 \rightarrow \rho_j \leq \frac{\lambda}{2} \quad -2\rho_j - \lambda \leq 0 \rightarrow \rho_j \geq \frac{\lambda}{2} \quad \frac{\lambda}{2} \leq \rho_j \leq \frac{\lambda}{2}\]

Case 3 \((w_j > 0)\):

\[
2z_j \hat{w}_j - 2\rho_j + \lambda = 0
\]

\[
\hat{w}_j = \frac{\rho_j - \frac{\lambda}{2}}{2z_j}
\]

For \(\hat{w}_j > 0\), need \(\rho_j > \frac{\lambda}{2}\)
Optimal solution:
Set subgradient = 0

\[
\partial_{w_j} [\text{lasso cost}] = \begin{cases} 
2z_jw_j - 2\rho_j - \lambda & \text{when } w_j < 0 \\
[-2\rho_j - \lambda, -2\rho_j + \lambda] & \text{when } w_j = 0 \\
2z_jw_j - 2\rho_j + \lambda & \text{when } w_j > 0
\end{cases}
\]

\[\hat{w}_j = \begin{cases} 
\frac{(\rho_j + \lambda/2)}{z_j} & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \in [-\lambda/2, \lambda/2] \\
\frac{(\rho_j - \lambda/2)}{z_j} & \text{if } \rho_j > \lambda/2
\end{cases}\]

Soft thresholding

\[\hat{w}_j = \begin{cases} 
\frac{(\rho_j + \lambda/2)}{z_j} & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \in [-\lambda/2, \lambda/2] \\
\frac{(\rho_j - \lambda/2)}{z_j} & \text{if } \rho_j > \lambda/2
\end{cases}\]
Coordinate descent for lasso

Precompute: \( z_j = \sum_{i=1}^{N} h_j(x_i)^2 \)

Initialize \( \hat{\mathbf{w}} = 0 \) (or smartly...)

while not converged

for \( j=0,1,...,D \)

compute: \( \rho_j = \sum_{i=1}^{N} h_j(x_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_j)) \)

set: \( \hat{w}_j \) =

\[
\begin{cases} 
(\rho_j + \lambda/2)/z_j & \text{if } \rho_j < -\lambda/2 \\
0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\
(\rho_j - \lambda/2)/z_j & \text{if } \rho_j > \lambda/2 
\end{cases}
\]