Simple Variable Selection
LASSO: Sparse Regression

Machine Learning – CSE546
Sham Kakade
University of Washington
October 11, 2016

Announcements:
- HW1 due on Friday.
- Readings: please do them.
- Project Proposals: please start thinking about it!

Today:
- Review: cross validation
- Feature selection
- Lasso
Regularization in Regression

- Overfitting usually leads to very large parameter choices, e.g.:
  \[-2.2 + 3.1 X - 0.30 X^2 \quad \quad -1.1 + 4,700,910.7 X - 8,585,638.4 X^2 + \ldots\]

- Regularization: or “Shrinkage” procedure

  \[\hat{w}_{\text{ridge}} = \arg\min_w \sum_{j=1}^N \left( t(x_j) - (w_0 + \sum_{i=1}^k w_i h_i(x_j)) \right)^2 + \lambda \sum_{i=1}^k w_i^2\]

- How do we pick the regularization constant \(\lambda??\) (and pick models?)
  - We could use the test set? Or another hold out set?
(LOO) Leave-one-out cross validation

- Consider a validation set with 1 example:
  - $D$ – training data
  - $D_j$ – training data with $j$th data point moved to validation set
- Learn classifier $h_{D_j}$ with $D_j$ dataset
- Estimate true error as squared error on predicting $t(x_j)$:
  - Unbiased estimate of $\text{error}_{\text{true}}(h_{D_j})$!
  - Seems really bad estimator, but wait!
- **LOO cross validation**: Average over all data points $j$:
  - For each data point you leave out, learn a new classifier $h_{D_j}$
  - Estimate error as:
    \[
    \text{error}_{\text{LOO}} = \frac{1}{N} \sum_{j=1}^{N} \left( t(x_j) - h_{D \setminus j}(x_j) \right)^2
    \]

LOO cross validation is (almost) unbiased estimate of true error of $h_D$!

- When computing LOOCV error, we only use $N$-1 data points
  - So it's not estimate of true error of learning with $N$ data points!
  - Usually pessimistic, though – learning with less data typically gives worse answer
- LOO is “almost” unbiased!
  - Asymptotically (for large $N$), under some conditions.
  - It is reasonable to use in practice.
  - Great news: Use LOO error for model selection!! (e.g., picking $\lambda$)
- LOO is computationally costly! (exception: see HW)
  - You have to run your algorithm $N$ times.
  - Practice: “K-fold” cross validation
What you need to know…

- Use cross-validation to choose parameters
  - Leave-one-out is usually the best, but it is slow…
  - use k-fold cross-validation
Sparsity

- Vector $\mathbf{w}$ is sparse, if many entries are zero:

  - Very useful for many tasks, e.g.,
    - **Efficiency**: If $\text{size}(\mathbf{w}) = 100\text{B}$, each prediction is expensive:
      - If part of an online system, too slow
      - If $\mathbf{w}$ is sparse, prediction computation only depends on number of non-zeros
    - **Interpretability**: What are the relevant dimension to make a prediction?
      - E.g., what are the parts of the brain associated with particular words?

  - But computationally intractable to perform “all subsets” regression

---

Simple greedy model selection algorithm

- Pick a dictionary of features
  - e.g., polynomials for linear regression

- Greedy heuristic:
  - Start from empty (or simple) set of features $F_0 = \emptyset$
  - Run learning algorithm for current set of features $F_t$
    - Obtain weights for these features
  - Select **next best feature** $h_i(x)^*$
    - e.g., $h_i(x)$ that results in lowest training error learner when using $F_t \cup \{h_i(x)^*\}$
  - $F_{t+1} \leftarrow F_t + \{h_i(x)^*\}$
  - Recurse
Greedy model selection

- Applicable in many other settings:
  - Considered later in the course:
    - Logistic regression: Selecting features (basis functions)
    - Naïve Bayes: Selecting (independent) features \( P(X_i|Y) \)
    - Decision trees: Selecting leaves to expand

- Only a heuristic!
  - Finding the best set of \( k \) features is computationally intractable!
  - Sometimes you can prove something strong about it…

- There are many more elaborate methods out there

When do we stop???

- Greedy heuristic:
  - ... 
  - Select next best feature \( X_i^* \)
    - E.g. \( h(x) \) that results in lowest training error learner when using \( F_i + \{h(x)^*\} \)

- Recurse

  When do you stop???
  - When training error is low enough?
  - When test set error is low enough?
  - Using cross validation?
Regularization in Linear Regression

- Overfitting usually leads to very large parameter choices, e.g.:
  - $-2.2 + 3.1 X - 0.30 X^2$
  - $-1.1 + 4,700,910.7 X - 8,585,638.4 X^2 + \ldots$

- Regularized or penalized regression aims to impose a “complexity” penalty by penalizing large weights
  - “Shrinkage” method

Variable Selection by Regularization

- Ridge regression: Penalizes large weights

- What if we want to perform “feature selection”?  
  - E.g., Which regions of the brain are important for word prediction?  
  - Can’t simply choose features with largest coefficients in ridge solution

- Try new (convex) penalty: Penalize non-zero weights
  - Regularization penalty:
    - Leads to sparse solutions
    - Just like ridge regression, solution is indexed by a continuous param $\lambda$
    - Major impact in: statistics, machine learning & electrical engineering
LASSO Regression

- **LASSO**: least absolute shrinkage and selection operator
- New objective:

\[
\hat{w}_{\text{LASSO}} = \arg \min_w \sum_{j=1}^N \left( t(x_j) - \left( w_0 + \sum_{i=1}^k w_i h_i(x_j) \right) \right)^2 + \lambda \sum_{i=1}^k |w_i|
\]

(Related) Constrained Optimization

- LASSO solution:
Geometric Intuition for Sparsity

Ridge Regression

Lasso

From Rob Tibshirani slides

Optimizing the LASSO Objective

LASSO solution:

\[ \hat{w}_{\text{LASSO}} = \arg \min_w \sum_{j=1}^{N} \left( t(x_j) - \left( w_0 + \sum_{i=1}^{k} w_i h_i(x_j) \right) \right)^2 + \lambda \sum_{i=1}^{k} |w_i| \]
Coordinate Descent

- Given a function F
  - Want to find minimum

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

- Coordinate descent:
  - How do we pick next coordinate?

- Super useful approach for *many* problems
  - Converges to optimum in some cases, such as LASSO

---

Optimizing LASSO Objective

One Coordinate at a Time

$$\sum_{j=1}^{N} \left( t(x_j) - \left( w_0 + \sum_{i=1}^{k} w_i h_i(x_j) \right) \right)^2 + \lambda \sum_{i=1}^{k} |w_i|$$

- Taking the derivative:
  - Residual sum of squares (RSS):
    $$\frac{\partial}{\partial w_\ell} RSS(w) = -2 \sum_{j=1}^{N} h_\ell(x_j) \left( t(x_j) - \left( w_0 + \sum_{i=1}^{k} w_i h_i(x_j) \right) \right)$$
  - Penalty term:
Subgradients of Convex Functions

- Gradients lower bound convex functions:

- Gradients are unique at \( \mathbf{w} \) iff function differentiable at \( \mathbf{w} \)

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

Taking the Subgradient

- Gradient of RSS term:
  \[
  \frac{\partial}{\partial \mathbf{w}_\ell} \text{RSS}(\mathbf{w}) = a_\ell \mathbf{w}_\ell - c_\ell
  \]
  \[
  a_\ell = 2 \sum_{j=1}^{N} (h_\ell(x_j))^2
  c_\ell = 2 \sum_{j=1}^{N} h_\ell(x_j) \left( t(x_j) - (w_0 + \sum_{i \neq \ell} w_i h_i(x_j)) \right)
  \]

- If no penalty:
- Subgradient of full objective:
Setting Subgradient to 0

\[ \partial_{w_{\ell}} F(w) = \begin{cases} 
-\alpha_{\ell} w_{\ell} - c_{\ell} - \lambda & w_{\ell} < 0 \\
[-c_{\ell} - \lambda, -c_{\ell} + \lambda] & w_{\ell} = 0 \\
\alpha_{\ell} w_{\ell} - c_{\ell} + \lambda & w_{\ell} > 0 
\end{cases} \]

Soft Thresholding

\[ \hat{w}_{\ell} = \begin{cases} 
(c_{\ell} + \lambda)/\alpha_{\ell} & c_{\ell} < -\lambda \\
0 & c_{\ell} \in [-\lambda, \lambda] \\
(c_{\ell} - \lambda)/\alpha_{\ell} & c_{\ell} > \lambda 
\end{cases} \]

From Kevin Murphy textbook
Coordinate Descent for LASSO (aka Shooting Algorithm)

- Repeat until convergence
  - Pick a coordinate $l$ at (random or sequentially)
    - Set: $\hat{w}_l = \begin{cases} 
    (c_l + \lambda)/a_l & c_l < -\lambda \\
    0 & c_l \in [-\lambda, \lambda] \\
    (c_l - \lambda)/a_l & c_l > \lambda 
  \end{cases}$
    - Where: $a_l = 2 \sum_{j=1}^{N} (h_l(x_j))^2$
    $c_l = 2 \sum_{j=1}^{N} h_l(x_j) \left( t(x_j) - (w_0 + \sum_{i \neq l} w_i h_i(x_j)) \right)$

- For convergence rates, see Shalev-Shwartz and Tewari 2009

- Other common technique = LARS
- Least angle regression and shrinkage, Efron et al. 2004

Recall: Ridge Coefficient Path

- Typical approach: select $\lambda$ using cross validation

From Kevin Murphy textbook
Now: LASSO Coefficient Path

What you need to know

- Variable Selection: find a sparse solution to learning problem
- \( L_1 \) regularization is one way to do variable selection
  - Applies beyond regression
  - Hundreds of other approaches out there
- LASSO objective non-differentiable, **but convex** ➔ Use subgradient
- No closed-form solution for minimization ➔ Use coordinate descent
- Shooting algorithm is simple approach for solving LASSO