Advances in Regularization:
Bridge Regression and Coordinate Descent Algorithms

Giovanni Seni, Ph.D.

Intuit – Applied Data Sciences Group
Giovanni_Seni@intuit.com

Santa Clara University
GSeni@scu.edu

Mountain View, CA – June 4, 2012

Overview

- In a Nutshell & Timeline
- Predictive Learning
- Regularization & Bridge Regression
- Path Finding by Generalized Gradient Descent
- Example: ~1M predictors

Regularization In a Nutshell

- What is *regularization*?
  - "any part of model building which takes into account – implicitly or explicitly – the finiteness and imperfection of the data and the limited information in it, which we can term 'variance' in an abstract sense" [Rosset, 2003]
- Forms of regularization
  1. Explicit via constraints on model complexity
  2. Implicit through incremental building of the model
  3. Choice of robust loss functions

Timeline

- Garotte (Breiman, 1995)
- Lasso (Tibshirani, 1996)
- LARS (Efron et al., 2004)
- Path Seeker (Friedman, 2004)
- Elastic Net (Zou, Hastie, 2005)
- GLMs via Coordinate Descent (Friedman et al., 2008)
- Generalized Path Seeker (Friedman, 2008)
- Penalized Matrix Decomposition (Witten et al., 2010)
- Bayesian Lasso (Hu, Rajaratnam, 2012)

Overview

- In a Nutshell & Timeline
- ➢ Predictive Learning
  - Procedure Summary
  - Model Complexity
- Regularization & Bridege Regression
- Path Finding by Generalized Gradient Descent
- Example: ~1M predictors

Predictive Learning
Procedure Summary

- Given "training" data $D = \{y_i, x_{i1}, x_{i2}, \cdots, x_{in}\}_1^N = \{y_i, \mathbf{x}_i\}_1^N$
  - D is a random sample from some unknown (joint) distribution
- Build a functional model $\hat{y} = \hat{F}(x_1, x_2, \cdots, x_n) = \hat{F}(\mathbf{x})$
  - Offers *adequate* and *interpretable* description of how the inputs affect the outputs
  - Parsimony is an important criterion: simpler models are preferred for the sake of scientific insight into the $\mathbf{x}$ - y relationship
- Need to specify: $<$ model, score criterion, search strategy $>$

Predictive Learning
Procedure Summary (2)

- *Model*: underlying functional form sought from data

  $\hat{F}(\mathbf{x}) = \hat{F}(\mathbf{x};\mathbf{a}) \in \mathcal{F}$ family of functions indexed by $\mathbf{a}$

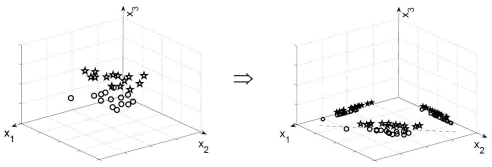
- *Score criterion*: judges (lack of) quality of fitted model
  - Loss function $L(y, \hat{F})$: penalizes individual errors in prediction
  - Risk $R(\mathbf{a}) = E_{y,\mathbf{x}} L(y, \hat{F}(\mathbf{x};\mathbf{a}))$: the expected loss over all predictions
- *Search Strategy*: minimization procedure of score criterion

  $\mathbf{a}^* = \arg\min_{\mathbf{a}} R(\mathbf{a})$

Predictive Learning
Procedure Summary (3)

- *"Surrogate" Score criterion*:
  - Training data: $\{y_i, \mathbf{x}_i\}_1^N \sim p(\mathbf{x}, y)$
  - $p(\mathbf{x}, y)$ unknown $\Rightarrow$ $\mathbf{a}^*$ unknown
  - $\Rightarrow$ Use approximation: *Empirical Risk*
    - $\bar{R}(\mathbf{a}) = \frac{1}{N}\sum_{i=1}^{N} L(y, \hat{F}(\mathbf{x}_i;\mathbf{a}))$ $\Rightarrow$ $\bar{\mathbf{a}} = \arg\min_{\mathbf{a}} \bar{R}(\mathbf{a})$
    - If not $N >> n$, $R(\bar{\mathbf{a}}) >> R(\mathbf{a}^*)$

Predictive Learning
What is the "right" size of a model?



- Dilemma
  - If model (# of variables) is too small, then approximation is too crude (bias) $\Rightarrow$ increased errors
  - If model is too large, then it fits the training data too closely (overfitting, increased variance) $\Rightarrow$ increased errors

Overview

- In a Nutshell & Timeline
- Predictive Learning
- ➢ Regularization
  - Linear Regression
  - "Constrained" vs. "Penalized" formulation
  - Coefficient Paths and Model Selection
  - Complexity Penalties
  - Bridge Regression
- Path Finding by Generalized Gradient Descent
- Example: ~1M predictors

Linear Regression
Overview

- Linear model: $F(\mathbf{x}) = a_0 + \sum_{j=1}^{n} a_j x_j$
- Standard coefficient estimation criterion (OLR):

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{a}} \sum_{i=1}^{N} L\left(y_i,\ a_0 + \sum_{j=1}^{n} a_j x_{ij}\right) \qquad \text{E.g., } \hat{\mathbf{a}} = (\mathbf{X}^t\mathbf{X} + \varepsilon\mathbf{I})^{-1}\mathbf{X}^t\mathbf{y}$$

- OLR often unsatisfactory:
  - Prediction accuracy: high variance in coefficient estimates
  - Interpretation: desire for a smaller subset of predictors that exhibit the strongest effects
    - *Subset Selection*: can be extremely variable because of its discrete process
    - *Regularized Regression*: continuous process often preferred

Regularized Linear Regression
Constrained Formulation

- Augmented coefficients estimation criterion:

$$\hat{\mathbf{a}} = \{\hat{a}_j\}_0^n = \arg\min_{\{a_j\}} \sum_{i=1}^{N} L\left(y_i,\ a_0 + \sum_{j=1}^{n} a_j x_{ij}\right) \text{ s.t. } P(\mathbf{a}) \le t$$

- "Constraining" function $P(\mathbf{a})$:
  - Non-negative
  - $0 < t < P(\hat{\mathbf{a}})$: bias-variance tradeoff
  - Deterministic and independent of the particular random sample
    - $\Rightarrow$ provides a stabilizing influence on the criterion being minimized
  - Best $P(\mathbf{a})$ requires knowledge of $\mathbf{a}^*$
    - E.g., $\mathbf{a} \approx \mathbf{a}^* \Rightarrow sparsity(\mathbf{a}) \approx sparsity(\mathbf{a}^*)$

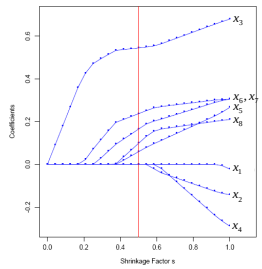
Regularized Linear Regression
Penalized Formulation

- Equivalent penalized formulation:

$$\hat{\mathbf{a}} = \{\hat{a}_j\}_0^n = \arg\min_{\{a_j\}} \sum_{i=1}^{N} L\left(y_i,\ a_0 + \sum_{j=1}^{n} a_j x_{ij}\right) + \lambda \cdot P(\mathbf{a}) \qquad (1)$$

- $\infty \geq \lambda \geq 0 \ \sim \ 0 < t < P(\hat{\mathbf{a}})$
- $P(\mathbf{a})$ penalizes for the increased variance associated with more complex model
- Coefficient "paths" $\hat{\mathbf{a}}(\lambda)$:
  - For each value of λ, we have a different solution to (1)
  - $\lambda = 0 \ \Rightarrow$ OLR solution
  - $\lambda = \infty \ \Rightarrow \ \{\hat{a}_j\}_1^n = 0;\ \ \hat{a}_0 = \arg\min_{\{a\}} \sum_{i=1}^{N} L(y_i,\ a)$

Regularized Linear Regression
Coefficient Paths



- Shrinkage factor: $s \approx \lambda_{\min} / \lambda$

Regularized Linear Regression
Model Selection

- Given $L(y, \bar{y})$ and $P(\mathbf{a})$:

$$\hat{\lambda} = \arg\min_{0 \leq \lambda \leq \infty} \tilde{R}(\hat{\mathbf{a}}(\lambda)) = \arg\min_{\mathbf{a}} \left[\tilde{R}(\mathbf{a}) + \lambda \cdot P(\mathbf{a})\right]$$

- Selected model: $\hat{\mathbf{a}}(\hat{\lambda})$
- Cross-validation often used on a predefined grid in $[\lambda_{\min}, \lambda_{\max}]$
- Challenge: rapidly produce paths without repeatedly optimizing

  $\Rightarrow$ Direct Path Seeking algorithms
  - Forward Stagewise (Hastie et al., 2001), LARS (Efron et al., 2004) , Path Seeker (Friedman, 2005), Coordinate Descent (Friedman, 2008)

Regularized Linear Regression
Complexity Penalties

- Ridge: $P(\mathbf{a}) = \sum_{j=1}^{n} a_j^2$
  - Shrinks coefficients towards 0
  - "Dense" solutions
  - Best for *large number of small effects*
  - k identical predictors $\Rightarrow$ each gets identical coefficient 1/kth the size
- Lasso: $P(\mathbf{a}) = \sum_{j=1}^{n} |a_j|$
  - "Sparse" solutions – i.e., does variable selection
  - Best for *small to moderate number of moderate-size effects*
  - Somewhat indifferent to very correlated predictors; will tend to pick one and ignore the rest

    … up to a limit: extreme correlations cause instability

Bridge Regression
Overview

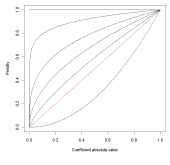
- Degree of Sparsity: $S(\mathbf{a}) = \#(a_j \cong 0)/n$
  - $S(\mathbf{a}) \cong 0 \Rightarrow \mathbf{a}$ is dense $S(\mathbf{a}) \cong 1 \Rightarrow \mathbf{a}$ is sparse
- We expect $\tilde{\mathbf{a}}(\lambda^*) \approx \mathbf{a}^*$ implies $S(\tilde{\mathbf{a}}(\lambda^*)) \approx S(\mathbf{a}^*)$
  - Choose a penalty that produces solutions $\tilde{\mathbf{a}}(\lambda)$ with sparsity similar to that of $\mathbf{a}^*$
  - Sparsity of $\mathbf{a}^*$ is unknown $\Rightarrow$ define family of penalties $P_\alpha(\mathbf{a})$
- Jointly estimate α (sparsity) and λ (shrinkage):
  - $(\tilde{\alpha}, \tilde{\lambda}) = \arg\min_{\alpha,\lambda} \left[\bar{R}(\tilde{\mathbf{a}}_\alpha(\lambda))\right]$; $\hat{\mathbf{a}}_\alpha(\lambda) = \arg\min_{\mathbf{a}} \left[\bar{R}(\mathbf{a}) + \lambda \cdot P_\alpha(\mathbf{a})\right]$

Bridge Regression
Penalties

- Convex constraints
  - $P_\alpha(\mathbf{a}) = (1-\alpha)\frac{1}{2}\|\mathbf{a}\|_{l2}^2 + \alpha\|\mathbf{a}\|_{l1}$ (Elastic Net)
  - "bridges" lasso ⇔ ridge
    - $\alpha = 0$: ridge-regression (dense)
    - $\alpha = 1$: lasso (sparse)
      - Often $\alpha = 1 - \varepsilon$ preferred
    - Allows searching for a compromise between these two penalties
  - Model selection to jointly estimate α (sparsity) and λ (shrinkage)

Bridge Regression
Penalties (2)

- Non-Convex constraints
  - $P_\alpha(\mathbf{a}) = \sum_{j=1}^{n} |a_j|^\alpha$ (Power Family)
    - $\alpha = 0$: all-subsets regression (sparsest)
    - $\alpha = 1$: lasso (sparse)
    - $\alpha = 2$: ridge-regression (dense)



  - For $\alpha < 1$, $P_\alpha(\mathbf{a})$ is non-convex

    ⇒ Path Finding by Generalized Gradient Descent

Overview

- In a Nutshell & Timeline
- Predictive Learning
- Model Complexity & Regularization
- Regularized Linear Regression
- ➢ Path Finding by Generalized Gradient Descent
  - Coordinate Descent Algorithms
  - Least-Squares/Elastic-Net Case
- Example: ~1M predictors

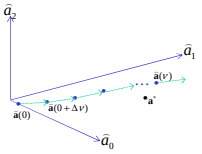
Path Finding by Generalized Gradient Descent
Overview

- One way to define a coefficient path:
  i. Specify a starting and an ending point for the path
     e.g., $\tilde{\mathbf{a}}(\lambda=\infty)=0,\ \tilde{\mathbf{a}}(\lambda=0)=\tilde{\mathbf{a}}^{OLR}$
  ii. Given any point on the path $\tilde{\mathbf{a}}(\nu)$, have a prescription defining the next point $\tilde{\mathbf{a}}(\nu+\Delta\nu)$
     e.g., $\tilde{\mathbf{a}}(\nu+\Delta\nu)=\tilde{\mathbf{a}}(\nu)+\Delta\nu\cdot\mathbf{d}(\nu)$
     - $\mathbf{d}(\nu)$: vector characterizing a direction in the parameter space
     - $\Delta\nu$: specified distance along that direction
- Methods differ for $\mathbf{d}(\nu)$, $\Delta\nu$
- All share *monotonicity property*: $\tilde{R}(\tilde{\mathbf{a}}(\nu+\Delta\nu))<\tilde{R}(\tilde{\mathbf{a}}(\nu))$

Path Finding by Generalized Gradient Descent
Overview (2)

- Algorithm:

Initialize $\nu=0;\ \mathbf{a}(\nu)=0$
Loop {
  // Get next path point
  $\hat{\boldsymbol{a}}(\nu+\Delta\nu)=\hat{\boldsymbol{a}}(\nu)+\mathbf{d}(\nu)\cdot\Delta\nu$
  // Increment path length
  $\nu\leftarrow\nu+\Delta\nu$
}
Until $(\tilde{R}(\hat{\boldsymbol{a}}(\nu))$ is min$)$



- Sample direction vector: $\mathbf{d}(\nu)=\{g_j(\nu)\}_0^n$
  where $g_j(\nu)=-\left[\frac{\partial\tilde{R}(\mathbf{a})}{\partial a_j}\right]_{\mathbf{a}=\mathbf{a}(\nu)}$

Path Finding by Generalized Gradient Descent
Coordinate-wise Descent

- "One-at-a-time" method for minimizing a class of convex functions:
  – Repeat
    - minimize over x_1, keeping $x_2,\ldots,x_n$ fixed
    - minimize over x_2, keeping $x_1,x_3,\ldots,x_n$ fixed
    - ...
- Computationally attractive when each coordinate minimization can be done quickly
- Class of functions where method works [Tseng, 2001]:
  $f(\mathbf{a})=g(\mathbf{a})+\sum_{j=1}^n h_j(a_j)$, where $g(\cdot)$ is differentiable and convex and the $h_j(\cdot)$ are convex

Coordinate-Wise Descent
$L(\cdot)$: least-squares; $P(\mathbf{a})$: elastic-net

- Recall: $\tilde{R}(\mathbf{a};\alpha,\lambda)=\frac{1}{N}\sum_{i=1}^N\left(y_i-a_0-\mathbf{x}_i'\mathbf{a}\right)^2+\lambda\cdot\sum_{j=1}^n\left[\frac{1}{2}(1-\alpha)\cdot a_j^2+\alpha\cdot|a_j|\right]$
  Risk Penalty
- Suppose we have estimates for $\tilde{a}_0$ and $\tilde{a}_l$; $l\neq j$, and wish to optimize with respect to a_j
- Need $g_j=-\left[\frac{\partial\tilde{R}}{\partial a_j}\right]_{\mathbf{a}=\tilde{\mathbf{a}}}$
  – Case $\tilde{a}_j>0$: $g_j=-\frac{1}{N}\sum_{i=1}^N x_{ij}\left(y_i-\tilde{a}_0-\sum_{k\neq j}x_{ik}\tilde{a}_k\right)+a_j+\lambda(1-\alpha)a_j+\lambda\alpha$
  – Case $\tilde{a}_j<0$: similar

Can be shown that, $g_j=0\Rightarrow\tilde{a}_j\leftarrow\frac{S\left(\frac{1}{N}\sum_{i=1}^N x_{ij}(y_i-\tilde{y}_i^{(j)}),\ \lambda\alpha\right)}{1+\lambda(1-\alpha)}$

Coordinate-Wise Descent

$L(\cdot)$: least-squares; $P(\mathbf{a})$: elastic-net (2)

- Efficient updates
  - Note that $\frac{1}{N}\sum_{i=1}^{N} x_{ij}(y_i - \tilde{y}_i^{(j)}) = \frac{1}{N}\sum_{i=1}^{N} x_{ij}(r_i + x_{ij}\tilde{a}_j) = \tilde{a}_j + \frac{1}{N}\sum_{i=1}^{N} x_{ij} r_i$
  - And $\sum_{i=1}^{N} x_{ij} r_i = \sum_{i=1}^{N} x_{ij}(y_i - \bar{y}_i)$

$$= < x_j, y > - \sum_{i=1}^{N} x_{ij} \cdot \left(\sum_{k:|\tilde{a}_k|>0} x_{ik}\tilde{a}_k \right)$$

$$= < x_j, y > - \sum_{k:|\tilde{a}_k|>0} < x_j, x_k > \tilde{a}_k$$

  - Compute and store inner products $< x_j, y >$
  - First time a variable x_j enters the model, compute and store $< x_j, x_k >$
  - Procedure stops after cycle with no new variable entering the model

Coordinate-Wise Descent

$L(\cdot)$: least-squares; $P(\mathbf{a})$: elastic-net (3)

- λ sequence
  - $\lambda_{\max}$: smallest λ for which $\tilde{\mathbf{a}} = 0$
    - $\tilde{a}_j$ will stay zero if $\frac{1}{N}|< x_j, y >| < \lambda\alpha \quad \Rightarrow \quad \lambda_{\max} = \frac{1}{N\alpha}\max_i |< x_i, y >|$
  - $\lambda_{\min} = \varepsilon \cdot \lambda_{\max}$
  - Sequence of K values in $[\lambda_{\min}, \lambda_{\max}]$ is constructed
- α sequence
  - Smaller sequence in $[0, 1-\varepsilon]$
- Active set – $\{\tilde{a}_k \neq 0\}$
  - Iterate on this set until convergence; then one more pass…
  - Stop if active set does not change

Overview

- In a Nutshell & Timeline
- Predictive Learning
- Model Complexity & Regularization
- Regularized Linear Regression
- Coordinate Descent Algorithms

➢ Example: ~1M predictors

Example

~1M Predictors

- Document classification task
  - "Bag of words" representation
  - Feature vector for each document is very sparse
  - R session:

```
load("Data/NewsGroup.RData")
attributes(NewsGroup)
[1] "x" "y"

x <- NewsGroup$x; dim(x)
[1]  11314 777811

length(which(x != 0))/(1.0*nrow(x)*ncol(x))
[1] 0.0005456915

y <- NewsGroup$y; length(y)
[1] 11314

summary(as.factor(y))
  -1    1
5420 5894
```

Example
~1M Predictors (2)

```
system.time(fit.news <- glmnet(x, y, family ="binomial")
   user  system elapsed
  65.88    3.46   69.96

attributes(fit.news)
 [1] "a0"         "beta"       "df"         "dim"        "lambda"
 [6] "dev.ratio"  "nulldev"    "npasses"    "jerr"       "offset"
[11] "classnames" "call"       "nobs"

length(fit.news$lambda)
[1] 100
dim(fit.news$beta)
[1] 777811    100

coefficients <- as.vector(coef(fit.news, s = 0.01))
length(coefficients)
[1] 777812

length(which(abs(coefficients) > 0))
[1] 846
```


Example
~1M Predictors (3)

```
system.time(fit.news <- cv.glmnet(x, y,family ="binomial"))
   user  system elapsed
 638.30   31.61  672.24

attributes(fit.news)
 [1] "lambda"     "cvm"        "cvsd"       "cvup"       "cvlo"
 [6] "nzero"      "name"       "glmnet.fit" "lambda.min" "lambda.1se"

# coefficients with best lambda
coefficients <- as.vector(coef(fit.news, s = fit.news$lambda.1se))
length(which(abs(coefficients) > 0))
[1] 4848

yHat <- predict(fit.news, x, s = fit.news$lambda.1se, type='class')[,1]
table(y, yHat)
    yHat
y       1    2
  -1 5142  278
  1    61 5833
```


Conclusions

- "Bridge Regression" allows for sparsity and shrinkage control
- New very fast algorithms for GLMs parameter estimation with convex penalties
  - Allows various loss-constraint combinations
    E.g., linear regression, logistic regression, multinomial regression
- Algorithm also available for non-convex penalties
- Speed of methods allow handling of very large problems
  - Ideally suited for sparse data