Logistic Regression with Lasso Penalty

Guangwei Weng

University of Minnesota

wengx076@umn.edu

April 27, 2017

Background and Application of Lasso

As datasets grow wide, cases of large p become more usual...

- **Document classification:** bag-of-words can leads to $p = 20K$ features and $N = 5K$ documents.
- **Image classfication:** features are $p = 65K$ pixels.
- **Genomics:** $p = 40K$ genes.

Regularization and Variable Selection are needed.

Existing Algorithms for Lasso

Objective function:

- Linear Regression: $f(\beta) = \frac{1}{2N} RSS(\beta) + \lambda \sum_{j=1}^{p} d_j |\beta_j|$.
- Logistic Regression: $f(\beta) = -\frac{1}{N} l(\beta) + \lambda \sum_{j=1}^{p} d_j |\beta_j|$.

Competitors in solving Lasso:

- lars Using LARS algorithm proposed by Efron to solve least square problems.
- glmnet Fortan based R package using coordinate descent.
- l1logreg Lasso-logistic regression package by Koh, Kim and Boyd, using state-of-art interior point methods.
- BBR Bayesian binomial regression package by Genkin, Lewis and Madigan.

Speed Trials

Linear Regression — Dense Features

	Average Correlation between Features					
	0	0.1	0.2	0.5	0.9	0.95
	$N = 5000,\ p = 100$					
glmnet	0.05	0.05	0.05	0.05	0.05	0.05
lars	0.29	0.29	0.29	0.30	0.29	0.29
	$N = 100,\ p = 50000$					
glmnet	2.66	2.46	2.84	3.53	3.39	2.43
lars	58.68	64.00	64.79	58.20	66.39	79.79

Timings (secs) for `glmnet` and `lars` algorithms for linear regression with lasso penalty. Total time for 100 λ values, averaged over 3 runs.

Speed Trials

Logistic Regression — Sparse Features

	0	0.1	0.2	0.5	0.9	0.95
			$N = 10,000,\ p = 100$			
glmnet	3.21	3.02	2.95	3.25	4.58	5.08
BBR	11.80	11.64	11.58	13.30	12.46	11.83
l1lognet	45.87	46.63	44.33	43.99	45.60	43.16
			$N = 100,\ p = 10,000$			
glmnet	10.18	10.35	9.93	10.04	9.02	8.91
BBR	45.72	47.50	47.46	48.49	56.29	60.21
l1lognet	130.27	124.88	124.18	129.84	137.21	159.54

Timings (seconds) for logistic model with lasso penalty and sparse features (95% zeros in X). Total time for ten-fold cross-validation over a grid of 100 λ values.

Newton's method with coordinate descent

Outer Loop: Given current estimated coefficients $\beta^{(k)}$, by Taylor expansion,

$$\begin{aligned}\beta^{(k+1)} &= \arg\min_{\beta}\left\{-\frac{1}{2}(\beta-\beta^{(k)})^T\nabla^2 l(\beta^{(k)})(\beta-\beta^{(k)})\right.\\ &\qquad\left.-\nabla l(\beta^{(k)})^T(\beta-\beta^{(k)})+N\lambda\sum_{j=1}^{p}d_j|\beta_j|\right\}\\ &= \arg\min_{\beta}\left\{\frac{1}{2}\sum_{i=1}^{N}w_i^{(k)}(Z_i^{(k)}-X_i^T\beta)^2+N\lambda\sum_{j=1}^{p}d_j|\beta_j|\right\},\end{aligned}$$

with $p_i^{(k)} = \mathrm{ilogit}(X_i^T\beta^{(k)})$,$w_i^{(k)} = p_i^{(k)}(1-p_i^{(k)})$,
$Z_i^{(k)} = X_i^T\beta^{(k)} + \frac{y_i - p_i^{(k)}}{w_i^{(k)}}$.

Newton's method with coordinate descent

Inner Loop: Begin with $\tilde{\beta} = \beta^{(k)}$ $r_i = Z_i^{(k)} - X_i^T \beta^{(k)}$ and apply coordinate descent.

For $j = 1, \ldots, p, 1 \ldots, p, 1, \ldots, p, \ldots,$ given current value $\tilde{\beta}$,

$$\text{Update} \quad \beta_j^{\text{update}} = S\left(\tilde{\beta}_j + \frac{\sum_{i=1}^N w_i^{(k)} x_{ij} r_i}{\sum_{i=1}^N w_i^{(k)} x_{ij}^2}, \frac{N\lambda d_j}{\sum_{i=1}^N w_i^{(k)} x_{ij}^2}\right)$$

$$\text{Update} \quad r_i = r_i - X_j(\beta_j^{\text{update}} - \tilde{\beta}_j), i = 1, \ldots N$$

until convergence.

Warm Start and Active Set Update

Need to give the solution path for $\lambda = \lambda_1, \lambda_2, \ldots, \lambda_n$.

- Warm start: Let the initial value of β be β_{λ_l} when solving problem for $\lambda = \lambda_{l+1}$.
- Active set update: Let A_l be the set of indexes of non-zero coefficients in $\{\beta_{\lambda_l}\}$. To solve for $\lambda = \lambda_{l+1}$, solve the following problem first,

$$\min_{\beta_{A_l}} -\frac{1}{N} l(\beta_{A_l}; X_{A_l}, y) + \lambda \sum_{j \in A_l} d_j |\beta_j|.$$

Then update active set by adding index of coefficients with non-zero subgradient.

glmlasso

```
glmlasso(X,y,family=c("gaussian","binomial"),
         nlambda=100,minlam=NULL,lambda=NULL,
         penalty.factor=c(0,rep(1,ncol(X)-1)),tol=1e-6)
```

Output: an object of S3 class “glmlasso” mainly containing

- lambda: The lambda sequence used.
- beta.matrix: Solution of β of each lambda.

```
plot.glmlasso(x)
```

for displaying the solution path.

Real Data Examples

Prostate Data

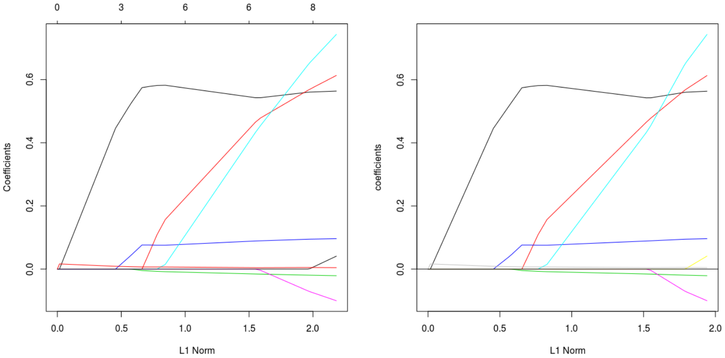

Figure: Soultion path given my glmnet (left) and glmlass (right)

Real Data Examples

South Africa Heart Disease Data

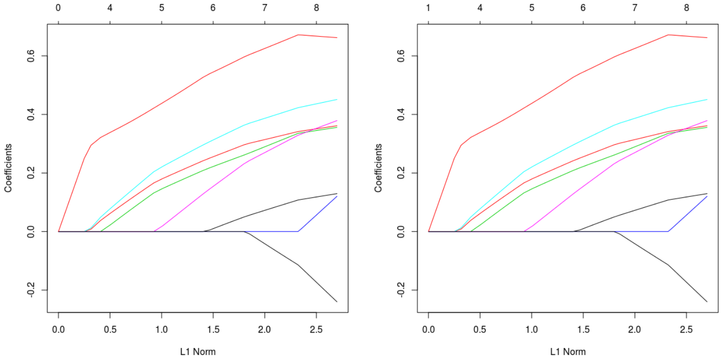

Figure: Soultion path given my glmnet (left) and glmlass (right)

Speed Trial: glmlasso VS glmnet

	N	p	$nlam$	rep	glmlasso	glmnet
Case 1	100	10	100	100	0.872	0.368
Case 2	1000	100	100	3	0.856	0.196
Case 3	100	1000	100	3	1.096	0.072
Case 4	1000	1000	10	1	3.8	0.45
Case 5	1000	1000	1	100	1.512	1.52

Table: Comparison of glmnet and glmlasso over binary data. Time is recorded in seconds

Why?

According to Hastie and Tibshirani...

- Covariance Updates: $\sum_{i=1}^{N} x_{ij} r_i = \langle X_j, y \rangle - \sum_{k:|\beta_k|>0} \langle x_j, x_k \rangle \beta_k$. Cross-covariance terms are computed one for active variables and stored.
- Sparse Updates: If data is sparse, inner products can be computed efficiently.
- Active Set Convergence: After one cycle of p variable, restrict further iterations to the active set till convergence + one more cycle to check if active set has changes.
- FFT:

Why?

According to Hastie and Tibshirani...

- Covariance Updates: $\sum_{i=1}^{N} x_{ij} r_i = \langle X_j, y \rangle - \sum_{k:|\beta_k|>0} \langle x_j, x_k \rangle \beta_k$. Cross-covariance terms are computed one for active variables and stored.
- Sparse Updates: If data is sparse, inner products can be computed efficiently.
- Active Set Convergence: After one cycle of p variable, restrict further iterations to the active set till convergence + one more cycle to check if active set has changes.
- FFT: Friedman + Fortran +Tricks.

Thanks!