Other likelihoods

Patrick Breheny

April 21

Introduction

- In principle, the idea of penalized regression can be extended to any sort of regression model, although of course complications may arise in the details
- In the final lecture for this topic, we'll take a brief look at three additional regression models and their penalized versions, along with any complications that arise:
    - Multinomial regression
    - Robust regression
    - Cox regression

Penalized multinomial regression

- Suppose, instead of a binary response, we have an outcome with multiple possible categories (i.e., that follows a multinomial response, $y_i \in \{1, \ldots, K\}$)
- Extending penalized regression methods to multinomial regression is fairly straightforward:

$$Q(\boldsymbol{\beta}|\mathbf{X}, \mathbf{y}) = -\frac{1}{n} \sum_{k=1}^{K} \sum_{i|y_i=k} \log \pi_{ik} + \sum_{k} P_\lambda(\boldsymbol{\beta}_k),$$

where $\eta_{ki} = \mathbf{x}_i^\top \boldsymbol{\beta}_k$ and

$$\pi_{ik} = \frac{\exp(\eta_{ki})}{\sum_l \exp(\eta_{li})}$$

Model fitting

- Can we use the same algorithm we discussed last time for multinomial models?
- At first glance, it might appear the answer is no, since now we have a $p \times K$ matrix of regression coefficients, and so on
- However, if we adopt the general framework of looping over the response categories, the problem is equivalent to running logistic regression updates for each $\boldsymbol{\beta}_k$:

```
repeat
    for $k = 1, 2, \ldots, K$
        Construct quadratic approximation based on $\widetilde{\boldsymbol{\beta}}_k$
        Coordinate descent to update $\widetilde{\boldsymbol{\beta}}_k$
until convergence
```

Identifiability and reference categories

- For the most part, if you have fit a multinomial regression model before, the penalized version will be familiar
- One noticeable exception, however, is the notion of a reference category
- In traditional multinomial regression, one category must be chosen as a reference group (*i.e.*, have $\boldsymbol{\beta}_k$ set to $\mathbf{0}$) or else the problem is not identifiable

Penalization and identifiability

- With penalized regression, this restriction is not necessary
- For example, suppose $K=2$; then $\beta_{1j}=1, \beta_{2j}=-1$ produces the exact same $\{\pi_{ik}\}$ as $\beta_{1j}=2, \beta_{2j}=0$
- As it is impossible to tell the two models apart (and an infinite range of other models), we cannot estimate $\{\boldsymbol{\beta}_k\}$
- With, say, a ridge penalty, this is no longer the case, as $\sum_k \beta_{jk}^2 = 2$ in the first situation and 4 in the second; the proper estimate is clear
- A similar phenomenon occurs for the lasso penalty, although of course there is now the possibility of sparsity, perhaps with respect to multiple classes

Ramaswamy data

- As an example of such data, consider a study by Ramaswamy et al. (2001)
- The authors collected 198 tumor samples, spanning 14 common tumor types, which they split into training ($n = 144$) and testing ($n = 54$) sets
- Their goal was to develop a method for classifying the samples based on microarray data only, with the clinical goal of aiding in diagnosis, particularly of metastatic or atypical tumors

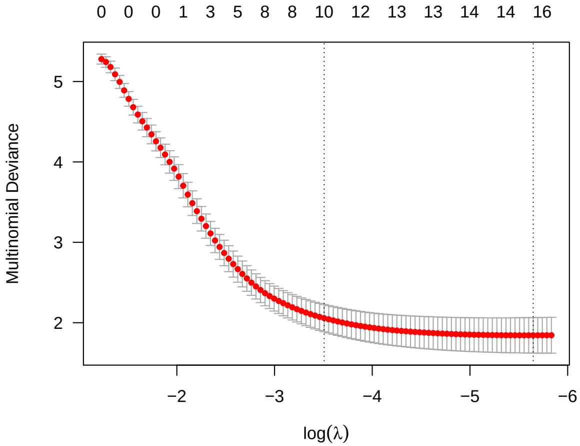
glmnet

- Penalized multinomial regression models are implemented in `glmnet` (but are not yet available in `ncvreg`)
- To use:

```
fit <- glmnet(x, y, family="multinomial")  ## Or:
cvfit <- cv.glmnet(x, y, family="multinomial")
```

- Here, `y` is allowed to be a matrix, a factor, or a vector capable of being coerced into a factor

Cross-validation (training data only)



Coefficients

- The number of selected variables reported at the top of the previous plot is potentially misleading; we have $\widehat{\boldsymbol{\beta}}_1$, $\widehat{\boldsymbol{\beta}}_2$, $\ldots$, and the plot only lists the number of nonzero values in $\widehat{\boldsymbol{\beta}}_1$ (which, here, refers to breast cancer)
- For example, at λ_{CV}, $\widehat{\beta}_{1,5466} = 0.58$, while $\widehat{\beta}_{j,5466} = 0$ for all $j \neq 1$
- This implies that a 1-unit change in gene 5466 doubles the odds of breast cancer ($e^{0.58} = 1.8$), but has no effect on the relative odds of the other tumor types

Ramaswamy results

	Br	Pr	Ln	Cl	Ly	Bl	Ml	Ut	Lk	Rn	Pn	Ov	Ms	CN	
Br	0	0	0	0	0	0	0	0	0	33	0	0	0	0	1
Pr	0	50	0	0	0	0	0	0	0	0	0	25	0	0	4
Ln	0	0	50	0	0	0	0	0	0	0	0	0	0	0	2
Cl	0	0	0	100	0	0	0	0	0	0	33	0	0	0	5
Ly	0	0	0	0	100	0	0	0	0	0	0	0	0	0	6
Bl	25	0	0	0	0	33	0	0	0	0	0	25	0	0	3
Ml	0	0	0	0	0	0	100	0	0	33	0	0	0	0	3
Ut	0	0	0	0	0	0	0	100	0	0	0	0	0	0	2
Lk	0	33	0	0	0	0	0	0	100	0	0	0	0	0	8
Rn	25	0	0	0	0	33	0	0	0	0	33	0	0	0	3
Pn	0	0	0	0	0	0	0	0	0	0	33	0	0	0	1
Ov	50	17	0	0	0	33	0	0	0	0	0	50	0	0	6
Ms	0	0	50	0	0	0	0	0	0	33	0	0	100	25	7
CN	0	0	0	0	0	0	0	0	0	0	0	0	0	75	3
	4	6	4	4	6	3	2	2	6	3	3	4	3	4	54

Overall classification accuracy: 65% (random accuracy: 11%)

Motivation

- Our linear regression models from earlier were based on a least squares loss function, which is appropriate for normally distributed data
- Real data, however, sometimes has thicker tails; least squares estimates tend to be highly influenced by outliers
- One alternative would be to fit a *least absolute deviation* (LAD) model:

$$Q(\boldsymbol{\beta}|\mathbf{X},\mathbf{y}) = \frac{1}{n}\sum_i |y_i - \mathbf{x}_i^\top \boldsymbol{\beta}| + P_\lambda(\boldsymbol{\beta});$$

  the effect is analogous to summarizing a univariate distribution with a median instead of a mean

Model fitting: Complications

- The main challenge introduced by this model is the fact that now the loss function is not differentiable either
- This poses a problem for coordinate descent methods: we didn't go into too much detail about their convergence properties, but it requires separable penalties (OK) and a differentiable loss function (problem)
- Indeed, one can construct counterexamples in which coordinate descent fails for L_1-penalized LAD regression

Huber loss approximation

- One solution is to approximate the absolute value function with a differentiable approximation known as the *Huber loss*, after Peter Huber, who first proposed the idea in 1964:

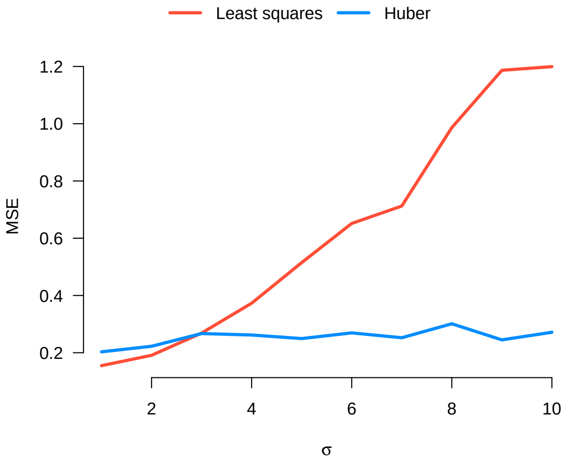
$$L(r_i) = \begin{cases} r_i^2 & \text{if } |r_i| \le \delta \\ \delta(2|r_i| - \delta) & \text{if } |r_i| > \delta \end{cases}$$

- One could either use this idea directly, or continuously relax $\delta \to 0$ to solve the original LAD problem; both of these options are available in the `R` package `hqreg`
- Its usage is very similar to `glmnet`/`ncvreg`, with an option `method=c("huber", "quantile", "ls")` to specify the loss function

Simulation setup

- To see the potential advantages of robust regression, consider the following simple setup: $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, $n = p = 100$, $\beta_1 = \beta_2 = 1$, $\beta_3 = \cdots = \beta_{100} = 0$, $x_{ij} \sim \mathrm{N}(0, 1)$
- However, instead of $\boldsymbol{\varepsilon}$ following a $\mathrm{N}(0, 1)$ distribution, in this simulation it will arise from a mixture distribution:
    - With 90% probability, $\varepsilon_i \sim \mathrm{N}(0, 1)$
    - With 10% probability, the errors are drawn from a noisier distribution, $\varepsilon_i \sim \mathrm{N}(0, \sigma^2)$, where we will be varying σ

Simulation results (lasso penalty used for both)



Remarks

- As you would expect, least squares is optimal for $\sigma = 1$, and still better than Huber loss for slightly messy data
- As the outliers get more extreme, however, Huber loss is clearly superior to least squares
- Both methods get worse as the noise increases, but least squares is much less robust

Cox regression

- A popular model for *time-to-event* data is Cox regression, after Sir David Cox, who proposed the method in 1972
- Suppose individual j dies at time t_j, and that $R(t_j)$ denotes the set of subjects who were *at risk* (i.e., could have died) at time t_j
- The Cox regression model states that the relative likelihood that subject j is the one who died, conditional on the fact that someone died at t_j (this is called a *partial likelihood*) is given by

$$\frac{\exp(\mathbf{x}_j^\top \boldsymbol{\beta})}{\sum_{k \in R(t_j)} \exp(\mathbf{x}_k^\top \boldsymbol{\beta})}$$

Challenge: Nondiagonal $\mathbf{W}$

- The Cox partial likelihood is differentiable, so we can form a quadratic approximation

$$L(\boldsymbol{\beta}) \approx \frac{1}{2n}(\tilde{\mathbf{y}} - \mathbf{X}\boldsymbol{\beta})^\top \mathbf{W}(\tilde{\mathbf{y}} - \mathbf{X}\boldsymbol{\beta})$$

  as in the GLM case
- However, the relative nature of the likelihood poses a challenge: unlike the GLM case, $\mathbf{W}$ is no longer a diagonal matrix
- This is not necessarily a conceptual difficulty, but it presents a considerable obstacle in terms of practical computation
- The speed of coordinate descent relies on the fact that each coordinate update takes an insignificant amount of time; if we have to do lots of calculations like $\mathbf{x}_j^\top \mathbf{W} \mathbf{x}_j$, the algorithm will be much slower

Do we need the exact value of $\mathbf{W}$?

- Thankfully, it turns out not to be terribly critical to use the exact Hessian matrix $\mathbf{W}$ when solving for the value of β that minimizes the quadratic approximation
- For example, in logistic regression using $w_i = 0.25$ for all i also works in terms of converging to the solution $\widehat{\beta}$, and surprisingly is often just as fast as using $w_i = \pi_i(1 - \pi_i)$
- Setting $w_i = 0.25$ is an example of something called an MM algorithm; convergence follows from Jensen's inequality
- Approximating $\mathbf{W}$ is also the main idea behind what are known as quasi-Newton methods

Approximating $\mathbf{W}$ with its diagonal

- A simple idea that turns out to work reasonably well is to simply approximate $\mathbf{W}$ by its diagonal
- Thus, although $\mathbf{W}$ is a dense $n \times n$ matrix, we can get away with simply calculating the elements along the diagonal and using this diagonalized $\mathbf{W}$ in the same manner as we did for penalized GLMs
- I am not aware of any formal proof that this guarantees convergence (indeed, convergence is never guaranteed for quadratic approximations without additional steps such as step-halving), but it seems to work well in practice

Challenge: Cross-validation

- Another challenge for Cox regression is how to carry out cross-validation
- In linear regression/logistic regression/multinomial regression/robust regression, we get actual predicted values for each observation and can evaluate the loss for each observation i in isolation
- For Cox regression, we estimate only *relative risk*, and thus have no way of assessing whether, say, $\mathbf{x}_i^\top \widehat{\boldsymbol{\beta}} = 0.8$ is a good predictor or not without a frame of reference

Proposal #1

- One proposal for how to carry out cross-validation for Cox regression models was proposed by Verweij and van Houwelingen (1993)
- Their idea was to calculate $L(\widehat{\boldsymbol{\beta}}_{(-i)})$, the partial likelihood for the full data based on data leaving out fold i, and $L_{(-i)}(\widehat{\boldsymbol{\beta}}_{(-i)})$, the partial likelihood for the data leaving out fold i based
- The cross-validation error is then

$$\mathrm{CVE} = \sum_i \{L(\widehat{\boldsymbol{\beta}}_{(-i)}) - L_{(-i)}(\widehat{\boldsymbol{\beta}}_{(-i)})\}$$

- This is the approach used by `glmnet`

Proposal #2

- An alternative approach, used by `ncvreg`, is to:
  (1) Calculate the full set of cross-validated linear predictors, $\boldsymbol{\eta} = \{\eta_{(-1)}, \ldots, \eta_{(-n)}\}$
  (2) Calculate the original Cox partial regression for the full data set based on the cross-validated linear predictors
- Both approaches are equivalent to regular cross-validation for ordinary likelihoods
- In simulation studies, the `ncvreg` approach has been shown to lead to somewhat more accurate estimates

Lung cancer data

- To illustrate, let's look at the data from a study by Shedden et al. (2008)
- In the study, retrospective data was collected for 442 patients with adenocarcinoma of the lung including their survival times, some additional clinical and demographic data, and expression levels of 22,283 genes taken from the tumor sample
- In the result that follow, I included age, sex, and treatment (whether the patient received adjuvant chemotherapy) as unpenalized predictors

R code

- To fit these models in `glmnet`:

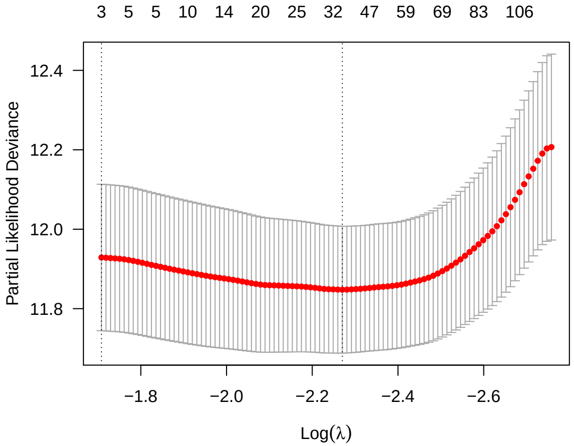
```
XX <- cbind(Z, X)
w <- rep(0:1, c(ncol(Z), ncol(X)))
cv.glmnet(XX, S, family="cox", penalty.factor=w)
```

- And in `ncvreg`:

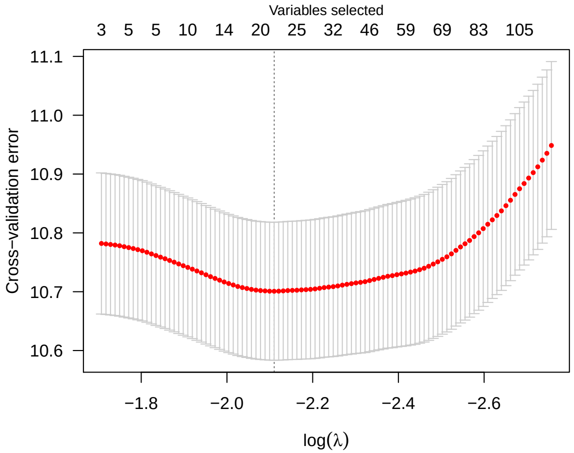
```
cv.ncvsurv(XX, S, penalty="lasso", penalty.factor=w)
```

- For both models, I used `lambda.min=0.35`; proceeding further along the path resulted in clear overfitting

Results: `glmnet`



Results: `ncvreg`



Remarks

- Both approaches suggest an improvement in accuracy as the first dozen or so genes are added to the model, followed by a relatively flat region; whether this drop is significant or not may depend on the approach
- As with other penalized regression models, an appealing feature is that coefficients retain their interpretation
- For example, the gene ZC2HC1A has a hazard ratio of 0.91 ($e^{\widehat{\boldsymbol{\beta}}_j} = 0.91$), indicating that patients with high ZC2HC1A expression have a 9% reduction in risk; as one might expect, the effect is estimated to be larger using MCP-penalized Cox regression (HR=0.88)