Different types of regression: Linear, Lasso, Ridge, Elastic net, Robust and K-neighbors

Agnieszka Prochenka

Faculty of Mathematics, Informatics and Mechanics, University of Warsaw

04.10.2009

We are given a linear problem:

$$y = \mathbb{X}\beta + \varepsilon$$
$$E(\varepsilon) = 0 \qquad Var(\varepsilon) = \sigma^2 \mathrm{I}_n$$

where:

- y is the vector of observations of length n,
- $\mathbb{X}$ the design matrix $n \times (p+1)$, where the first column is a ones vector,
- $\mathbb{X} = (x_1, x_2, \ldots, x_n)^T$
- β the coefficients vector of length $(p+1)$, $\beta = (\beta_0, \beta_1 \ldots, \beta_p)^T$,
- ε the random error of dimension n.

Linear Regression

If we want to find an estimator $\hat{\beta}$ as a function of $\mathbb{X}$ and y which minimizes the sum of the squared errors:

$$\sum_{i=1}^{n} \hat{\varepsilon}_i^{\,2} = \sum_{i=1}^{n} (y_i - x_i' \hat{\beta})^2$$

it turns out that $\hat{\beta}$ has to be of the form:

Estimators:

$$\hat{\beta} = (\mathbb{X}^T \mathbb{X})^{-1} \mathbb{X}^T \mathbb{Y}$$
$$\hat{y} = \mathbb{X}\hat{\beta}$$

R function

lm(formula, data, subset, weights,. . .)

R function

lm(formula, data, subset, weights,...)

However, in some conditions the linear regression doesn't work well:

1. p>n: the matrix $(\mathbb{X}^T\mathbb{X})$ is invertible
2. the rows of the design matrix $\mathbb{X}$ are highly correlated: the $\hat{\beta}$ coefficients are dependent on different x_i

The next 4 models bring solutions to these problems.

Regression “elastic net” solves the following problem:

$$\hat{\beta} = \arg_{b \in \mathbb{R}^{p+1}} \min \left[\frac{1}{2n} \sum_{i=1}^{n} (y_i - x_i^T b)^2 + \lambda P_\alpha(b_1, \ldots, b_p) \right]$$

where

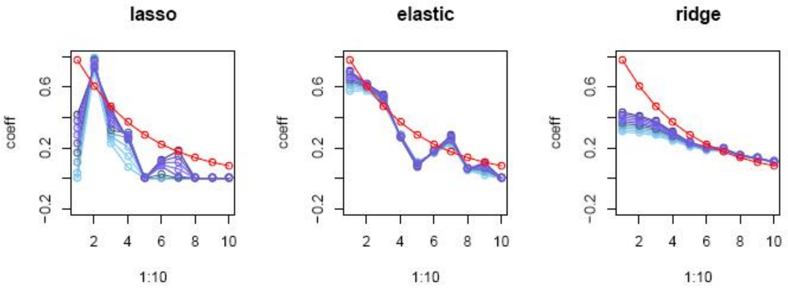
$$P_\alpha = \sum_{i=1}^{p} \left[\frac{1}{2}(1-\alpha) b_j^2 + \alpha |b_j| \right]$$

For different α we can have the following types of regression:

- $\alpha = 1$ lasso
- $\alpha = 0$ ridge
- $\alpha \in (0, 1)$ the general case of the elastic net

"Elastic net" regression

Ridge regression is known to shrink the coefficients of correlated predictors towards each other. Lasso is somewhat indifferent to very correlated predictors and will tend to pick one and ignore the rest.



“Elastic net” regression

R function

W pakiecie glmnet:
glmnet(x, y, weights, alpha,nlambda, lambda.min , lambda,. . .)
glmnet$\$a_0$; glmnet$\$beta$

“Elastic net” regression

R function

W pakiecie glmnet:
glmnet(x, y, weights, alpha,nlambda, lambda.min , lambda,. . .)
glmnet\$$a_0$; glmnet\$*beta*

literature: Friedman, Hastie, Tibshirani, “Regularization Paths for Generalized Linear Models via Coordinate Descent”, Stanford University, May 2008

This method assumes calculating the $\hat{y}$ without estimating $\hat{\beta}$. For each observation x, minimizing the euclidean distance, we find k closest observations from the design matrix $\mathbb{X}$ with indexes $j_1, \ldots, j_k$. $\hat{y}$ is the arithmetic mean of $y_{j_1}, \ldots, y_{j_k}$.

$$
\begin{aligned}
& A = \{1, \ldots, n\} \\
& for \quad (s \quad in \quad 1:k) \quad \{ \\
& \qquad l_s = \arg_{j \in A} \min \|x_j - x\| \\
& \qquad A = A \backslash \{l_s\} \\
& \} \\
& \hat{y} = mean(y_{l_1}, \ldots, y_{l_k})
\end{aligned}
$$

M estimator

M estimator minimizes a function:

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - x_i'\hat{\beta})$$

where ρ is the loss function.

Robust regression

M estimator

M estimator minimizes a function:

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - x_i'\hat{\beta})$$

where ρ is the loss function.

Some examples of the loss functions:

Least-Squares

$$\rho_{LS}(e) = e^2$$

Huber

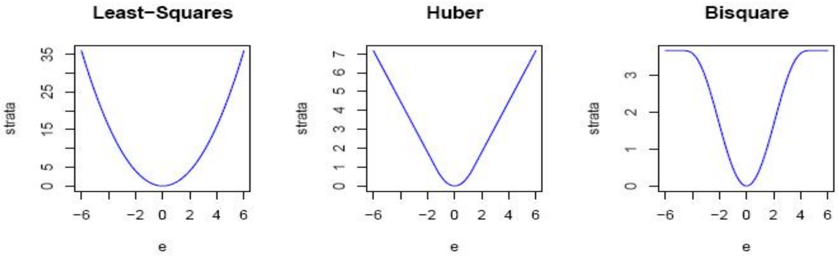
$$\rho_H(e) = \begin{cases} \frac{1}{2}e^2, & \text{for } |e| \leqslant k; \\ k|e| - \frac{1}{2}e^2, & \text{for } |e| > k. \end{cases}$$

Robust regression

Bisquare

$$\rho_B(e) = \begin{cases} \frac{k^2}{6}\left\{1 - \left[1 - \left(\frac{e}{k}\right)^2\right]^3\right\}, & \text{for } |e| \leqslant k; \\ \frac{k^2}{6}, & \text{for } |e| > k. \end{cases}$$

A graph of the loss functions with $k = 1.345$ for Huber, $k = 4.685$ for Bisquare:



R function

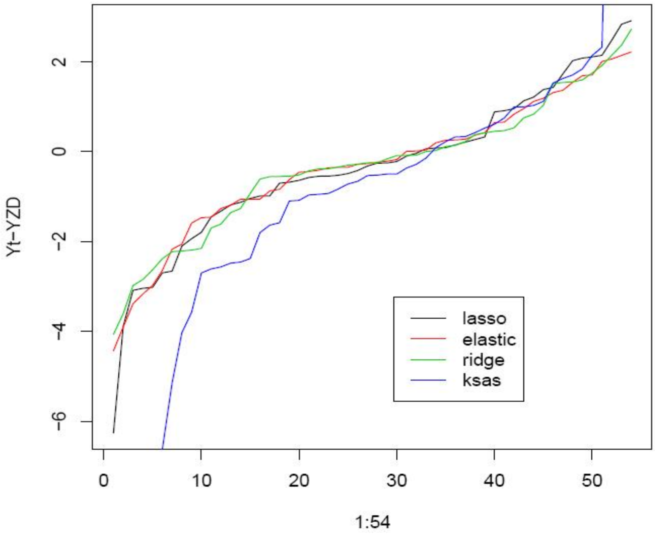
W pakiecie RLMM:
rlm(x, y, weights, psi = psi.huber,. . .)

Mice example $p \gg n$

In this example the $\mathbb{X}$ matrix has dimensions n=54, p=1000 and comes from real genetic data of mice, y was simulated.

```
#chr  loc rs  observed  129S1/SvImJ A/J AKR/J ALR/LtJ ALS/LtJ BA
01  3.013441  rs31192577  A/T T A T T T A A T T T T T = A A A T
01  3.036178  rs32166183  A/C C A C C C A A C C C C C A A A A C
01  3.036265  rs30543887  A/G G A G G G A A G G G G G A A A A G
01  3.039187  rs6365082 G/T T T T T T T T T T T T T T T T T T T
01  3.050333  rs46229295  G/T T G T T T G G T T T T T G G G G T
01  3.050460  rs45964436  G/T T T T T T T T T T T T T T T T T T
01  3.051362  rs30717399  A/G G A G G G A A G G G G G A A A A G
01  3.051854  rs32156135  A/G G A G G G A A G G G G G G G A A G
01  3.054018  rs47643955  A/G A A A A A A A A A A A A A A A A A
01  3.062749  rs31606309  A/C A C A A A C C A A A A A C C C C A
01  3.063538  rs30884626  C/G G C G G G C C G G G G G C C C C G
01  3.091209  rs47277169  A/G G G G G G G G G G G G G G G G G G
01  3.091406  rs31918559  G/T T G T = T G G = T H = T G G G G T
01  3.091519  rs51444971  C/G C C C C C C C C C C C C C C C C C
01  3.093816  rs31797356  C/T T C T T T C C T T T T T C C C C T
```

These are the sorted values of $y - \hat{y}$ for each model:



Mice example $p \gg n$

Measuring the goodness of fit with the mean of the squared errors $RSS = \sum_{i=1}^{n}(y_{t\,i} - x_i'\hat{\beta})^2$, we get:

model	RSS
lasso	3.21
elastic net	2.45
ridge	2.37
k neighbors	165.49

(The mean value of y_t is 415)

My master's thesis

In my master's thesis I am going to, looking at the data structure, try to find the best model for predictions in this situation.

1. I will numerously draw data
2. Choose different parameters
3. Try my models and look for some regularity

This is an example:

I repeat N times building the real data list:

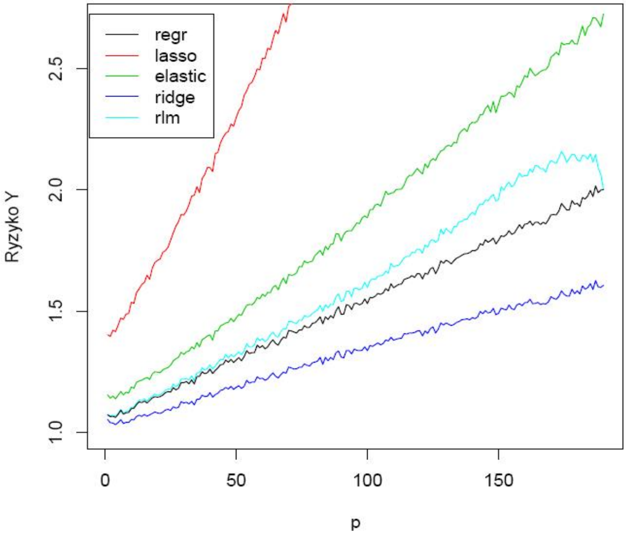
1. Drawing the matrix $\mathbb{X}$ from the p-dimensional normal distribution with the correlation matrix

$$\begin{pmatrix} 1 & \rho & \rho^2 & \dots \\ \rho & 1 & \rho & \dots \\ \rho^2 & \rho & 1 & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix}$$

2. Drawing ε_i from a normal distribution building the ε vector
3. Choosing β
4. Calculating the real value of y, $y_i = x_i'\beta + \varepsilon_i$

For the given data list I estimate the parameters for different models and check which fit best (for example which minimizes the sum of the squared loss). This is an example of what can come out:

β uniformly decreases from 1 to 0; $\rho = 0.1$; $\lambda = 0.5$; $n = 200$; $N = 250$



$\beta = (1, 0, 1, 0, 1, \ldots)$; $\rho = 0.9$; $\lambda = 0.5$; $n = 200$; $N = 250$