Linear Methods for Regression

Outline

- The simple linear regression model
- Multiple linear regression
- Model selection and shrinkage—the state of the art

Preliminaries

Data $(x_1, y_1), \ldots (x_N, y_N)$.

x_i is the predictor (regressor, covariate, feature, independent variable)

y_i is the response (dependent variable, outcome)

We denote the *regression function* by

$$\eta(x) = \mathrm{E}\,(Y|x)$$

This is the conditional expectation of Y given x.

The linear regression model assumes a specific linear form for η

$$\eta(x) = \alpha + \beta x$$

which is usually thought of as an approximation to the truth.

Fitting by least squares

Minimize:

$$\hat{\beta}_0, \hat{\beta} = \text{argmin}_{\beta_0,\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \beta x_i)^2$$

Solutions are

$$\hat{\beta} = \frac{\sum_{j=1}^{N}(x_i - \bar{x})y_i}{\sum_{j=1}^{N}(x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}\bar{x}$$

$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}x_i$ are called the fitted or predicted values

$r_i = y_i - \hat{\beta}_0 - \hat{\beta}x_i$ are called the residuals

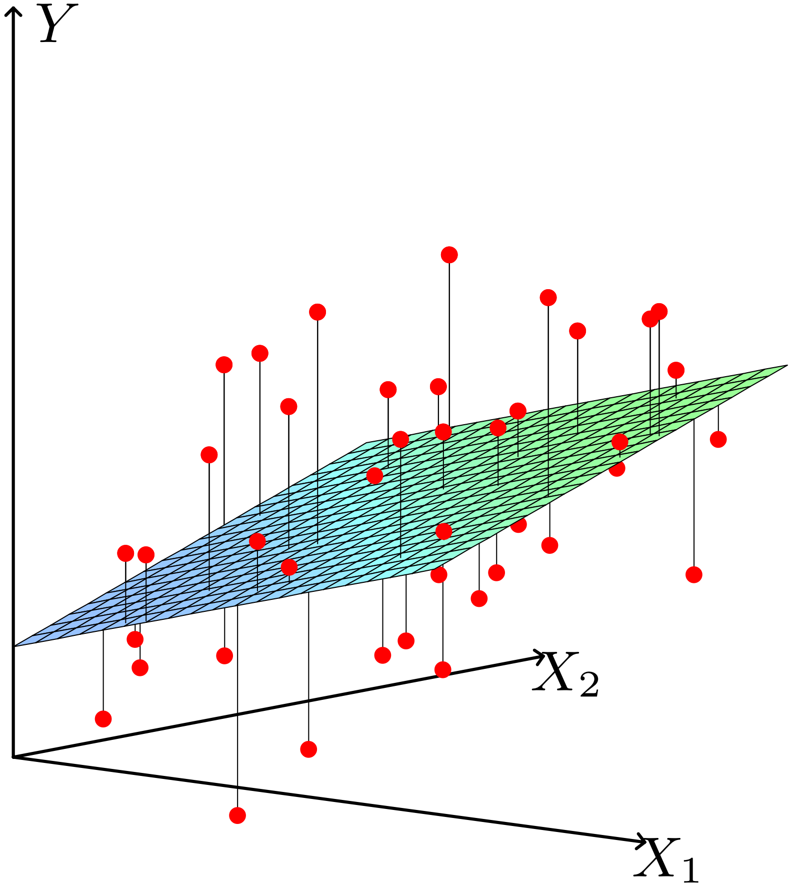


Figure 3.1 - view of linear regression in $\mathbb{R}^{p+1}$.

Standard errors & confidence intervals

We often assume further that

$$y_i = \beta_0 + \beta x_i + \epsilon_i$$

where $\mathrm{E}\,(\epsilon_i) = 0$ and $\mathrm{Var}\,(\epsilon_i) = \sigma^2$. Then

$$\mathrm{se}\,(\hat{\beta}) = \left[\frac{\sigma^2}{\sum(x_i - \bar{x})^2}\right]^{\frac{1}{2}}$$

Estimate σ^2 by $\hat{\sigma}^2 = \sum(y_i - \hat{y}_i)^2/(N-2)$.

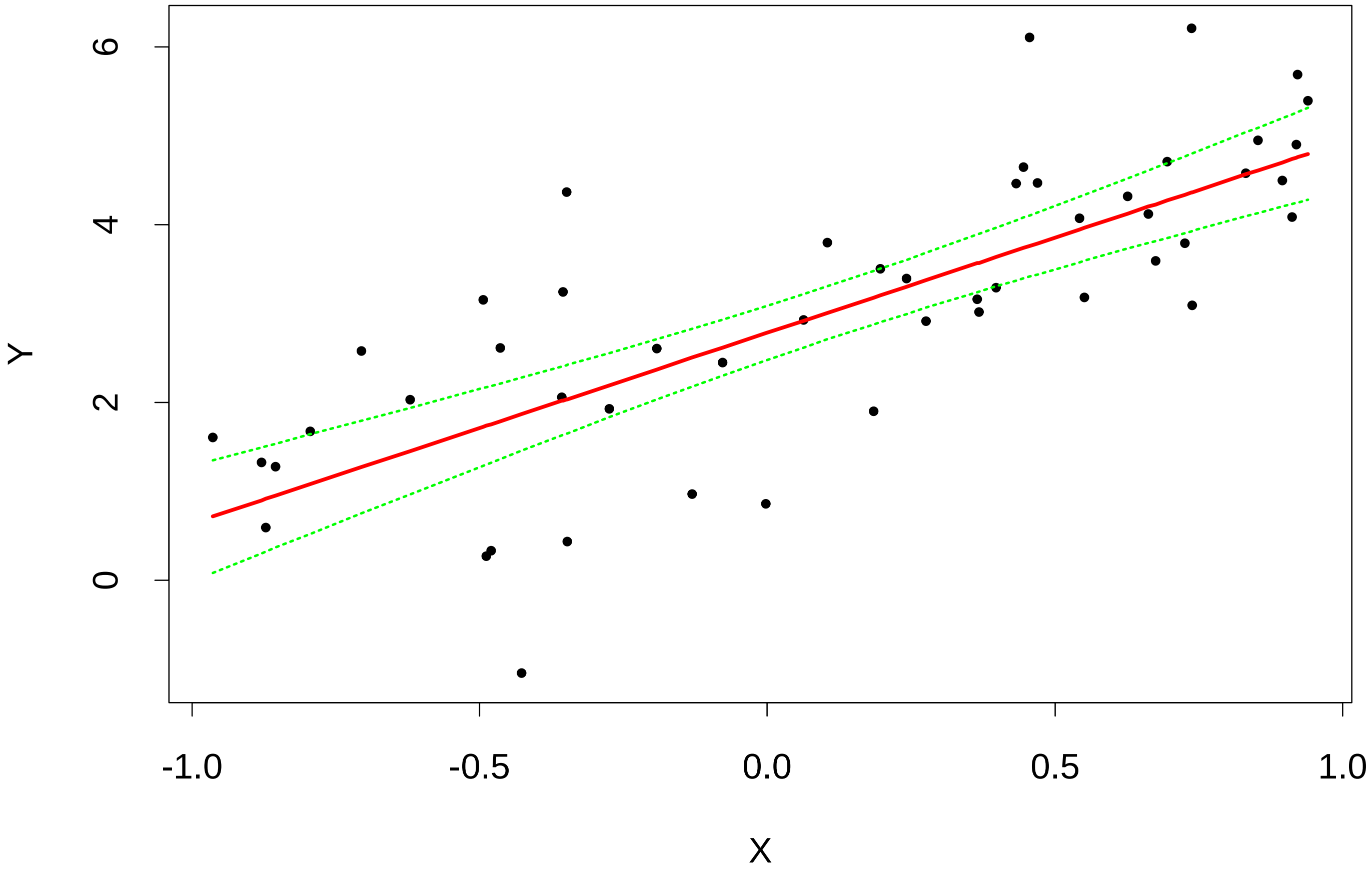
Under additional assumption of normality for the ϵ_is, a 95% confidence interval for β is: $\hat{\beta} \pm 1.96\hat{\mathrm{se}}(\hat{\beta})$

$$\hat{\mathrm{se}}\,(\hat{\beta}) = \left[\frac{\hat{\sigma}^2}{\sum(x_i - \bar{x})^2}\right]^{\frac{1}{2}}$$

Fitted Line and Standard Errors

$$\begin{aligned}\hat{\eta}(x) &= \hat{\beta}_0 + \hat{\beta}x \\ &= \bar{y} + \hat{\beta}(x - \bar{x})\end{aligned}$$

$$\begin{aligned}\text{se}[\hat{\eta}(x)] &= \left[\text{var}(\bar{y}) + \text{var}(\hat{\beta})(x - \bar{x})^2\right]^{\frac{1}{2}} \\ &= \left[\frac{\sigma^2}{n} + \frac{\sigma^2(x - \bar{x})^2}{\sum(x_i - \bar{x})^2}\right]^{\frac{1}{2}}\end{aligned}$$



Fitted regression line with pointwise standard errors: $\hat{\eta}(x) \pm 2\text{se}[\hat{\eta}(x)]$.

Multiple linear regression

Model is

$$f(x_i) = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j$$

Equivalently in matrix notation:

$$\mathbf{f} = X\beta$$

$\mathbf{f}$ is N-vector of predicted values

X is $N \times p$ matrix of regressors, with ones in the first column

β is a p-vector of parameters

Estimation by least squares

$$\begin{aligned}\hat{\beta} &= \operatorname{argmin} \sum_i (y_i - \beta_0 - \sum_{j=1}^{p-1} x_{ij}\beta_j)^2 \\ &= \operatorname{argmin} (\mathbf{y} - X\beta)^T (\mathbf{y} - X\beta)\end{aligned}$$

Figure 3.2 shows the N-dimensional geometry

Solution is

$$\begin{aligned}\hat{\beta} &= (X^T X)^{-1} X^T \mathbf{y} \\ \hat{\mathbf{y}} &= X\hat{\beta}\end{aligned}$$

Also Var $(\hat{\beta}) = (X^T X)^{-1}\sigma^2$

The course website has some additional notes (linear.pdf) on multiple linear regression, with an emphasis on computations.

The Bias-variance tradeoff

A good measure of the quality of an estimator $\hat{\mathbf{f}}(x)$ is the mean squared error. Let $\mathbf{f}_0(x)$ be the true value of $\mathbf{f}(x)$ at the point x. Then

$$\mathrm{Mse}\,[\hat{\mathbf{f}}(x)] = \mathrm{E}\,[\hat{\mathbf{f}}(x) - \mathbf{f}_0(x)]^2$$

This can be written as

$$\mathrm{Mse}\,[\hat{\mathbf{f}}(x)] = \mathrm{Var}\,[\hat{\mathbf{f}}(x)] + [\mathrm{E}\,\hat{\mathbf{f}}(x) - \mathbf{f}_0(x)]^2$$

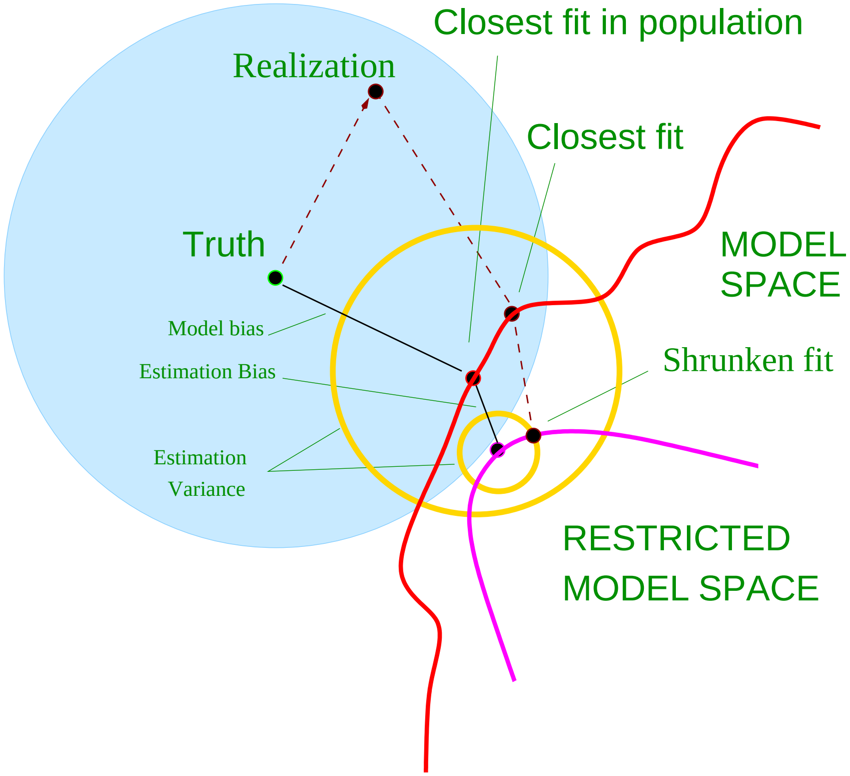
This is *variance* plus squared *bias*.

Typically, when bias is low, variance is high and vice-versa. Choosing estimators often involves a tradeoff between bias and variance.

- If the linear model is correct for a given problem, then the least squares prediction $\hat{\mathbf{f}}$ is unbiased, and has the lowest variance among all unbiased estimators that are linear functions of $\mathbf{y}$
- But there can be (and often exist) biased estimators with smaller Mse .
- Generally, by *regularizing* (shrinking, dampening, controlling) the estimator in some way, its variance will be reduced; if the corresponding increase in bias is small, this will be worthwhile.
- Examples of regularization: subset selection (forward, backward, all subsets); ridge regression, the lasso.
- In reality models are almost never correct, so there is an additional *model bias* between the closest member of the linear model class and the truth.

Model Selection

Often we prefer a restricted estimate because of its reduced estimation variance.



Analysis of time series data

Two approaches: *frequency domain* (fourier)—see discussion of wavelet smoothing.

Time domain. Main tool is auto-regressive (AR) model of order k:

$$y_t = \beta_1 y_{t-1} + \beta_2 y_{t-2} \cdots + \beta_k y_{t-k} + \epsilon_t$$

Fit by linear least squares regression on lagged data

$$\begin{aligned} y_t &= \beta_1 y_{t-1} + \beta_2 y_{t-2} \cdots \beta_k y_{t-k} \\ y_{t-1} &= \beta_1 y_{t-2} + \beta_2 y_{t-3} \cdots \beta_k y_{t-k-1} \\ \vdots &= \vdots \\ y_{k+1} &= \beta_1 y_k + \beta_2 y_{k-1} \cdots \beta_k y_1 \end{aligned}$$

Example: NYSE data

Time series of 6200 daily measurements, 1962-1987

`volume` — log(trading volume) — *outcome*

`volume.Lj` — $\text{log(trading volume)}_{\text{day}-j}, \quad j = 1, 2, 3$

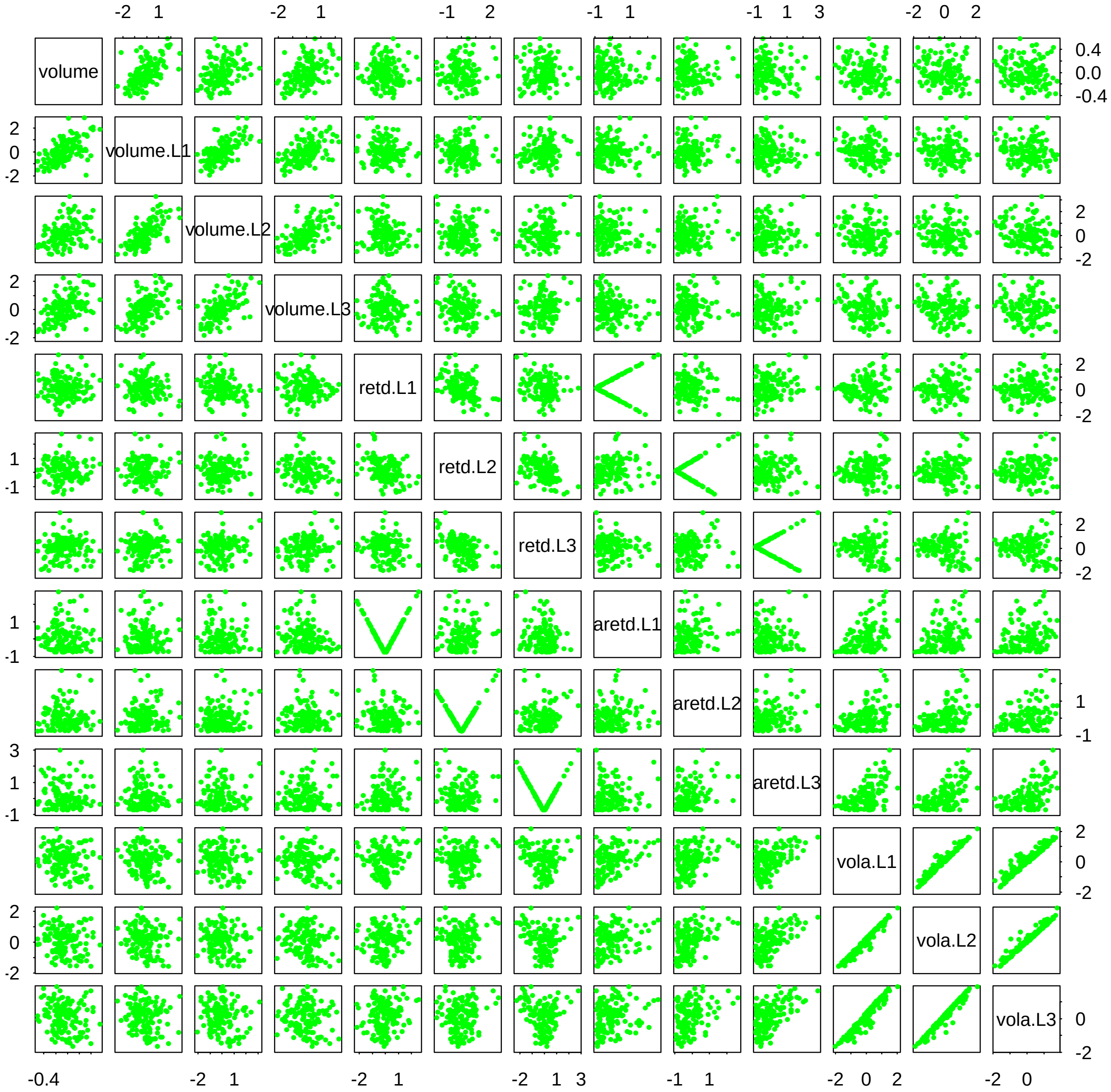
`ret.Lj` — $\Delta\ \text{log(Dow Jones)}_{\text{day}-j}, \quad j = 1, 2, 3$

`aret.Lj` — $|\Delta\text{log(Dow Jones)}|_{\text{day}-j}, \quad j = 1, 2, 3$

`vola.Lj` — $\text{volatility}_{\text{day}-j}, \quad j = 1, 2, 3$

Source—Weigend and LeBaron (1994)

We randomly selected a training set of size 50 and a test set of size 500, from the first 600 observations.


volume
volume.L1
volume.L2
volume.L3
retd.L1
retd.L2
retd.L3
aretd.L1
aretd.L2
aretd.L3
vola.L1
vola.L2
vola.L3
-2 1
-2 1
-1 2
-1 1
-1 1 3
-2 0 2
0.4
0.0
-0.4
2
0
-2
-0.4
-2 1
-2 1
-2 1 3
-1 1
-2 0 2
-2 0
1
-1
3
1
-1

OLS Fit

Results of ordinary least squares analysis of NYSE data

Term	Coefficient	Std. Error	t-Statistic
Intercept	-0.02	0.04	-0.64
volume.L1	0.09	0.05	1.80
volume.L2	0.06	0.05	1.19
volume.L3	0.04	0.05	0.81
retd.L1	0.00	0.04	0.11
retd.L2	-0.02	0.05	-0.46
retd.L3	-0.03	0.04	-0.65
aretd.L1	0.08	0.07	1.12
aretd.L2	-0.02	0.05	-0.45
aretd.L3	0.03	0.04	0.77
vola.L1	0.20	0.30	0.66
vola.L2	-0.50	0.40	-1.25
vola.L3	0.27	0.34	0.78

Variable subset selection

We retain only a subset of the coefficients and set to zero the coefficients of the rest.

There are different strategies:

- *All subsets regression* finds for each $s \in 0, 1, 2, \ldots p$ the subset of size s that gives smallest residual sum of squares. The question of how to choose s involves the tradeoff between bias and variance: can use cross-validation (see below)
- Rather than search through all possible subsets, we can seek a good path through them. *Forward stepwise selection* starts with the intercept and then sequentially adds into the model the variable that most improves the fit. The improvement in fit is usually based on the

F ratio

$$F = \frac{RSS(\hat{\beta}^{old}) - RSS(\hat{\beta}^{new})}{RSS(\hat{\beta}^{new})/(N-s)}$$

- *Backward stepwise selection* starts with the full OLS model, and sequentially deletes variables.
- There are also hybrid *stepwise selection* strategies which add in the best variable and delete the least important variable, in a sequential manner.
- Each procedure has one or more *tuning parameters*:
  - subset size
  - P-values for adding or dropping terms

Model Assessment

Objectives:

1. Choose a value of a tuning parameter for a technique
2. Estimate the prediction performance of a given model

For both of these purposes, the best approach is to run the procedure on an independent test set, if one is available

If possible one should use different test data for (1) and (2) above: a *validation set* for (1) and a *test set* for (2)

Often there is insufficient data to create a separate validation or test set. In this instance *Cross-Validation* is useful.

K-Fold Cross-Validation

Primary method for estimating a tuning parameter λ (such as subset size)

Divide the data into K roughly equal parts (typically K=5 or 10)

1	2	3	4	5
Train	Train	Test	Train	Train

- for each $k = 1, 2, \ldots K$, fit the model with parameter λ to the other $K - 1$ parts, giving $\hat{\beta}^{-k}(\lambda)$ and compute its error in predicting the kth part: $E_k(\lambda) = \sum_{i \in kth\ part}(y_i - \mathbf{x}_i \hat{\beta}^{-k}(\lambda))^2$.

  This gives the cross-validation error

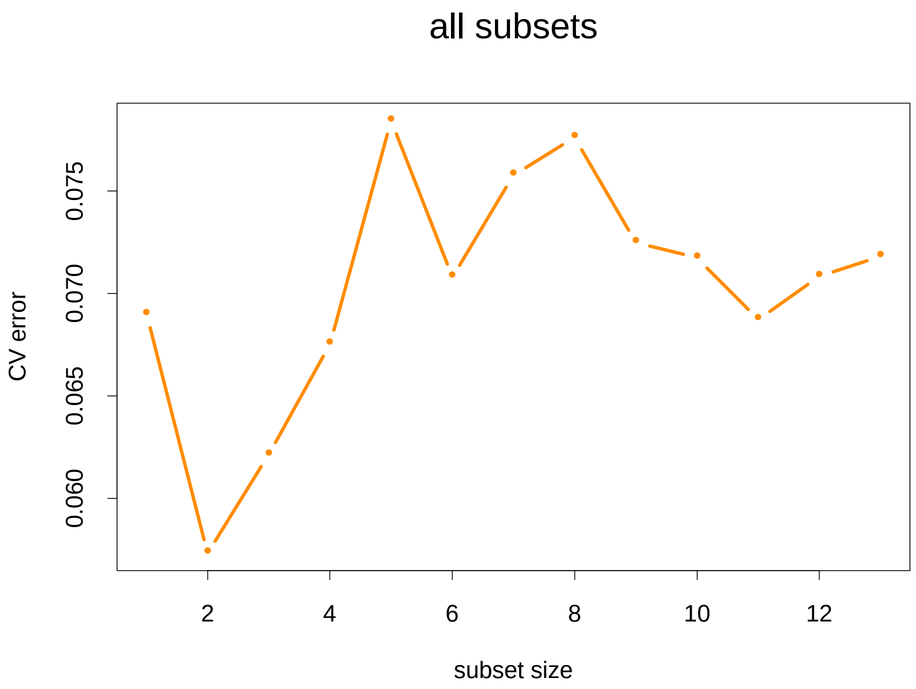
$$CV(\lambda) = \frac{1}{K} \sum_{k=1}^{K} E_k(\lambda)$$

- do this for many values of λ and choose the value of λ that makes $CV(\lambda)$ smallest.

- In our variable subsets example, λ is the subset size
- $\hat{\beta}^{-k}(\lambda)$ are the coefficients for the best subset of size λ, found from the training set that leaves out the kth part of the data
- $E_k(\lambda)$ is the estimated test error for this best subset.
- from the K cross-validation training sets, the K test error estimates are averaged to give

$$CV(\lambda) = (1/K) \sum_{k=1}^{K} E_k(\lambda).$$

- Note that different subsets of size λ will (probably) be found from each of the K cross-validation training sets. Doesn't matter: focus is on subset size, not the actual subset.



CV curve for NYSE data

- The focus is on *subset size*—not which variables are in the model.
- Variance increases slowly—typically σ^2/N per variable.

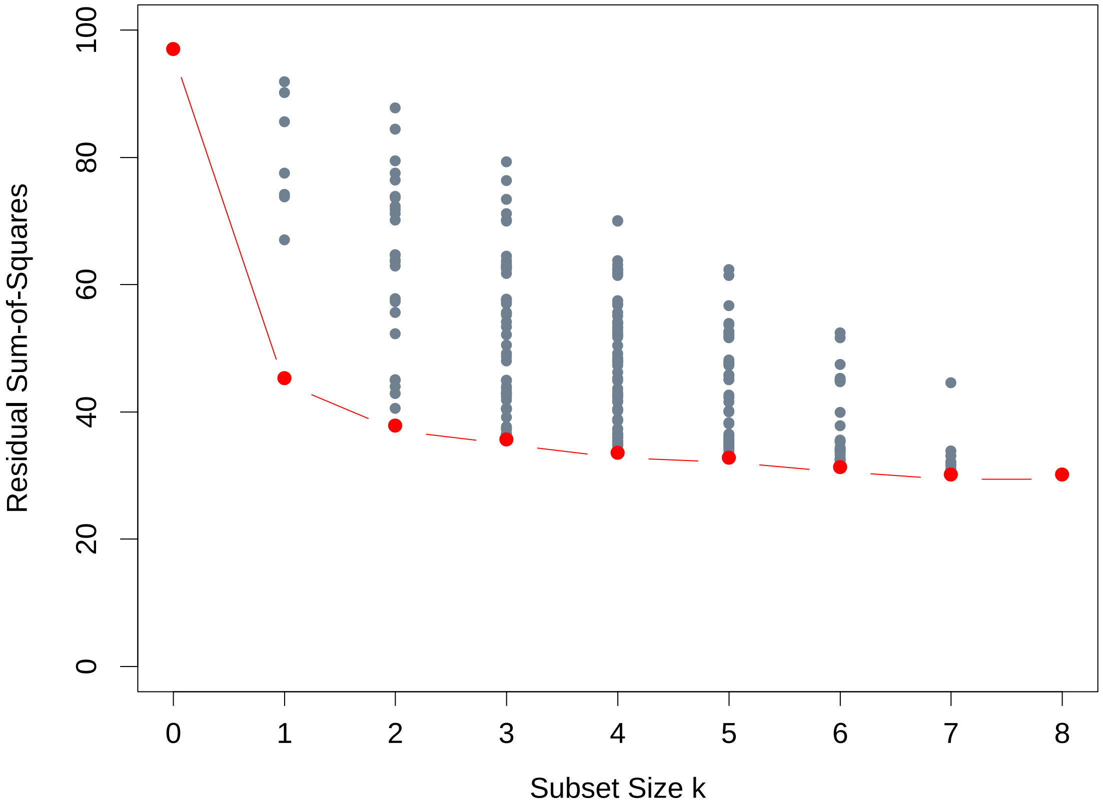


Figure 3.5: *All possible subset models for the prostate cancer example. At each subset size is shown the residual sum-of-squares for each model of that size.*

The Bootstrap approach

- Bootstrap works by sampling N times with replacement from training set to form a "bootstrap" data set. Then model is estimated on bootstrap data set, and predictions are made for original training set.
- This process is repeated many times and the results are averaged.
- Bootstrap most useful for estimating standard errors of predictions.
- Can also use modified versions of the bootstrap to estimate prediction error. Sometimes produces better estimates than cross-validation (topic for current research)

Cross-validation- revisited

Consider a simple classifier for wide data:

1. Starting with 5000 predictors, find the 200 predictors having the largest correlation with the class labels
2. Carry about nearest-centroid classification using only these 200 genes

How do we estimate the test set performance of this classifier?

Wrong: Apply cross-validation in step 2. *Right:* Apply cross-validation to steps 1 and 2.

It is easy to simulate realistic data with the class labels independent of the outcome, — so that true test error =50%— but "Wrong" CV error estimate is zero!

We have seen this error made in 4 high profile microarray papers in the last couple of years. See Ambroise and McLachlan PNAS 2002.

A little cheating goes a long way

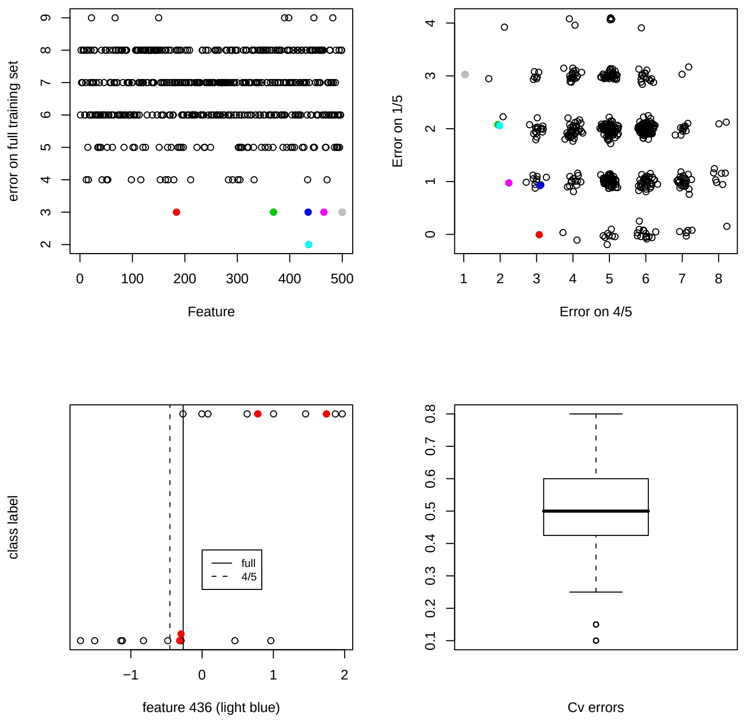
Validation and test set issues

Important to have both cross-validation and test sets, since we often run CV many times, fiddling with different parameters. This can bias the CV results A separate test set provides a convincing, independent assessment of a model's performance Test set results might still overestimate actual performance, as a real future test set may differ in many ways from today's data

Does cross-validation really work?

Consider a scenario with $N = 20$ samples in two equal-sized classes, and $p = 500$ quantitative features that are independent of the class labels. The true error rate of any classifier is 50%. Consider a simple univariate classifier- a single split that minimizes the misclassification error (a "stump"). Fitting to the entire training set, we will find a feature that splits the data very well If we do 5-fold CV, this same feature should split any $4/5$ths and $1/5$th of the data well too, and hence its CV error will be small (much less than 50%) Thus CV does not give an accurate estimate of error. Is this argument correct?

Simulation results

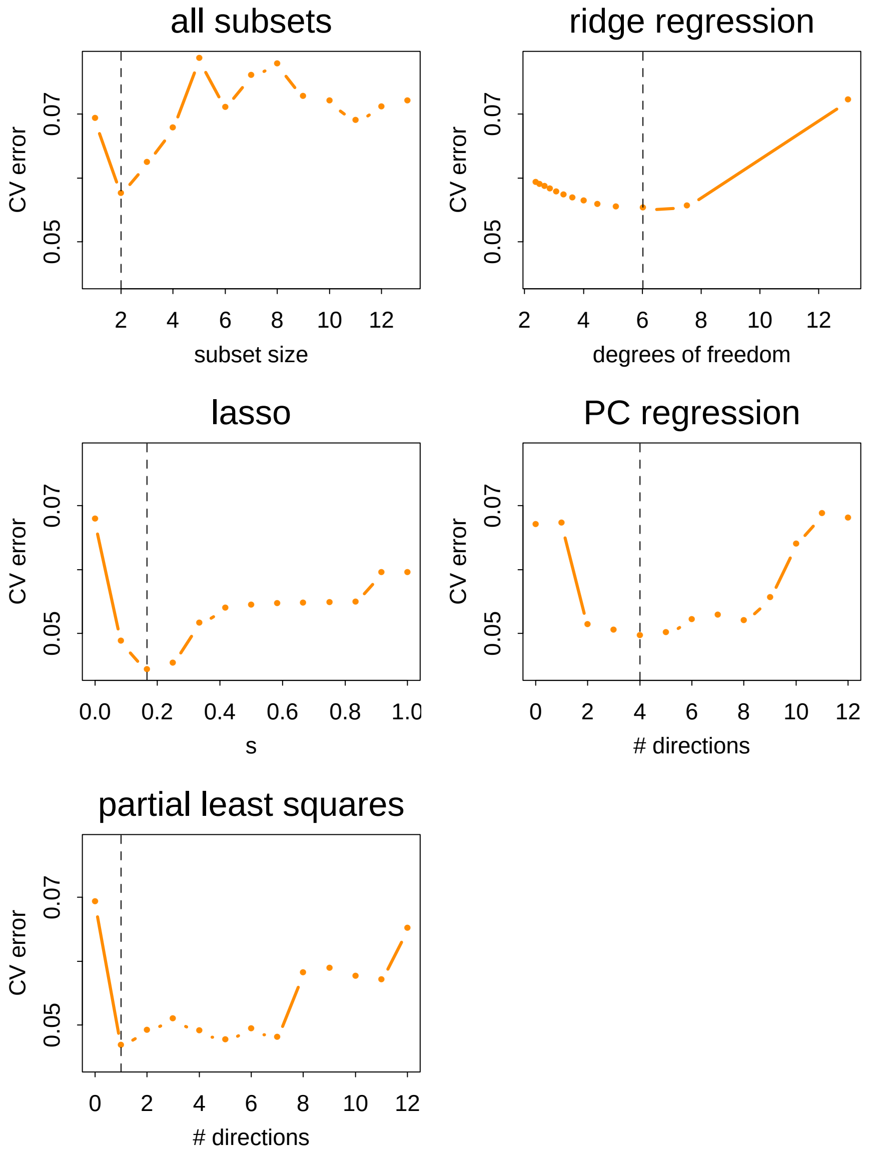


NYSE example continued

Table shows the coefficients from a number of different selection and shrinkage methods, applied to the NYSE data.

Term	OLS	VSS	Ridge	Lasso	PCR	PLS
Intercept	-0.02	0.00	-0.01	-0.02	-0.02	-0.04
volume.L1	0.09	0.16	0.06	0.09	0.05	0.06
volume.L2	0.06	0.00	0.04	0.02	0.06	0.06
volume.L3	0.04	0.00	0.04	0.03	0.04	0.05
retd.L1	0.00	0.00	0.01	0.01	0.02	0.01
retd.L2	-0.02	0.00	-0.01	0.00	-0.01	-0.02
retd.L3	-0.03	0.00	-0.01	0.00	-0.02	0.00
aretd.L1	0.08	0.00	0.03	0.02	-0.02	0.00
aretd.L2	-0.02	-0.05	-0.03	-0.03	-0.01	-0.01
aretd.L3	0.03	0.00	0.01	0.00	0.02	0.01
vola.L1	0.20	0.00	0.00	0.00	-0.01	-0.01
vola.L2	-0.50	0.00	-0.01	0.00	-0.01	-0.01
vola.L3	0.27	0.00	-0.01	0.00	-0.01	-0.01
Test err	0.050	0.041	0.042	0.039	0.045	0.044
SE	0.007	0.005	0.005	0.005	0.006	0.006

CV was used on the 50 training observations (except for OLS). Test error for constant: 0.061.



Estimated prediction error curves for the various selection and shrinkage methods. The arrow indicates the estimated minimizing value of the complexity parameter. Training sample size = 50.

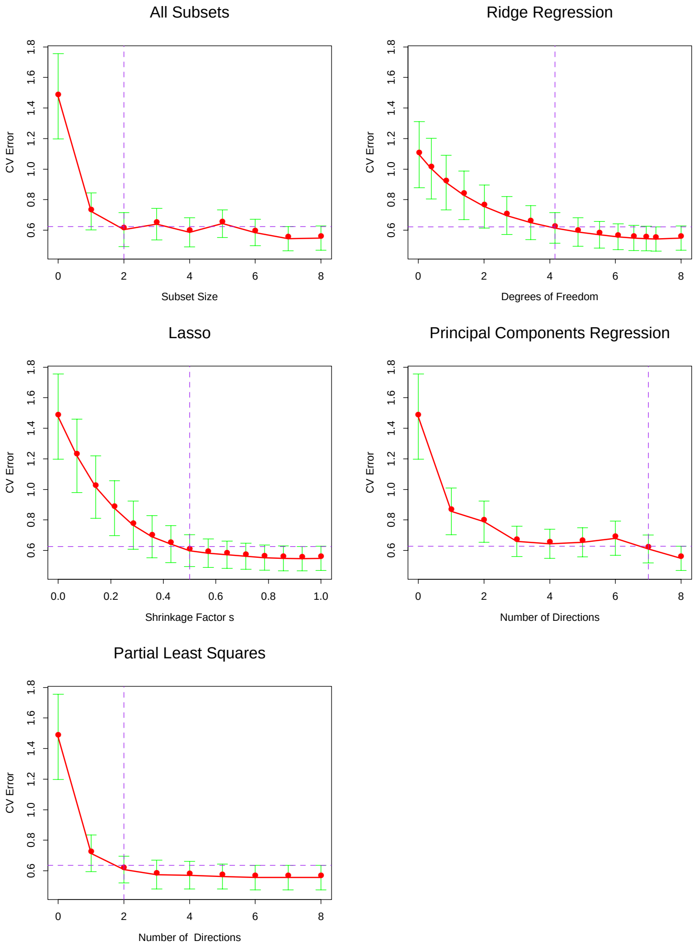


Figure 3.6: *Estimated prediction error curves and their standard errors for the various selection and shrinkage methods, found by 10-fold cross-validation.*

Shrinkage methods

Ridge regression

The ridge estimator is defined by

$$\hat{\beta}^{\text{ridge}} = \text{argmin}(\mathbf{y} - X\beta)^T(\mathbf{y} - X\beta) + \lambda\beta^T\beta$$

Equivalently,

$$\begin{aligned} \hat{\beta}^{\text{ridge}} \quad &= \quad \text{argmin } (\mathbf{y} - X\beta)^T(\mathbf{y} - X\beta) \\ & \quad\quad \text{subject to } \sum \beta_j^2 \le s. \end{aligned}$$

The parameter $\lambda > 0$ penalizes β_j proportional to its size β_j^2. Solution is

$$\hat{\beta}_\lambda = (X^TX + \lambda I)^{-1}X^T\mathbf{y}$$

where I is the identity matrix. This is a biased estimator that for some value of $\lambda > 0$ may have smaller mean squared error than the least squares estimator.

Note $\lambda = 0$ gives the least squares estimator; if $\lambda \to \infty$, then $\hat{\beta} \to 0$.

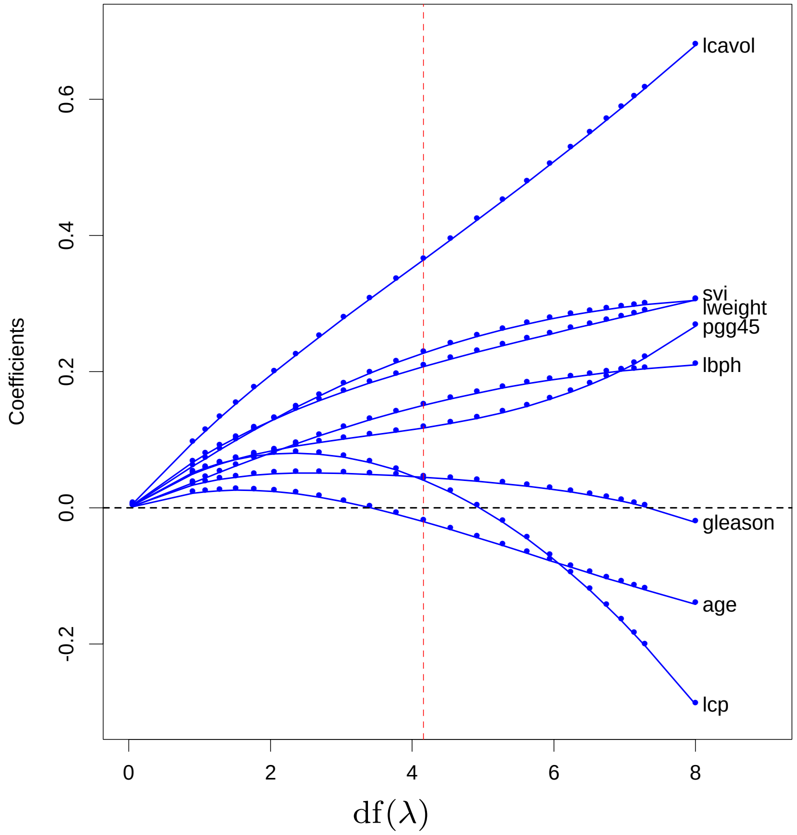


Figure 3.7: *Profiles of ridge coefficients for the prostate cancer example, as tuning parameter* λ *is varied. Coefficients are plotted versus* $\mathrm{df}(\lambda)$*, the effective degrees of freedom. A vertical line is drawn at* $\mathrm{df} = 4.16$*, the value chosen by cross-validation.*

The Lasso

The lasso is a shrinkage method like ridge, but acts in a nonlinear manner on the outcome y.

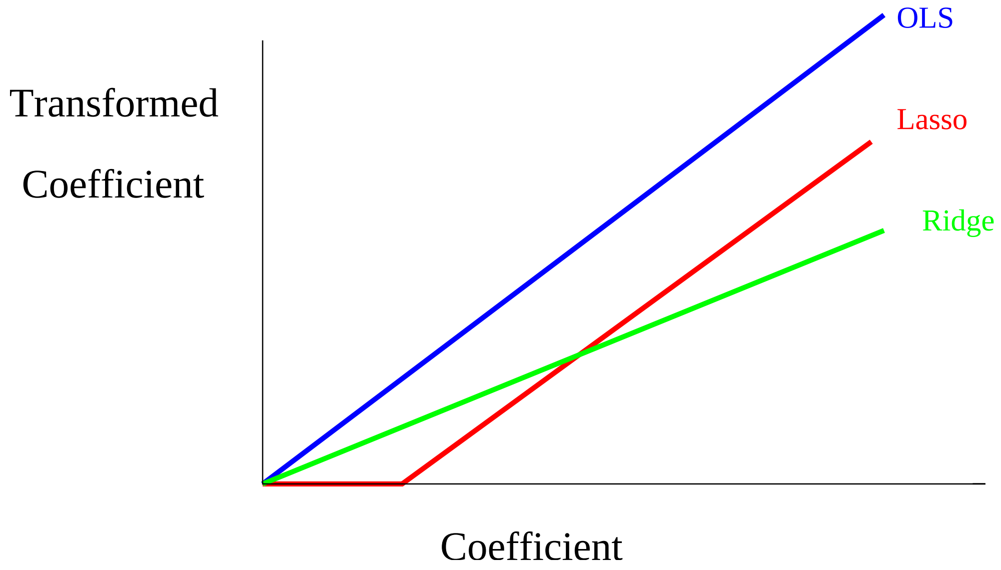
The lasso is defined by

$$\begin{aligned}\hat{\beta}^{\text{lasso}} &= \text{argmin}\,(\mathbf{y}-X\beta)^T(\mathbf{y}-X\beta)\\ &\ \text{subject to} \sum|\beta_j| \le t\end{aligned}$$

- Notice that ridge penalty $\sum\beta_j^2$ is replaced by $\sum|\beta_j|$.
- this makes the solutions nonlinear in $\mathbf{y}$, and a quadratic programming algorithm is used to compute them.
- because of the nature of the constraint, if t is chosen small enough then the lasso will set some coefficients exactly to zero. Thus the lasso does a kind of continuous model selection.

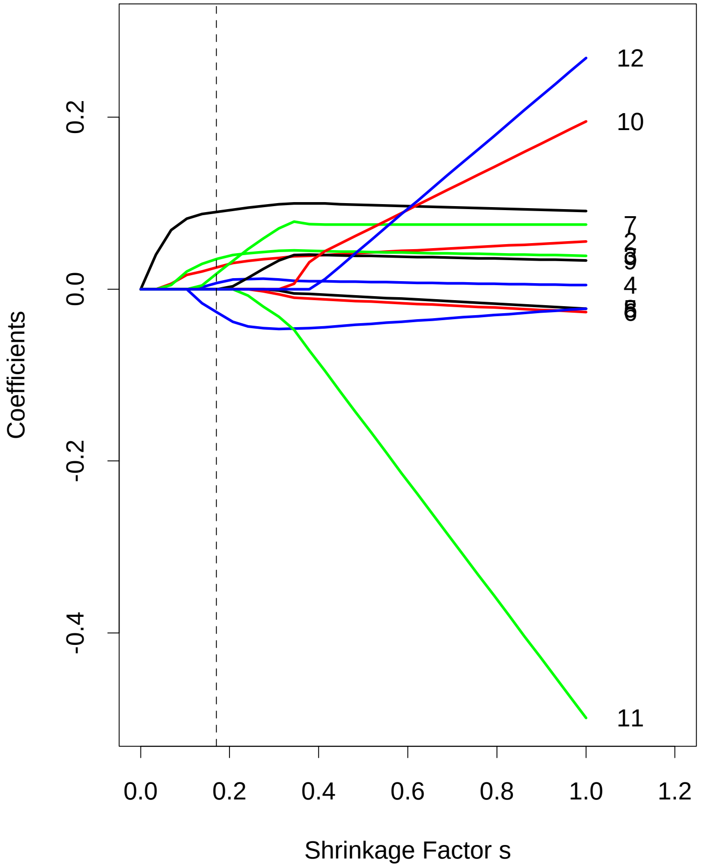
- The parameter t should be adaptively chosen to minimize an estimate of expected, using say cross-validation

- *Ridge vs Lasso:* if inputs are orthogonal, ridge *multiplies* least squares coefficients by a constant < 1, lasso *translates* them towards zero by a constant, truncating at zero.



Lasso in Action

Profiles of coefficients for NYSE data as lasso shrinkage is varied.



$$s = t/t_0 \in [0, 1], \text{ where } t_0 = \sum |\hat{\beta}_{OLS}|.$$

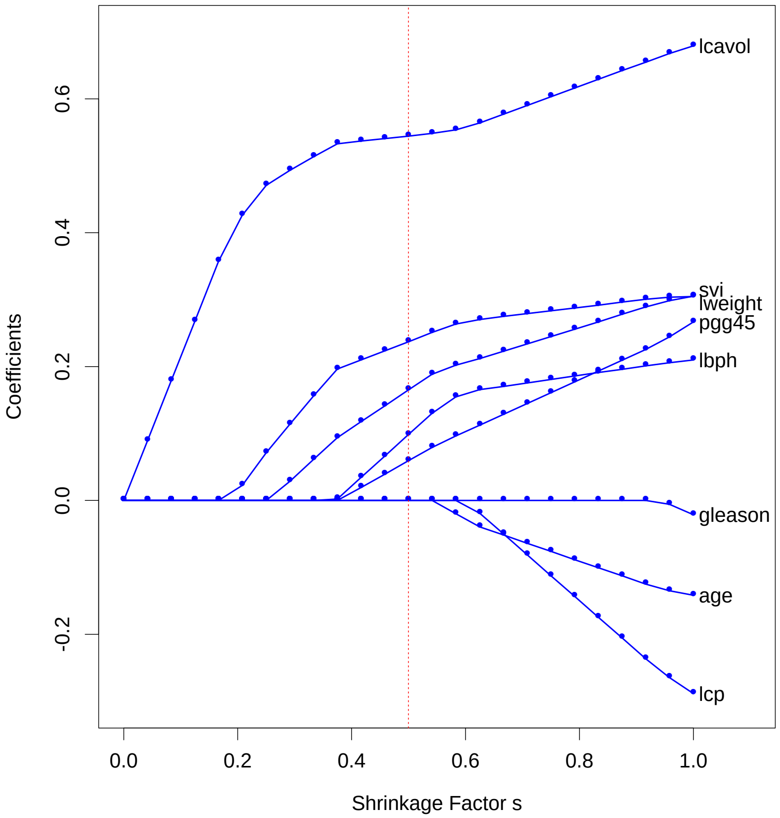


Figure 3.9: *Profiles of lasso coefficients, as tuning parameter t is varied. Coefficients are plotted versus $s = t/\sum_1^p |\hat{\beta}_j|$. A vertical line is drawn at $s = 0.5$, the value chosen by cross-validation. Compare Figure 3.7 on page 7; the lasso profiles hit zero, while those for ridge do not.*

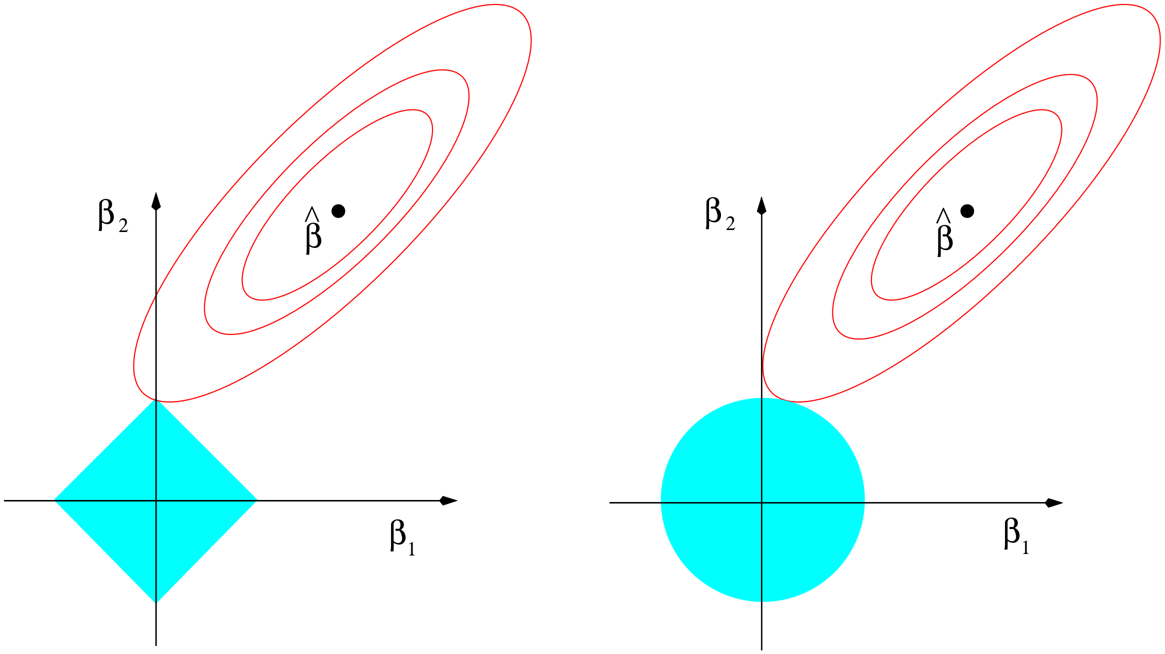


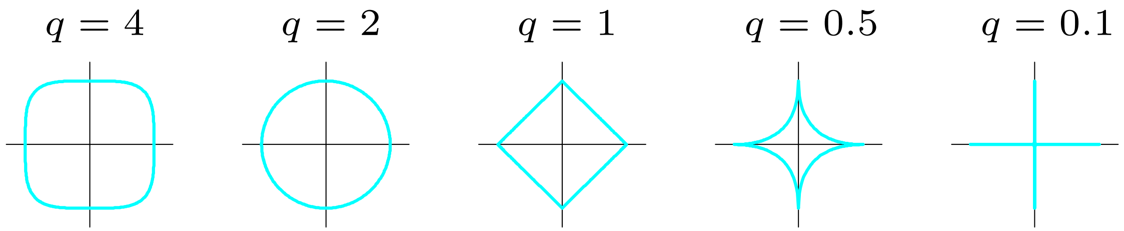
Figure 3.12: *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

A family of shrinkage estimators

Consider the criterion

$$\begin{aligned}\tilde{\beta} &= \text{argmin}_{\,\beta} \sum_{i=1}^{N} (y_i - x_i^T\beta)^2 \\ &\quad \text{subject to} \sum |\beta_j|^q \leq s\end{aligned}$$

for $q \geq 0$. The contours of constant value of $\sum_j |\beta_j|^q$ are shown for the case of two inputs.



Contours of constant value of $\sum_j |\beta_j|^q$ for given values of q.

Thinking of $|\beta_j|^q$ as the log-prior density for β_j, these are also the equi-contours of the prior.

Use of derived input directions

Principal components regression

We choose a set of linear combinations of the x_js, and then regress the outcome on these linear combinations.

The particular combinations used are the sequence of principal components of the inputs. These are uncorrelated and ordered by decreasing variance.

If S is the sample covariance matrix of $x_1, \ldots, x_p$, then the eigenvector equations

$$S\mathbf{q}_\ell = d_j^2 \mathbf{q}_\ell$$

define the principal components of S.

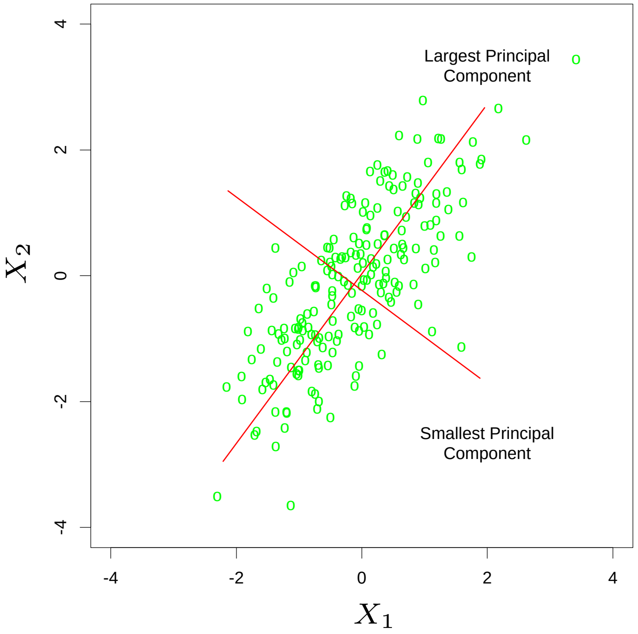


Figure 3.8: *Principal components of some input data points. The largest principal component is the direction that maximizes the variance of the projected data, and the smallest principal component minimizes that variance. Ridge regression projects* $\mathbf{y}$ *onto these components, and then shrinks the coefficients of the low-variance components more than the high-variance components.*

Digression: some notes on Principal Components and the SVD (PCA.pdf)

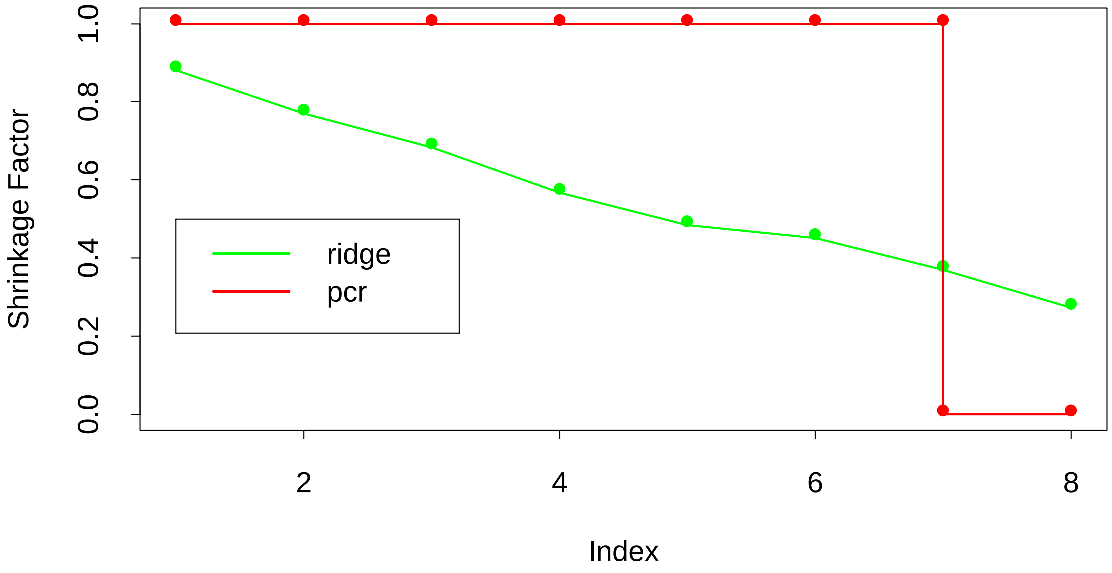
PCA regression continued

- Write $\mathbf{q}_{(j)}$ for the ordered principal components, ordered from largest to smallest value of d_j^2.
- Then principal components regression computes the derived input columns $\mathbf{z}_j = X\mathbf{q}_{(j)}$ and then regresses $\mathbf{y}$ on $\mathbf{z}_1, \mathbf{z}_2, \ldots \mathbf{z}_J$ for some $J \leq p$.
- Since the $\mathbf{z}_j$s are orthogonal, this regression is just a sum of univariate regressions:

$$\hat{\mathbf{y}}^{\mathrm{pcr}} = \bar{y} + \sum_{j=1}^{J} \hat{\gamma}_j \mathbf{z}_j$$

where $\hat{\gamma}_j$ is the univariate regression coefficient of $\mathbf{y}$ on $\mathbf{z}_j$.

- Principal components regression is very similar to ridge regression: both operate on the principal components of the input matrix.
- Ridge regression shrinks the coefficients of the principal components, with relatively more shrinkage applied to the smaller components than the larger; principal components regression discards the $p - J + 1$ smallest eigenvalue components.



Partial least squares

This technique also constructs a set of linear combinations of the x_js for regression, but unlike principal components regression, it uses $\mathbf{y}$ (in addition to X) for this construction.

- We assume that $\mathbf{y}$ is centered and begin by computing the univariate regression coefficient $\hat{\gamma}_j$ of $\mathbf{y}$ on each $\mathbf{x}_j$
- From this we construct the derived input $\mathbf{z}_1 = \sum \hat{\gamma}_j \mathbf{x}_j$, which is the first partial least squares direction.
- The outcome $\mathbf{y}$ is regressed on $\mathbf{z}_1$, giving coefficient $\hat{\beta}_1$, and then we orthogonalize $\mathbf{y}, \mathbf{x}_1, \ldots \mathbf{x}_p$ with respect to $\mathbf{z}_1$: $\mathbf{r}_1 = \mathbf{y} - \hat{\beta}_1 \mathbf{z}_1$, and $\mathbf{x}_\ell^* = \mathbf{x}_\ell - \hat{\theta}_\ell \mathbf{z}_1$
- We continue this process, until J directions have been obtained.

- In this manner, partial least squares produces a sequence of derived inputs or directions $\mathbf{z}_1, \mathbf{z}_2, \ldots \mathbf{z}_J$.
- As with principal components regression, if we continue on to construct $J = p$ new directions we get back the ordinary least squares estimates; use of $J < p$ directions produces a reduced regression
- Notice that in the construction of each $\mathbf{z}_j$, the inputs are weighted by the strength of their univariate effect on $\mathbf{y}$.
- It can also be shown that the sequence $\mathbf{z}_1, \mathbf{z}_2, \ldots \mathbf{z}_p$ represents the conjugate gradient sequence for computing the ordinary least squares solutions.

Ridge vs PCR vs PLS vs Lasso

Recent study has shown that ridge and PCR outperform PLS in prediction, and they are simpler to understand.

Lasso outperforms ridge when there are a moderate number of sizable effects, rather than many small effects. It also produces more interpretable models.

These are still topics for ongoing research.