Shrinkage and Selection Methods: LAR, LASSO, Ridge & Elastic Net

Paweł Polak

November 13, 2017

Linear Regression Models - Lecture 12

Content:

- Sparse Problem
- Forward Selection
- LAR - Least Angle Regression
- The LASSO
- Ridge regression
- Elastic-Net

General Linear Model

- *Independent responses* of the form $Y_i \sim N(\mu_i, \sigma^2)$, where

$$\mu_i = \mathbf{X}_i^\top \boldsymbol{\beta}$$

  for some known vector of *explanatory* variables $\mathbf{X}_i^\top = (X_{i1}, \ldots, X_{ip})$.

- Unknown *parameter* vector $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_{P-1})^\top$, where $P < N$.

- This is the *linear model* and is usually written as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

  (in vector notation) where

$$\mathbf{Y} = \begin{pmatrix} Y_1 \\ \vdots \\ Y_N \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} x_1^\top \\ \vdots \\ x_N^\top \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_{P-1} \end{pmatrix}, \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_N \end{pmatrix},$$

  where $\varepsilon_i \overset{i.i.d.}{\sim} N(0, \sigma^2)$, for $i = 1, 2, \ldots, N$.

Sparse Problem

- Unless stated otherwise, $\mathbf{Y}$ is assumed centered and $\mathbf{X}$ is assumed centered and normalized such that each variable has zero mean and unit Euclidean length.
- A sparse method for regression estimates a coefficient vector β with many *zero elements*, giving an estimate $\widehat{\mathbf{Y}}$ of $\mathbf{Y}$ which is a linear combination of a *subset* of available variables in $\mathbf{X}$.
- The set $\mathcal{A}$ denotes the indices in β corresponding to non-zero elements; we refer to this as the *active set*.
- The set $\mathcal{I}$ is called the *inactive set* and denotes the complement of $\mathcal{A}$.
- We use these sets also to denote submatrices such as the $(N \times |\mathcal{A}|)$ matrix $\mathbf{X}_{\mathcal{A}}$, consisting of the columns (variables) of $\mathbf{X}$ corresponding to the indices in $\mathcal{A}$.

Forward Selection Algorithm

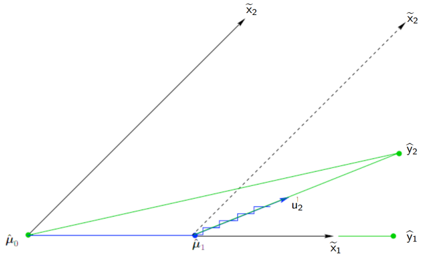
```
(1) Initialize the active set $\mathcal{A} = \varnothing$ and the inactive set $\mathcal{I} = \{1, \dots, P\}$.
(2) Initialize the coefficient vector $\boldsymbol{\beta}^{(0)} = 0$
(3) for $k \in \{0, \dots, P-1\}$ do
    (4) Find variable maximally correlated with the current residual
        $$i = \arg\max_{i \in \mathcal{I}} \left| \mathbf{X}_i^T \left( \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^{(k)} \right) \right|$$
    (5) Move $i$ from $\mathcal{I}$ to $\mathcal{A}$.
    (6) Update the active set coefficients $\boldsymbol{\beta}_{\mathcal{A}}^{(k+1)} = \left( \mathbf{X}_{\mathcal{A}}^T \mathbf{X}_{\mathcal{A}} \right)^{-1} \mathbf{X}_{\mathcal{A}}^T \mathbf{Y}$
(7) end for
(8) Output the series of coefficients $B = [\boldsymbol{\beta}^{(0)} \dots \boldsymbol{\beta}^{(P)}]$.
```

LAR: Least Angle Regression

- Least Angle Regression (LAR) is a regression method that modifies the forward selection. algorithm in only one account:
    - instead of choosing a step size which yields the (partial) least squares solution in each step, we shorten the step length such that we stop when any inactive variable becomes equally important as the active variable in terms of correlation with the residual vector.
- That variable is then included in the active set and a new direction is calculated.
- Note that all active variables are uncorrelated with the residual vector at the least squares solution, therefore the step length is always shorter or equal to the least squares solution.

LAR: Least Angle Regression

- The algorithm starts with $\mathcal{A} = \varnothing$.
- The correlation between each variable and the response is measured, and the variable with the highest correlation becomes the first variable included into the model.
- The first direction is then towards the least squares solution using this single active variable.



- Walking along this direction, the angles between the variables and the residual vector are measured. Along this walk, the angles will change;
- In particular, the correlation between the residual vector and the active variable will shrink linearly towards 0.
- At some stage before this point, another variable will obtain the same correlation with respect to the residual vector as the active variable.
- The walk stops and the new variable is added to the active set. The new direction of the walk is towards the least squares solution of the two active variables, and so on.
- After P steps, the full least squares solution will be reached.

LAR: Least Angle Regression

- Denoting the model estimate of $\mathbf{Y}$ at iteration k by $\widehat{\mathbf{Y}}^{(k)}$ and the least squares solution including the newly added active variable $\widehat{\mathbf{Y}}_{OLS}^{(k+1)}$, the walk from $\widehat{\mathbf{Y}}^{(k)}$ towards $\widehat{\mathbf{Y}}_{OLS}^{(k+1)}$ can be formulated as $(1-\gamma)\,\widehat{\mathbf{Y}}^{(k)} + \gamma\widehat{\mathbf{Y}}_{OLS}^{(k+1)}$, where $0 \le \gamma \le 1$.

- Estimating $\widehat{\mathbf{Y}}^{(k+1)}$, the position where the next active variable is to be added, then amounts to estimating γ.

- We seek the smallest positive γ where correlations become equal, that is

$$\mathbf{X}_{i\in\mathcal{I}}^{T}\left(\mathbf{Y}-(1-\gamma)\,\widehat{\mathbf{Y}}^{(k)}-\gamma\widehat{\mathbf{Y}}_{OLS}^{(k+1)}\right)=\mathbf{X}_{j\in\mathcal{A}}^{T}\left(\mathbf{Y}-(1-\gamma)\,\widehat{\mathbf{Y}}^{(k)}-\gamma\widehat{\mathbf{Y}}_{OLS}^{(k+1)}\right).$$

- Solving this expression for γ, we get, for all $j\in\mathcal{A}$,

$$\gamma_{i\in\mathcal{I}}=\frac{(\mathbf{X}_i-\mathbf{X}_j)^{T}\left(\mathbf{Y}-\widehat{\mathbf{Y}}^{(k)}\right)}{(\mathbf{X}_i-\mathbf{X}_j)^{T}\left(\widehat{\mathbf{Y}}_{OLS}^{(k+1)}-\widehat{\mathbf{Y}}^{(k)}\right)}=\frac{(\mathbf{X}_i-\mathbf{X}_j)^{T}\,\mathbf{e}}{(\mathbf{X}_i-\mathbf{X}_j)^{T}\,\mathbf{d}},$$

where $\mathbf{d}=\widehat{\mathbf{Y}}_{OLS}^{(k+1)}-\widehat{\mathbf{Y}}^{(k)}$ is the direction of the walk, and $j\in\mathcal{A}$.

LAR: Least Angle Regression

- $\mathbf{d}$ is an orthogonal projection of $\mathbf{e}$ onto the plane spanned by the variables in $\mathcal{A}$, therefore

$$\mathbf{X}_j^T\mathbf{e} = \mathbf{X}_j^T\mathbf{d} \equiv c$$

  is representing the angle at the current breakpoint $\widehat{\mathbf{Y}}^{(k)}$.

- The sign of the correlation between variables is irrelevant. Therefore, we have

$$\gamma = \min_{i\in\mathcal{I}}\left\{\frac{\mathbf{X}_i^T\mathbf{e}-c}{\mathbf{X}_i^T\mathbf{d}-c}, \frac{\mathbf{X}_i^T\mathbf{e}+c}{\mathbf{X}_i^T\mathbf{d}+c}\right\}, \quad 0<\gamma\leq 1,$$

  where the two terms are for correlations/angles of equal and opposite sign, respectively.

- The coefficients at this next step are given by

$$\boldsymbol{\beta}^{(k+\mathbf{1})} = (1-\gamma)\boldsymbol{\beta}^{(k)} + \gamma\boldsymbol{\beta}_{OLS}^{(k+\mathbf{1})}.$$

LAR algorithm is efficient since there is a closed form solution for the step length at each stage.

LAR Algorithm

```
(1) Initialize the active set $\mathcal{A} = \varnothing$ and the inactive set $\mathcal{I} = \{1, \ldots, P\}$.
(2) Initialize the coefficient vector $\boldsymbol{\beta}^{(\mathbf{0})} = 0$
(3) for $k \in \{0, \ldots, P\}$ do
    (4) Update the residual $\mathbf{e} = \mathbf{y} - \widehat{\mathbf{y}}^{(k)}$
    (5) Find the maximal correlation $c = \max_{i \in \mathcal{I}} \left\| \mathbf{X}_i^T \mathbf{e} \right\|$
    (6) Move variable corresponding to $c$ from $\mathcal{I}$ to $\mathcal{A}$.
    (7) Calculate the least squares solution $\widehat{\boldsymbol{\beta}}_{OLS}^{(k+\mathbf{1})} = \left(\mathbf{X}_{\mathcal{A}}^T \mathbf{X}_{\mathcal{A}}\right)^{-\mathbf{1}} \mathbf{X}_{\mathcal{A}}^T \mathbf{Y}$
    (8) Calculate the current direction $\mathbf{d} = \mathbf{X}_{\mathcal{A}} \widehat{\boldsymbol{\beta}}_{OLS}^{(k+\mathbf{1})} - \widehat{\mathbf{Y}}^{(k)}$
    (9) Calculate $c = \mathbf{X}_j^T \mathbf{d}$ (this value is the same for all $j \in \mathcal{A}$, one can skip this step
        because it is the same as in (5) above).
    (10) Calculate the step length $\gamma = \min_{i \in \mathcal{I}} \left\{ \frac{\mathbf{X}_i^T \mathbf{e} - c}{\mathbf{X}_i^T \mathbf{d} - c}, \frac{\mathbf{X}_i^T \mathbf{e} + c}{\mathbf{X}_i^T \mathbf{d} + c} \right\}$,
    (11) Update the regression coefficients $\boldsymbol{\beta}^{(k+\mathbf{1})} = (1 - \gamma)\boldsymbol{\beta}^{(k)} + \gamma \boldsymbol{\beta}_{OLS}^{(k+\mathbf{1})}$
    (12) Update the fitted vector $\widehat{\mathbf{Y}}^{(k+\mathbf{1})} = \widehat{\mathbf{Y}}^{(k)} + \gamma \mathbf{d}$.
(13) end for
(14) Let $\widehat{\boldsymbol{\beta}}^{(P)}$ be the full least squares solution $\widehat{\boldsymbol{\beta}}^{(P)} = \left(\mathbf{X}^T \mathbf{X}\right) \mathbf{X}^T \mathbf{Y}$
(15) Output the series of coefficients $B = [\boldsymbol{\beta}^{(\mathbf{0})} \ldots \boldsymbol{\beta}^{(P)}]$.
```

Model Selection

- Each step of LAR algorithm adds a covariate to the model until the full least squares solution is reached.
- We can parameterize this process by the size of the coefficients at each step as well as in between steps of the algorithm.
- We can return the following parametrization

$$s(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{i=1}^{P} |\beta_i| .$$

- Picking the suitable model for a particular analysis thus means selecting a suitable value of

$$s \in \left(0, \left\|\widehat{\boldsymbol{\beta}}_{OLS}\right\|_1\right) .$$

- Cross-validation is an obvious choice for this purpose, however, the algorithm provides information which substitute or complement this process.

Model Selection

- Degrees of freedom: Efron et al. (2004) showed that the number of degrees of freedom at each step of the LAR algorithm is well approximated by the number of non-zero elements of β. The algorithm therefore returns the following sequence,

$$df_{LAR}^{(k)} = |\mathcal{A}| = k, \quad \text{where } k = 0, \ldots, P.$$

- Mallow's C_p: Given the above measure of the number of degrees of freedom, we can calculate a number of model selection criteria. Mallow's C_p measure is defined as (Zou, Hastie, and Tibshirani 2007)

$$C_p^{(k)} = \frac{\left\| \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^{(k)} \right\|^2}{N} + \frac{2}{N} df^{(k)} \sigma^2.$$

- Akaike's Information Criterion (AIC): AIC is similar to Mallow's C_p and is defined by

$$AIC^{(k)} = \frac{\left\| \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^{(k)} \right\|^2}{N\sigma^2} + \frac{2}{N} df^{(k)}.$$

- Bayesian Information Cirterion (BIC): BIC tends to choose a more sparse models than AIC and C_p and it is defiend as

$$BIC^{(k)} = \frac{\left\| \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}^{(k)} \right\|^2}{N\sigma^2} + \frac{\log(N)}{N} df^{(k)}.$$

You replace σ^2 with s^2 from the OLS, and the latter three criteria can be used to pick a suitable model, typically indicated by the smallest value of each criterion.

The LASSO & Ridge Regression

- The LASSO (Tibshirani 1996) represents the most basic augmentation of the ordinary least squares solution which implements coefficients shrinkage and selection.

- The sum of squared residuals loss function $L(\boldsymbol{\beta}(\lambda))$ is combined with a penalty function $J(\boldsymbol{\beta}(\lambda))$ based on the ℓ_1 norm as,

$$\begin{aligned}\widehat{\boldsymbol{\beta}}_{LASSO}(\lambda) &= \arg\min_{\boldsymbol{\beta}} L(\boldsymbol{\beta}(\lambda)) + \lambda J(\boldsymbol{\beta}(\lambda)) \\ &= \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1 ,\end{aligned}$$

  where $\|\mathbf{x}\|^2 = \sum_{i=1}^N x_i^2$, and $\|\boldsymbol{\beta}\|_1 = \sum_{i=1}^P |\beta_i|$.

- Ridge regression (Hoerl and Kennard 1970) represents an effective way of shrinking the OLS coefficients towards zero. The ℓ_1 penalty of the LASSO is replaced with an ℓ_2 penalty,

$$\widehat{\boldsymbol{\beta}}_{Ridge}(\lambda) = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|^2 ,$$

  where $\|\mathbf{x}\|^2 = \sum_{i=1}^N x_i^2$, and $\|\boldsymbol{\beta}\|^2 = \sum_{i=1}^P \beta_i^2$.

LASSO vs. Ridge

Note that we can formulate LASSO and Ridge regressions as constrained Least Squares problems:

$$\widehat{\boldsymbol{\beta}}_{LASSO}(\lambda) = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 ,$$

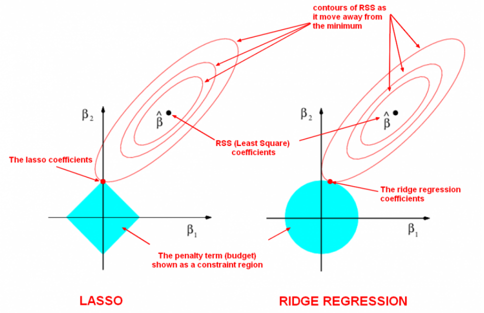
$$\text{s.t.} \quad \|\boldsymbol{\beta}\|_1 \leq c_1$$

for some $c_1 \geq 0$, and

$$\widehat{\boldsymbol{\beta}}_{Ridge}(\lambda) = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 ,$$

$$\text{s.t.} \quad \|\boldsymbol{\beta}\|^2 \leq c_2$$

for some $c_2 \geq 0$. Hence, the two methods have the following geometric interpretation which reveals the selection property of LASSO


contours of RSS as it move away from the minimum
β_2
$\hat{\beta}$
β_1
The lasso coefficients
RSS (Least Square) coefficients
The ridge regression coefficients
The penalty term (budget) shown as a constraint region
LASSO
RIDGE REGRESSION

The LASSO

- The corresponding normal equations around λ and around nearby point $\lambda + \epsilon$ are then

$$-2\mathbf{X}_{\mathcal{A}}^T\left(\mathbf{Y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) + \lambda \text{sign}\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}\right) = 0$$
$$-2\mathbf{X}_{\mathcal{A}}^T\left(\mathbf{Y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda+\epsilon)\right) + (\lambda+\epsilon) \text{sign}\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda+\epsilon)\right) = 0$$

- Using Taylor series expansion

$$f(x) = f(a) + \nabla f(a)(x-a) + \dots, \text{ with } x = \widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda+\epsilon), \; a = \widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda), \text{ and}$$
$$f\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) = -2\mathbf{X}_{\mathcal{A}}^T\left(\mathbf{Y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) + (\lambda+\epsilon)\text{sign}\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right)$$
$$\nabla f\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) = 2\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}$$
$$\nabla^{(n)} f\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) = 0 \text{ for } n = 2, 3, \dots,$$

we can write the normal equation around $\lambda + \epsilon$ as

$$-2\mathbf{X}_{\mathcal{A}}^T\left(\mathbf{Y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) + (\lambda+\epsilon)\text{sign}\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) + 2\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda+\epsilon) - \widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right) = 0$$

- Using the normal equation around λ, we can rearrange this expression to

$$\frac{\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda+\epsilon) - \widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)}{\epsilon} = -\left(2\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}\right)^{-1}\text{sign}\left(\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)\right)$$

- This approaches $\nabla\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)$ as $\epsilon \to 0$ (using $\nabla\widehat{\boldsymbol{\beta}}_{\mathcal{I}}(\lambda) = 0$).
- This is a constant function, meaning that the coefficients paths between events (changes to $\mathcal{A}$ and $\mathcal{I}$) are piecewise linear, similarly to LAR.

The LASSO

Now that an expression for the change in $\widehat{\boldsymbol{\beta}}_{\mathcal{A}}(\lambda)$ between events has been established, we focus on finding the values of λ for which changes to $\mathcal{A}$ and $\mathcal{I}$ take place. It is beneficial here to consider on an expanded set of $\boldsymbol{\beta}$-values, chosen such that

$$\beta_j = \beta_j^+ - \beta_j^-, \text{ and } |\beta_j| = \beta_j^+ + \beta_j^-,$$

where $\beta_j^+ \geq 0$ and $\beta_j^- \geq 0$ for all j.

$$\begin{aligned}
\widehat{\boldsymbol{\beta}}_{LASSO}(\lambda) &= \arg\min_{\boldsymbol{\beta}} L(\boldsymbol{\beta}(\lambda)) + \lambda J(\boldsymbol{\beta}(\lambda)) \\
&= \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1, \\
&= \arg\min_{\boldsymbol{\beta}^+, \boldsymbol{\beta}^-} L(\boldsymbol{\beta}(\lambda)) + \lambda\left(\boldsymbol{\beta}^+ + \boldsymbol{\beta}^-\right), \\
&\text{s.t. } \boldsymbol{\beta}^+ \geq 0 \text{ and } \boldsymbol{\beta}^- \geq 0.
\end{aligned}$$

This formulation of LASSO is differentiable at the price of having to deal with twice as many variables. The Lagrange primal function is

$$\begin{aligned}
\widehat{\boldsymbol{\beta}}_{LASSO}(\lambda) &= \arg\min_{\boldsymbol{\beta}^+, \boldsymbol{\beta}^-} L(\boldsymbol{\beta}(\lambda)) + \lambda\left(\boldsymbol{\beta}^+ + \boldsymbol{\beta}^-\right) - \sum_{j=1}^{P} \lambda_j^+ \beta_j^+ - \sum_{j=1}^{P} \lambda_j^- \beta_j^- \\
&\lambda_j^+ \geq 0 \text{ and } \lambda_j^- \geq 0 \text{ for all } j.
\end{aligned}$$

The LASSO

The Karush-Kuhn-Tucker conditions are

$$\begin{aligned}\nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j + \lambda - \lambda_j^+ &= 0\\ -\nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j + \lambda - \lambda_j^- &= 0\\ \lambda_j^+\beta_j^+ &= 0\\ \lambda_j^-\beta_j^- &= 0,\end{aligned}$$

where $\nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j = -2\mathbf{X}_j^T\left(\mathbf{Y} - \mathbf{X}^T\boldsymbol{\beta}\left(\lambda\right)\right)$.

From these conditions a number of useful properties arise:

- Setting $\lambda = 0$ indeed gives us $\nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j = 0$, for all $j \in \mathcal{A}$
- For positive values of λ we have

$$\begin{aligned}\beta_j^+ > 0 \Rightarrow \lambda_j^+ = 0 \Rightarrow \nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j = -\lambda \Rightarrow \lambda_j^- > 0 \Rightarrow \beta_j^- = 0\\ \beta_j^- > 0 \Rightarrow \lambda_j^- = 0 \Rightarrow \nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j = \lambda \Rightarrow \lambda_j^+ > 0 \Rightarrow \beta_j^+ = 0.\end{aligned}$$

  I.e. elements in $\mathcal{A}$ have either $\beta_j^+ > 0$ or $\beta_j^- > 0$, but cannot both be non-zero.
- Hence,

$$\begin{aligned}\left|\nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j\right| &= \lambda, \text{ for } j \in \mathcal{A}\\ \left|\nabla\left(L\left(\boldsymbol{\beta}\left(\lambda\right)\right)\right)_j\right| &< \lambda, \text{ for } j \in \mathcal{I}.\end{aligned}$$

The LASSO

We are seeking a value of $\gamma \geq 0$ for which the variable in $\mathcal{I}$ joins $\mathcal{A}$ or vice versa. We have arrived at the following conditions

$$j \in \mathcal{A} \to \mathcal{I} : \widehat{\beta}_j^{(k)} + \gamma \nabla \widehat{\beta}_j^{(k)} = 0 \quad \text{for } j \in \mathcal{A}$$

which defines distances $\{\gamma\}$ at which active variables hit zero and join $\mathcal{I}$, and

$$j \in \mathcal{I} \to \mathcal{A} : \left| \left(\nabla L \left(\widehat{\boldsymbol{\beta}}^{(k)} + \gamma \nabla \widehat{\boldsymbol{\beta}}^{(k)} \right) \right)_i \right| = \left| \left(\nabla L \left(\widehat{\boldsymbol{\beta}}^{(k)} + \gamma \nabla \widehat{\boldsymbol{\beta}}^{(k)} \right) \right)_j \right|, \text{ for } \; j \in \mathcal{A}, \; i \in \mathcal{I}$$

which defines distances at which inactive variables violate the condition $\left| \nabla \left(L \left(\boldsymbol{\beta} \left(\lambda \right) \right) \right)_j \right| < \lambda$ and thus must join $\mathcal{A}$.

Comments:

- Note that any element in $\mathcal{A}$ can be used to calculate λ.
- The smallest value $\gamma_{\mathbf{min}}$ of the distances $\{\gamma\}$ is where the next event will happen.

The coefficients can now be updated by

$$\widehat{\boldsymbol{\beta}}^{(k+\mathbf{1})} = \widehat{\boldsymbol{\beta}}^{(k)} + \gamma_{\mathbf{min}} \nabla \widehat{\boldsymbol{\beta}}^{(k)}.$$

We have arrived at LASSO algorithm.

The LASSO Algorithm

```
(1)  Initialize $\boldsymbol{\beta}^{(\mathbf{0})} = \mathbf{0}$, $\mathcal{A} = \arg\max_j \left|\mathbf{X}_j^T \mathbf{Y}\right|$, $\nabla\widehat{\boldsymbol{\beta}}_{\mathcal{A}}^{(\mathbf{0})} = -\text{sign}\left(\mathbf{X}_{\mathcal{A}}^T \mathbf{Y}\right)$, $\nabla\widehat{\boldsymbol{\beta}}_{\mathcal{I}}^{(\mathbf{0})} = 0$, $k = 0$
(2)  while $\mathcal{I} \neq \emptyset$ do
         (3)  $\gamma_j = \min^+_{j\in\mathcal{A}} -\widehat{\beta}_j^{(k)} / \nabla\widehat{\beta}_j^{(k)}$
         (4)  $\gamma_i = \min^+_{i\in\mathcal{I}} \left\{ \frac{\left(\mathbf{x}_i + \mathbf{x}_j\right)^T \left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}^{(k)}\right)}{\left(\mathbf{x}_i + \mathbf{x}_j\right)^T \left(\mathbf{X}\nabla\widehat{\boldsymbol{\beta}}^{(k)}\right)}, \frac{\left(\mathbf{x}_i - \mathbf{x}_j\right)^T \left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}^{(k)}\right)}{\left(\mathbf{x}_i - \mathbf{x}_j\right)^T \left(\mathbf{X}\nabla\widehat{\boldsymbol{\beta}}^{(k)}\right)} \right\}$, for any $j \in \mathcal{A}$.
         (5)  $\gamma = \min\{\gamma_i, \gamma_j\}$
         (6)  if $\gamma = \gamma_j$, then move $j$ from $\mathcal{A}$ to $\mathcal{I}$, else move $i$ from $\mathcal{I}$ to $\mathcal{A}$.
         (7)  $\widehat{\boldsymbol{\beta}}^{(k+\mathbf{1})} = \widehat{\boldsymbol{\beta}}^{(k)} + \gamma\nabla\widehat{\boldsymbol{\beta}}^{(k)}$.
         (8)  $\nabla\widehat{\boldsymbol{\beta}}_{\mathcal{A}}^{(k+\mathbf{1})} = -\left(2\mathbf{X}_{\mathcal{A}}^T \mathbf{X}_{\mathcal{A}}\right)^{-\mathbf{1}} \text{sign}\left(\widehat{\beta}_{\mathcal{A}}^{(k+\mathbf{1})}\right)$
         (9)  $k = k + 1$
(10) end while loop
(11) Output the series of coefficients $\mathbf{B} = \left[\widehat{\boldsymbol{\beta}}^{(\mathbf{0})}, \ldots, \widehat{\boldsymbol{\beta}}^{(k)}\right]$
```

Coefficients Paths of LAR vs. LASSO

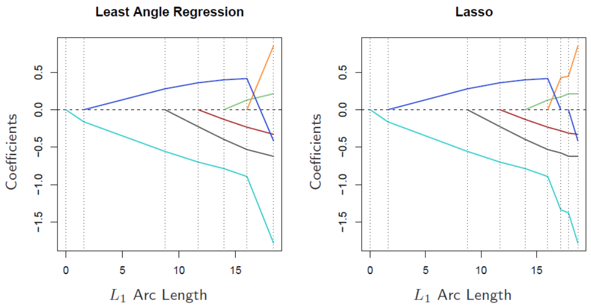


FIGURE 3.15. *Left panel shows the LAR coefficient profiles on the simulated data, as a function of the L_1 arc length. The right panel shows the Lasso profile. They are identical until the dark-blue coefficient crosses zero at an arc length of about 18.*

(Source: ***The elements of statistical learning*** **by Hastie, Tibshirani, and Friedman)**

Comparison of Four Subset Selection Techniques

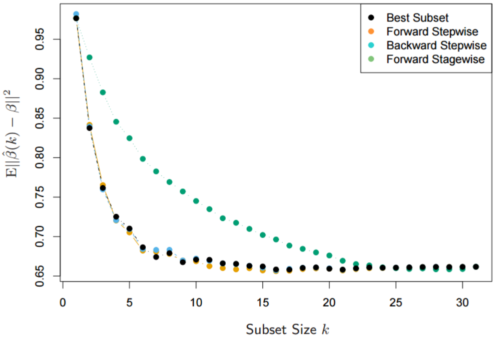


FIGURE 3.6. *Comparison of four subset-selection techniques on a simulated linear regression problem $Y = X^T\beta + \varepsilon$. There are $N = 300$ observations on $p = 31$ standard Gaussian variables, with pairwise correlations all equal to 0.85. For 10 of the variables, the coefficients are drawn at random from a $N(0, 0.4)$ distribution; the rest are zero. The noise $\varepsilon \sim N(0, 6.25)$, resulting in a signal-to-noise ratio of 0.64. Results are averaged over 50 simulations. Shown is the mean-squared error of the estimated coefficient $\hat{\beta}(k)$ at each step from the true β.*

(Source: *The elements of statistical learning* by Hastie, Tibshirani, and Friedman)

Comparison of LAR and LASSO vs. Forward Stepwise Regression

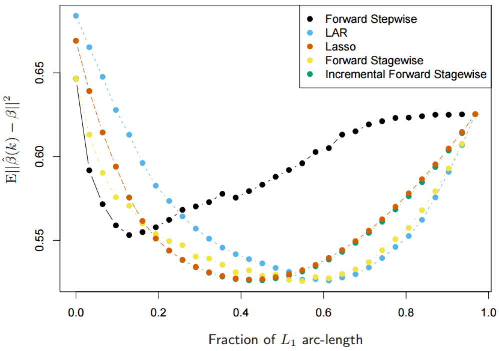


FIGURE 3.16. *Comparison of LAR and lasso with forward stepwise, forward stagewise (FS) and incremental forward stagewise (FS_0) regression. The setup is the same as in Figure 3.6, except $N = 100$ here rather than 300. Here the slower FS regression ultimately outperforms forward stepwise. LAR and lasso show similar behavior to FS and FS_0. Since the procedures take different numbers of steps (across simulation replicates and methods), we plot the MSE as a function of the fraction of total L_1 arc-length toward the least-squares fit.*

(Source: *The elements of statistical learning* by Hastie, Tibshirani, and Friedman)

LASSO, Ridge regression & Elastic Net

- The LASSO (Tibshirani 1996) represents the most basic augmentation of the ordinary least squares solution which implements coefficients shrinkage and selection. The sum of squared residuals loss function $L(\boldsymbol{\beta}(\lambda))$ is combined with a penalty function $J(\boldsymbol{\beta}(\lambda))$ based on the ℓ_1 norm as,

$$\begin{aligned}\widehat{\boldsymbol{\beta}}_{LASSO}(\lambda) &= \arg\min_{\boldsymbol{\beta}} L(\boldsymbol{\beta}(\lambda)) + \lambda J(\boldsymbol{\beta}(\lambda)) \\ &= \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1 ,\end{aligned}$$

  where $\|\mathbf{x}\|^2 = \sum_{i=1}^N x_i^2$, and $\|\boldsymbol{\beta}\|_1 = \sum_{i=1}^P |\beta_i|$.
- Ridge regression (Hoerl and Kennard 1970) represents an effective way of shrinking the OLS coefficients towards zero. The ℓ_1 penalty of the LASSO is replaced with an ℓ_2 penalty,

$$\widehat{\boldsymbol{\beta}}_{Ridge}(\lambda) = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|^2 ,$$

  where $\|\mathbf{x}\|^2 = \sum_{i=1}^N x_i^2$, and $\|\boldsymbol{\beta}\|^2 = \sum_{i=1}^P \beta_i^2$.
    - LASSO induces sparse solutions but the benefit of Ridge regression is that a unique solution is available, which even has a closed form

$$\widehat{\boldsymbol{\beta}}_{Ridge}(\lambda) = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{Y}$$

      also when the data matrix $\mathbf{X}$ is rank deficient, e.g., when there are more predictors than observations ($P > N$).
    - The LASSO algorithm is terminated when the active set size $|\mathcal{A}|$ becomes larger than P since the matrix $\mathbf{X}_{\mathcal{A}}^T\mathbf{X}_{\mathcal{A}}$ in (8) on "The LASSO Algorithm" slide is no longer invertible.

LASSO, Ridge regression & Elastic Net

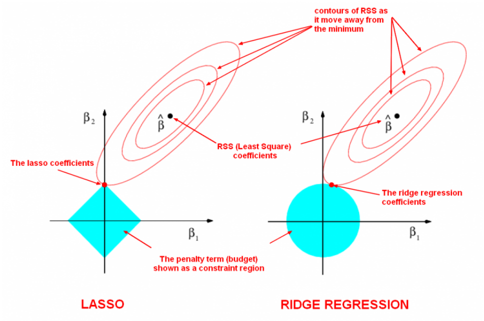
Elastic net regression (Zou and Hastie 2005) combines the virtues of ridge regression and the LASSO by considering solutions penalized by both an ℓ_2 and an ℓ_1 term,

$$\widehat{\boldsymbol{\beta}}_{ENet}(\lambda_1, \lambda_2) = \arg\min_{\boldsymbol{\beta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\boldsymbol{\beta}\|^2 ,$$

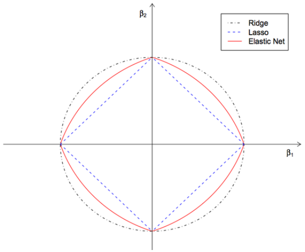
thus bridging the gap between the LASSO ($\lambda_2 = 0$) and ridge regression ($\lambda_1 = 0$).

- The ℓ_2 penalty ensures a unique solution also when $P > N$.
- The ℓ_1 penalty offers variable selection via a sparse vector of coefficients $\widehat{\boldsymbol{\beta}}_{ENet}$.
- The ℓ_2 leads to a grouping effect (Zou and Hastie 2005), a term that alludes to the characteristic that highly correlated predictors tend to have similar regression coefficients for nonzero λ_2.
- For fixed λ_2, we can use the LASSO algorithm to obtain the full regularization path of elastic net solutions.

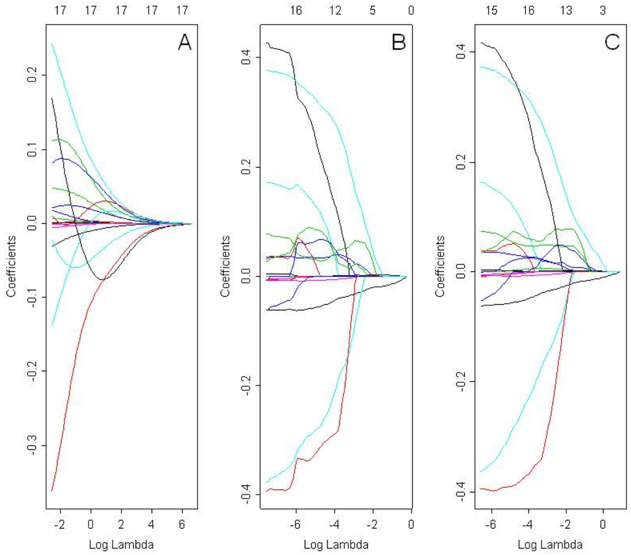
LASSO, Ridge regression & Elastic Net



contours of RSS as it move away from the minimum
β_2
$\hat{\beta}$
RSS (Least Square) coefficients
The lasso coefficients
β_1
The penalty term (budget) shown as a constraint region
The ridge regression coefficients
LASSO
RIDGE REGRESSION



Coefficients Paths: LASSO, Ridge regression & Elastic Net



A - Ridge; B - LASSO; C - Elastic Net.

Three Examples

True model

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \text{ where } \boldsymbol{\varepsilon} \sim \mathrm{N}(0, \mathbf{I}_N),$$

where

(1) Small signal a lot of noise
- $\boldsymbol{\beta} = (\underbrace{1, \ldots, 1}_{15}, \underbrace{0, \ldots, 0}_{4985})$,
- $P = 5000 > N = 1000$.
- uncorrelated predictors $\mathbf{X} \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0, \mathbf{I}_P)$

(2) Big signal and big noise
- $\boldsymbol{\beta} = (\underbrace{1, \ldots, 1}_{1500}, \underbrace{0, \ldots, 0}_{3500})$,
- $P = 5000 > N = 1000$ (note that LASSO can pick at most 1000 predictors).
- uncorrelated predictors $\mathbf{X} \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0, \mathbf{I}_P)$

(3) Varying signal, high correlation between predictors
- $\boldsymbol{\beta} = (10, 10, 5, 5, \underbrace{1, \ldots, 1}_{10}, \underbrace{0, \ldots, 0}_{36})$,
- $P = 50 < N = 100$.
- correlated predictors $\mathrm{Cov}(\mathbf{X}_i, \mathbf{X}_j) = 0.7^{|i-j|}$

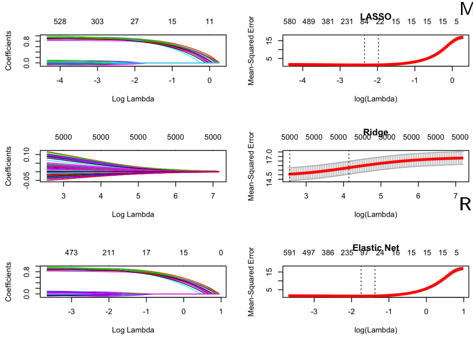
For each of these models, we will compare LASSO, Ridge, and Elastic Net in terms of the *coefficients paths* and the *prediction performance*.

Example 1: R code (using glmnet package)

```r
# Generate data
set.seed(19875)  # Set seed for reproducibility
n <- 1000  # Number of observations
p <- 5000  # Number of predictors included in model
real_p <- 15  # Number of true predictors
x <- matrix(rnorm(n*p), nrow=n, ncol=p)
y <- apply(x[,1:real_p], 1, sum) + rnorm(n)

# Split data into train (2/3) and test (1/3) sets
train_rows <- sample(1:n, .66*n)
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]

y.train <- y[train_rows]
y.test <- y[-train_rows]
# Fit models
# (For plots on left):
fit.lasso <- glmnet(x.train, y.train, family="gaussian", alpha=1)
fit.ridge <- glmnet(x.train, y.train, family="gaussian", alpha=0)
fit.elnet <- glmnet(x.train, y.train, family="gaussian", alpha=.5)


# 10-fold Cross validation for each alpha = 0, 0.1, ... , 0.9, 1.0
# (For plots on Right)
for (i in 0:10) {
    assign(paste("fit", i, sep=""), cv.glmnet(x.train, y.train, type.measure="mse",
                                              alpha=i/10,family="gaussian"))
}
```

```r
# Plot solution paths:
par(mfrow=c(3,2))
# For plotting options, type '?plot.glmnet' in R console
plot(fit.lasso, xvar="lambda")
plot(fit10, main="LASSO")

plot(fit.ridge, xvar="lambda")
plot(fit0, main="Ridge")

plot(fit.elnet, xvar="lambda")
plot(fit5, main="Elastic Net")
yhat0 <- predict(fit0, s=fit0$lambda.1se, newx=x.test)
yhat1 <- predict(fit1, s=fit1$lambda.1se, newx=x.test)
yhat2 <- predict(fit2, s=fit2$lambda.1se, newx=x.test)
yhat3 <- predict(fit3, s=fit3$lambda.1se, newx=x.test)
yhat4 <- predict(fit4, s=fit4$lambda.1se, newx=x.test)
yhat5 <- predict(fit5, s=fit5$lambda.1se, newx=x.test)
yhat6 <- predict(fit6, s=fit6$lambda.1se, newx=x.test)
yhat7 <- predict(fit7, s=fit7$lambda.1se, newx=x.test)
yhat8 <- predict(fit8, s=fit8$lambda.1se, newx=x.test)
yhat9 <- predict(fit9, s=fit9$lambda.1se, newx=x.test)
yhat10 <- predict(fit10, s=fit10$lambda.1se, newx=x.test)
mse0 <- mean((y.test - yhat0)^2)
mse1 <- mean((y.test - yhat1)^2)
mse2 <- mean((y.test - yhat2)^2)
mse3 <- mean((y.test - yhat3)^2)
mse4 <- mean((y.test - yhat4)^2)
mse5 <- mean((y.test - yhat5)^2)
mse6 <- mean((y.test - yhat6)^2)
mse7 <- mean((y.test - yhat7)^2)
mse8 <- mean((y.test - yhat8)^2)
mse9 <- mean((y.test - yhat9)^2)
mse10 <- mean((y.test - yhat10)^2)
```

Example 1: Coefficients Paths and MSE



α	MSE
$\alpha = 0$ (Ridge)	16.1313
$\alpha = 0.2$	1.8063
$\alpha = 0.4$	1.4351
$\alpha = 0.6$	1.4146
$\alpha = 0.8$	1.4427
$\alpha = 1$ (LASSO)	1.3759

Model:

(1) Small signal a lot of noise

- $\boldsymbol{\beta} = (\underbrace{1, \dots, 1}_{15}, \underbrace{0, \dots, 0}_{4085}),$
- $P = 5000 > N = 1000.$
- uncorrelated predictors $\mathbf{X} \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0, \mathbf{I}_P)$

Results:

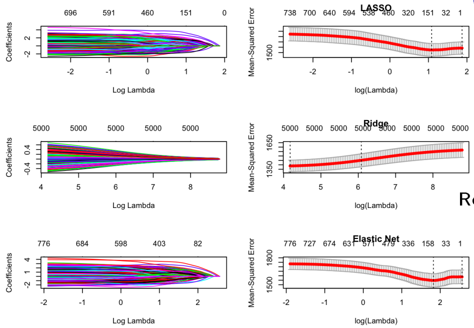
- Coefficients paths for LASSO and Elastic Net look quite similar.
- For sufficiently large λ, LASSO and Elastic Net shrink some of the coefficients to 0.
- Coefficients from Ridge regression smoothly approach 0 but remain always different from 0.
- The prediction performance evaluated on the test set reveals that LASSO is the best performing model in terms of the forecasting of $\mathbf{Y}$.

Example 2: R code (using glmnet package)

```r
# Generate data
set.seed(19874)
n <- 1000    # Number of observations
p <- 5000     # Number of predictors included in model
real_p <- 1500  # Number of true predictors
x <- matrix(rnorm(n*p), nrow=n, ncol=p)
y <- apply(x[,1:real_p], 1, sum) + rnorm(n)

# Split data into train and test sets
train_rows <- sample(1:n, .66*n)
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]

y.train <- y[train_rows]
y.test <- y[-train_rows]
# Fit models:
fit.lasso <- glmnet(x.train, y.train, family="gaussian", alpha=1)
fit.ridge <- glmnet(x.train, y.train, family="gaussian", alpha=0)
fit.elnet <- glmnet(x.train, y.train, family="gaussian", alpha=.5)


# 10-fold Cross validation for each alpha = 0, 0.1, ... , 0.9, 1.0
fit.lasso.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=1,
                          family="gaussian")
fit.ridge.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=0,
                          family="gaussian")
fit.elnet.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=.5,
                          family="gaussian")

for (i in 0:10) {
    assign(paste("fit", i, sep=""), cv.glmnet(x.train, y.train, type.measure="mse",
                                              alpha=i/10,family="gaussian"))
}
```

```r
# Plot solution paths:
par(mfrow=c(3,2))
# For plotting options, type '?plot.glmnet' in R console
plot(fit.lasso, xvar="lambda")
plot(fit10, main="LASSO")

plot(fit.ridge, xvar="lambda")
plot(fit0, main="Ridge")

plot(fit.elnet, xvar="lambda")
plot(fit5, main="Elastic Net")
yhat0 <- predict(fit0, s=fit0$lambda.1se, newx=x.test)
yhat1 <- predict(fit1, s=fit1$lambda.1se, newx=x.test)
yhat2 <- predict(fit2, s=fit2$lambda.1se, newx=x.test)
yhat3 <- predict(fit3, s=fit3$lambda.1se, newx=x.test)
yhat4 <- predict(fit4, s=fit4$lambda.1se, newx=x.test)
yhat5 <- predict(fit5, s=fit5$lambda.1se, newx=x.test)
yhat6 <- predict(fit6, s=fit6$lambda.1se, newx=x.test)
yhat7 <- predict(fit7, s=fit7$lambda.1se, newx=x.test)
yhat8 <- predict(fit8, s=fit8$lambda.1se, newx=x.test)
yhat9 <- predict(fit9, s=fit9$lambda.1se, newx=x.test)
yhat10 <- predict(fit10, s=fit10$lambda.1se, newx=x.test)
mse0 <- mean((y.test - yhat0)^2)
mse1 <- mean((y.test - yhat1)^2)
mse2 <- mean((y.test - yhat2)^2)
mse3 <- mean((y.test - yhat3)^2)
mse4 <- mean((y.test - yhat4)^2)
mse5 <- mean((y.test - yhat5)^2)
mse6 <- mean((y.test - yhat6)^2)
mse7 <- mean((y.test - yhat7)^2)
mse8 <- mean((y.test - yhat8)^2)
mse9 <- mean((y.test - yhat9)^2)
```

Example 2: Coefficients Paths and MSE



α	MSE
$\alpha = 0$ (Ridge)	1490.7833
$\alpha = 0.2$	1637.0238
$\alpha = 0.4$	1641.1414
$\alpha = 0.6$	1633.5475
$\alpha = 0.8$	1641.1414
$\alpha = 1$ (LASSO)	1641.1414

Model:

(2) Big signal and big noise

- $\beta = (\underbrace{1, \ldots, 1}_{1500}, \underbrace{0, \ldots, 0}_{3500})$,
- $P = 5000 > N = 1000$ (note that LASSO can pick at most 1000 predictors).
- uncorrelated predictors $\mathbf{X} \overset{\text{i.i.d.}}{\sim} \mathrm{N}(0, \mathbf{I}_P)$

Results:

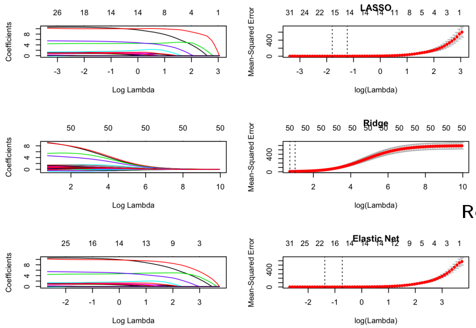
- Coefficients paths for LASSO and Elastic Net are different (different coefficients enter at different moments).
- For sufficiently large λ, LASSO and Elastic Net shrink some of the coefficients to 0.
- Coefficients from Ridge regression smoothly approach 0 but remain always different from 0.
- The prediction performance evaluated on the test set reveals that Ridge is the best performing model in terms of the forecasting of $\mathbf{Y}$.

Example 3: R code (using glmnet package)

```r
# Generate data
set.seed(19873)
n <- 100    # Number of observations
p <- 50     # Number of predictors included in model
CovMatrix <- outer(1:p, 1:p, function(x,y) {.7^abs(x-y)})
x <- mvrnorm(n, rep(0,p), CovMatrix)
y <- 10 * apply(x[, 1:2], 1, sum) +
  5 * apply(x[, 3:4], 1, sum) +
  apply(x[, 5:14], 1, sum) +
  rnorm(n)

# Split data into train and test sets
train_rows <- sample(1:n, .66*n)
x.train <- x[train_rows, ]
x.test <- x[-train_rows, ]

y.train <- y[train_rows]
y.test <- y[-train_rows]
# Fit models:
fit.lasso <- glmnet(x.train, y.train, family="gaussian", alpha=1)
fit.ridge <- glmnet(x.train, y.train, family="gaussian", alpha=0)
fit.elnet <- glmnet(x.train, y.train, family="gaussian", alpha=.5)


# 10-fold Cross validation for each alpha = 0, 0.1, ... , 0.9, 1.0
fit.lasso.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=1,
                          family="gaussian")
fit.ridge.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=0,
                          family="gaussian")
fit.elnet.cv <- cv.glmnet(x.train, y.train, type.measure="mse", alpha=.5,
                          family="gaussian")

for (i in 0:10) {
    assign(paste("fit", i, sep=""), cv.glmnet(x.train, y.train, type.measure="mse",
                                              alpha=i/10,family="gaussian"))
}
```

```r
# Plot solution paths:
par(mfrow=c(3,2))
# For plotting options, type '?plot.glmnet' in R console
plot(fit.lasso, xvar="lambda")
plot(fit10, main="LASSO")

plot(fit.ridge, xvar="lambda")
plot(fit0, main="Ridge")

plot(fit.elnet, xvar="lambda")
plot(fit5, main="Elastic Net")
yhat0 <- predict(fit0, s=fit0$lambda.1se, newx=x.test)
yhat1 <- predict(fit1, s=fit1$lambda.1se, newx=x.test)
yhat2 <- predict(fit2, s=fit2$lambda.1se, newx=x.test)
yhat3 <- predict(fit3, s=fit3$lambda.1se, newx=x.test)
yhat4 <- predict(fit4, s=fit4$lambda.1se, newx=x.test)
yhat5 <- predict(fit5, s=fit5$lambda.1se, newx=x.test)
yhat6 <- predict(fit6, s=fit6$lambda.1se, newx=x.test)
yhat7 <- predict(fit7, s=fit7$lambda.1se, newx=x.test)
yhat8 <- predict(fit8, s=fit8$lambda.1se, newx=x.test)
yhat9 <- predict(fit9, s=fit9$lambda.1se, newx=x.test)
yhat10 <- predict(fit10, s=fit10$lambda.1se, newx=x.test)
mse0 <- mean((y.test - yhat0)^2)
mse1 <- mean((y.test - yhat1)^2)
mse2 <- mean((y.test - yhat2)^2)
mse3 <- mean((y.test - yhat3)^2)
mse4 <- mean((y.test - yhat4)^2)
mse5 <- mean((y.test - yhat5)^2)
mse6 <- mean((y.test - yhat6)^2)
mse7 <- mean((y.test - yhat7)^2)
mse8 <- mean((y.test - yhat8)^2)
mse9 <- mean((y.test - yhat9)^2)
mse10 <- mean((y.test - yhat10)^2)
```

Example 3: Coefficients Paths and MSE



α	MSE
$\alpha = 0$ (Ridge)	6.7541
$\alpha = 0.2$	1.2461
$\alpha = 0.4$	1.4948
$\alpha = 0.6$	1.4219
$\alpha = 0.8$	1.4985
$\alpha = 1$ (LASSO)	1.7152

Model:

(3) Varying signal, high correlation between predictors

- $\boldsymbol{\beta} = (10, 10, 5, 5, \underbrace{1, \ldots, 1}_{10}, \underbrace{0, \ldots, 0}_{36})$,
- $P = 50 < N = 100$.
- correlated predictors $\text{Cov}(\mathbf{X}_i, \mathbf{X}_j) = 0.7^{|i-j|}$

Results:

- Coefficients paths for Elastic Net are smoother than for LASSO.
- For sufficiently large λ, LASSO and Elastic Net shrink some of the coefficients to 0.
- Coefficients from Ridge regression smoothly approach 0 but remain always different from 0.
- The prediction performance evaluated on the test set reveals that Elastic Net, with $\alpha = 0.2$, is the best performing model in terms of the forecasting of $\mathbf{Y}$.