23

LINEAR MODELS FOR REGRESSION

23 LINEAR MODELS FOR REGRESSION 1268

23.1 Standard linear regression 1269

23.1.1 Ordinary linear regression 1269

23.1.2 Application examples 1270

23.1.2.1 Boston housing prices 1270

23.2 Penalized linear regression 1274

23.2.1 Ridge regression 1274

23.2.1.1 Basics 1274

23.2.1.2 Dual form of ridge regression 1277

23.2.2 Lasso regression 1277

23.2.3 Elastic net 1280

23.2.4 Shrinkage Comparison 1280

23.2.5 Effective degree of freedom 1283

23.3 Basis function extension 1284

23.4 Note on bibliography 1286

23.1 Standard linear regression

23.1.1 Ordinary linear regression

In the canonical linear regression model [section 15.1], we usually have the following setup:

Definition 23.1.1 (multiple linear regression model, recap). *Definition 15.1.2 The multiple linear regression model* ***assumes*** *that a random variable Y has a linear dependency on a non-random vector $X = (X_1, X_2, ..., X_{p-1}) \in \mathbb{R}^{p-1}$ given as*

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_{p-1} X_{p-1} + \epsilon$$

where $\beta_0, \beta_1, ..., \beta_p$ are unknown model parameters, and ϵ is a random variable. Given the observed sample pairs $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n), x \in \mathbb{R}^{p-1}, y \in \mathbb{R}$ as $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \epsilon_i$ and we ***further make the following assumptions on*** *ϵ as*

- *$E[\epsilon_i] = 0, \forall i$*
- *$cov(\epsilon_i, \epsilon_j) = \sigma^2 \delta_{ij}$ and σ^2 is unknown.*

The task of finding coefficients β and σ can be formulated as a least square minimization problem [Theorem 15.1.1] given by

$$\min \left\| Y - X\beta \right\|^2$$

where the coefficient vector is $\beta = (\beta_0, \beta_1, ..., \beta_{p-1})^T$, the design matrix X is row stack every sample(each row is an observation on $X_1, X_2, ..., X_n$), the minimizer β is given by

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

Different from previous statistical treatment in [section 15.1], in the machine learning context here, we are more concerned with **high-dimensional regression**, where the number of predictors far exceed the number of samples. There are several issues arising in the ordinary linear regression approach:

Remark 23.1.1 (rank deficiency results and solution).

- Suppose $p \geq n$ (dimensionality of input equals or greater then the number of samples). $X^T X$ is not invertible and $\hat{\beta}$ cannot be evaluated.
- Even when $n \geq p$ and $X^T X$ is invertible, the test error is approximately given by $p\sigma^2/n$ [Lemma 22.2.2], which can be large when $p \gg 1$.
- When the number of predictors is large, the resulting model will lack interpretability. Instead, we may prefer models with a smaller set of important predictors.

- We can reduce the feature dimensionality by performing PCA regression [Methodology 15.3.2].

23.1.2 Application examples

23.1.2.1 *Boston housing prices*

One classical predictive modeling problem is the Boston housing prices problem. We are given 506 samples and 13 feature variables and the goal is to model the relationship between features and the prices. The features and target variable are listed below.

- CRIM: Per capita crime rate by town
- ZN: Proportion of residential land zoned for lots over 25,000 sq. ft
- INDUS: Proportion of non-retail business acres per town.
- CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX: Nitric oxide concentration (parts per 10 million)
- RM: Average number of rooms per dwelling
- AGE: Proportion of owner-occupied units built prior to 1940
- DIS: Weighted distances to five Boston employment centers
- RAD: Index of accessibility to radial highways
- TAX: Full-value property tax rate per $10,000
- PTRATIO: Pupil-teacher ratio by town
- B: $1000(Bk^{-}0.63)^2$, where Bk is the proportion of [people of African American descent] by town.
- LSTAT: Percentage of lower status of the population
- **(target)** MEDV: Median value of owner-occupied homes in $1000

After some initial data preprocessing data (for example, we drop the feature variable ZN and CHAS because more than 50 percent are missing), we perform correlation analysis among features and price [Figure 23.1.1].

Correlation analysis shows that RM has the highest positive correlation with MEDV(0.7) whereas LSTAT has a highest negative correlation with MEDV(-0.74). t static analysis of the linear regression results also indicate the two features play important roles in determining housing prices. An more comprehensive exploratory analysis is in Figure 23.1.2.

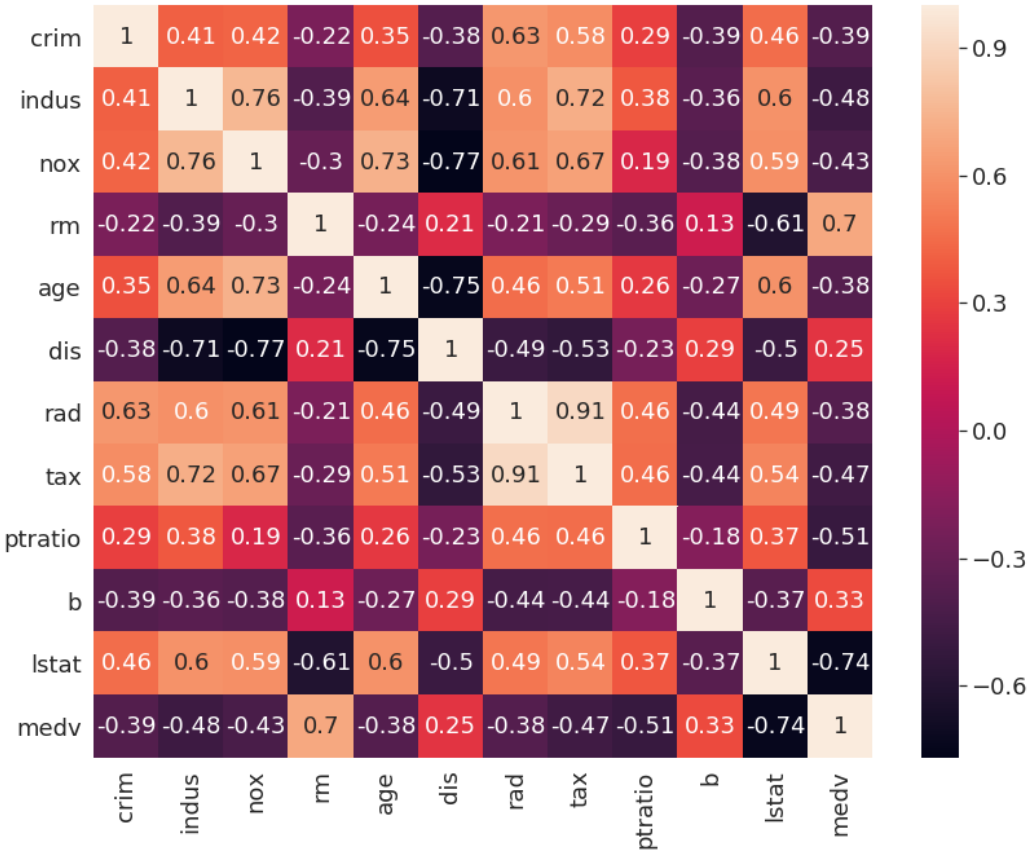


Figure 23.1.1: Correlation among features and the label MEDV.

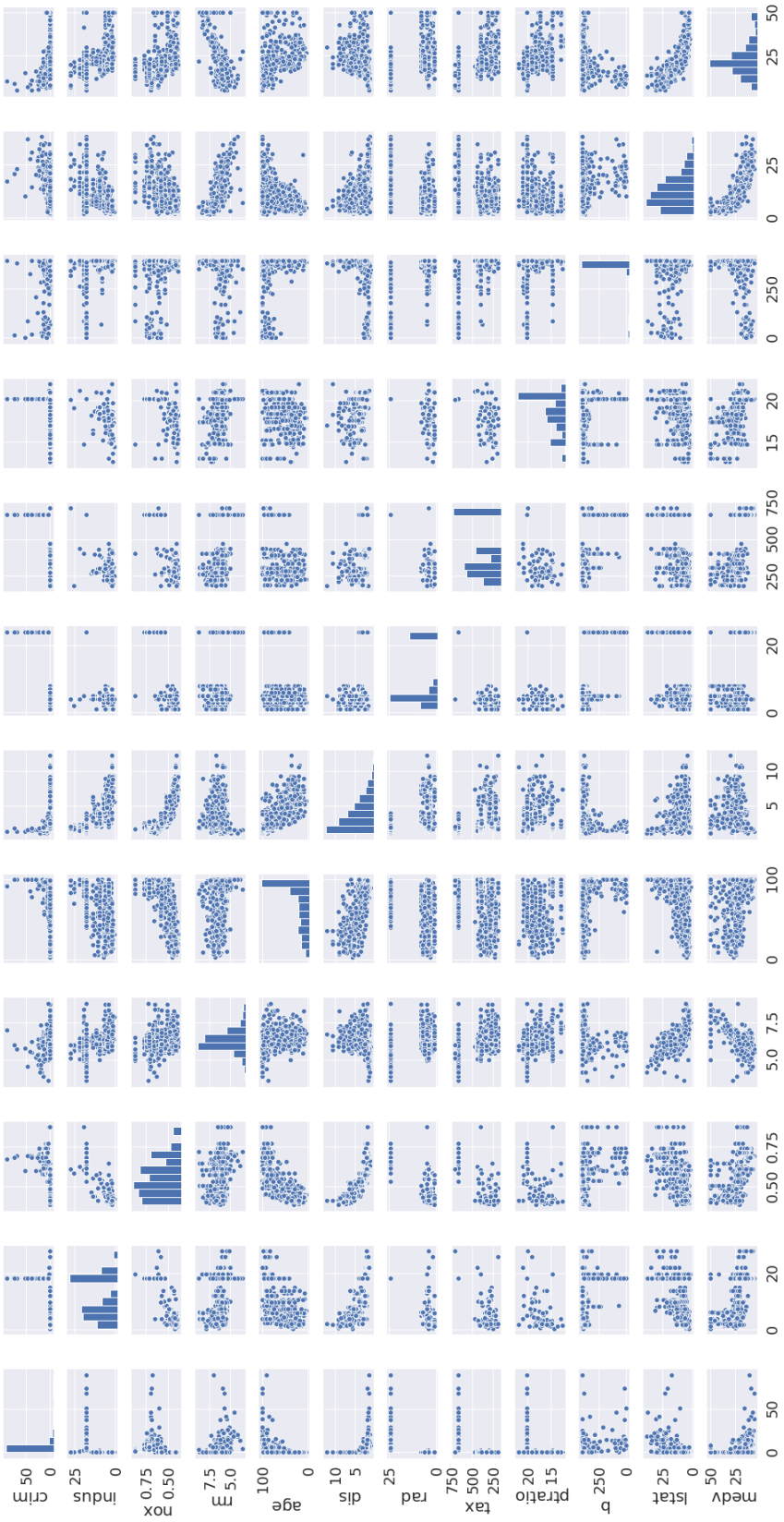


Figure 23.1.2: Pair plot among features and the label.

The linear regression results are showed below. The mean square error is 5.34. t statics also shows that lstat and rm are most significant variables.

variables	coef	std err	t	P>|t|	[0.025	0.975]
const	37.3083	5.2	7.175	0	27.092	47.525
crim	-0.1034	0.033	-3.102	0.002	-0.169	-0.038
indus	0.0182	0.062	0.294	0.769	-0.104	0.14
nox	-17.8292	3.89	-4.584	0	-25.472	-10.187
rm	4.0744	0.421	9.686	0	3.248	4.901
age	-0.0026	0.013	-0.198	0.843	-0.029	0.024
dis	-1.2102	0.186	-6.502	0	-1.576	-0.844
rad	0.3046	0.067	4.555	0	0.173	0.436
tax	-0.0109	0.004	-2.939	0.003	-0.018	-0.004
ptratio	-1.1311	0.126	-8.972	0	-1.379	-0.883
b	0.0099	0.003	3.603	0	0.004	0.015
lstat	-0.5251	0.052	-10.187	0	-0.626	-0.424

Table 23.1.1: Linear regression results.

23.2 Penalized linear regression

23.2.1 Ridge regression

23.2.1.1 *Basics*

Although OLS estimator has the desired property of being unbiased, it can also suffer from the rank deficiency problem and the multi-collinearity problem [Remark 23.1.1] that lead to a huge variance in its estimate and thus it prediction performance.

The second reason is interpretation. With a large number of predictors, we often would like to determine a smaller subset that exhibit the strongest effects. In order to get the essential picture, we are willing to sacrifice some of the small details.

In ridge regression, the goal is to estimate $\hat{\beta}_{ridge}$ by minimizing

$$\min_{\beta} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_2^2, \lambda \geq 0,$$

the additional term $\lambda \|\beta\|_2^2$ penalize the estimated coefficient being too large. It is easy to see: when $\lambda = 0$, we recover ordinary linear regression; when $\lambda \to \infty$, we get $\hat{\beta} \to 0$.

It is worth noting that we need to center the data X and Y first before performing ridge regression, such that $\hat{\beta}_0$ will not be affected. As we will see in the following, ridge regression has the effect of shrinking coefficients. Therefore, we need to preprocess that data: Each column of X data should be centered and divided by standard deviation, such that each column has zero sample mean and unit sample variance. For canonical linear regression, we do not need to preprocess the data in prediction(the $X\hat{\beta} = X(X^TX)^{-1}X^TY$ from preprocessed data and original data make no difference.

The following theorem summarize solution and properties in ridge regression.

Theorem 23.2.1 (solution and properties in ridge regression). [a]

- *The solution to the ridge regression is*

$$\hat{\beta}_{ridge} = (X^TX + \lambda I)^{-1}X^TY.$$

- *The ridge estimator for β is biased*

$$E[\hat{\beta}_{ridge}] = (X^TX + \lambda I)^{-1}X^TX\beta \neq \beta\beta_{OLS},$$

whereas in the ordinary linear regression $\hat{\beta}_{OLS} = \beta$. Particularly if we denote the eigen-decomposition $X^TX = U\Sigma U^T$, then

$$E[\hat{\beta}_{ridge}] = U(\Sigma + \lambda I)^{-1}\lambda I\beta.$$

- *The covariance of the and*

$$Cov[\hat{\beta}_{ridge}] = \sigma^2(X^TX + \lambda I)^{-1}X^TX(X^TX + \lambda I)^{-1}$$

- *Further, let $X^TX = U^TD^2U, X = DU$(via SVD, Theorem 4.9.1], where $U = [u_1, ..., u_p]$ is orthogonal basis of the subspace spanned by X, $D = diag(d_1, d_2, ..., d_p)$, then the fitted response is given as*

$$\hat{y} = X\hat{\beta}_{ridge} = \sum_{j=1}^{p} u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^T y.$$

(Note that $u_j^T y$ is the projection on the basis vector u_j.)

a code: ridgeRegression.py

Proof. (1)(2) Direct optimization.
(3)

$$\begin{aligned}
\hat{y} &= X\hat{\beta}_{ridge} \\
&= X(X^TX + \lambda I)^{-1}X^Ty \\
&= UD(D^2 + \lambda I)^{-1}DU^Ty \\
&= \sum_{j=1}^{p} u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^T y
\end{aligned}$$

□

Remark 23.2.1 (choice of λ: bias and variance reduction).

- If $\lambda > 0$, the estimation of $\hat{\beta}$ from ridge regression is **lower biased**; and the variance of $\hat{\beta}$ is reduced(smaller than that of canonical linear regression.)
- In high-dimensional regression, the canonical linear regression ($\lambda = 0$) will usually end up with a better fit to the training data, but will perform worse fit to the new data.
- The λ will chosen in the following way:
  - A more traditio., AIC or BIC, is the smallest.

– A more mnal approach would be to choose λ such that some information criterion, e.gachine learning-flavor approach is to perform cross-validation and select the value of λ with minimal cross-validation errors.

Remark 23.2.2 (necessary for data preprocessing).

- The data preprocessing procedure: Each column of X data should be centered and divided by standard deviation, such that each column has zero sample mean and unit sample variance.
- For canonical linear regression, we do not need to preprocess the data in prediction(the $X\hat{\beta} = X(X^TX)^{-1}X^TY$ from preprocessed data and original data make no difference. Note that scaling amounts to times a diagonal matrix)
- For ridge regression, we need to preprocess that data, because the shrinkage effect depends on the scale of the predicators.

Remark 23.2.3 (effect of shrinkage).

- Note that in canonical linear regression, the projection coordinate of y is $u_j^T y$ (i.e.$\hat{y} = \sum_{j=1}^p u_j u_j^T y$), whereas in the ridge regression, the projection coordinate is $\frac{d_j^2}{d_j^2+\lambda} u_j^T y$.
- The shrinkage effect is such that the basis with **smaller variance(i.e. smaller d_j^2) will shrink more.**
- If X^TX is diagonal, i.e., $X^TX = diag(n_1, ..., n_p)$, then

$$\hat{\beta}_{ridge,i} = \frac{n_i}{n_i + \lambda}\hat{\beta}_i.$$

Remark 23.2.4 (interpretation from Bayesian point of view). Assume $\epsilon \sim MN(0, \sigma^2 I)$. To perform an MAP estimation from Bayesian statistics, we would minimize the log function

$$\min_\beta -\log P(\beta|X,Y) = \min_\beta -\log P(Y|X,\beta) - \log P(\beta).$$

Specifically, if β is assumed to have prior distribution of $MN(0, \sigma^2/\lambda)$, we have (after removing terms irrelevant to β)

$$\begin{aligned}
&\min_\beta \left[\sum_{i=1}^n \frac{\left(y_i - x_i^T\beta\right)^2}{2\sigma^2} + \lambda \sum_{j=1}^p \frac{\beta_j^2}{2\sigma^2} \right] \\
=&\min_\beta \left[\sum_{i=1}^n \left(y_i - x_i^T\beta\right)^2 + \lambda \sum_{j=1}^p \beta_j^2 \right]
\end{aligned}$$

It is clear that optimization problem is the same the least square problem of the ridge regression.

23.2.1.2 *Dual form of ridge regression*

Let $x \in \mathbb{R}^D$ be some feature vector, and X be the $N \times D$ design matrix. The goal is

$$\min_w (y - Xw)^T (y - Xw) + \lambda \|w\|_,^2 \lambda > 0$$

with the optimal solution given as [Theorem 23.2.1]

$$w = (X^T X + \lambda I)^{-1} X^T y = (\sum_i x_i x_i^T + \lambda I)^{-1} X^T y.$$

If we use the matrix inversion lemma [Lemma A.8.3] to invert $(X^T X + \lambda I)^{-1}$, we have the following results.

Lemma 23.2.1 (the dual form for prediction). *[1, p. 494] Using matrix inversion lemma, $w = X^T (XX^T + \lambda I_N)^{-1} y$. Define $\alpha = (XX^T + \lambda I_N)^{-1} y$, then $w = X^T \alpha$.*

Moreover, given a new input x, the prediction is

$$y = w^T x = \alpha^T X^T x = \sum_i \alpha_i x_i^T x.$$

Proof. Using the matrix inversion lemma [Lemma A.8.3] can prove it. □

There are several advantages of dual form:

- The dual form has advantage when $D \gg N$ in calculating $(XX^T + \lambda I_N)^{-1}$, which takes $O(N^3)$ whereas in origin form $(X^T X + \lambda I)^{-1}$ takes $O(D^3)$.
- The dual form uses inner product XX^T and $x_i^T x$ in solving w and calculating the prediction, which can be combined with kernel methods [section 22.6] to extend feature space to an nonlinear one.

23.2.2 Lasso regression

Although Ridge regression addresses the rank-deficiency issue and shrink model parameters, its shrinkage effect is not dramatic and many parameters remain non-zero. In high-dimensional regression, shrink less important parameter to zero to help model interpretation at the expense of model prediction error sometimes is more desirable.

Lasso regression is a penalized linear regression can perform more aggressive parameter shrinkage, more precisely, shrinking to exact zeros.

Definition 23.2.1 (The Lasso regression). *Consider **centered** data in the linear regression model. The Lasso regression is to estimate $\hat{\beta}_{Lasso}$ by minimizing*

$$\min_{\beta} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1 .$$

Lasso regression does not adopt analytical solution. In the following, we develop an iterative algorithm for Lasso regression based on gradient descent.

Let the training data set be $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(n)}, y^{(n)})\}$. Let $J^{OLS} = \|Y - X\beta\|_2^2$, and we can write the derivative with respect to β_j as

$$\begin{aligned}
\frac{\partial}{\partial \beta_j} J^{OLS} &= -\sum_{i=1}^{m} x_j^{(i)} \left[y^{(i)} - \sum_{j=0}^{n} \beta_j x_j^{(i)} \right] \\
&= -\sum_{i=1}^{m} x_j^{(i)} \left[y^{(i)} - \sum_{k \neq j}^{n} \beta_k x_k^{(i)} - \beta_j x_j^{(i)} \right] \\
&= -\sum_{i=1}^{m} x_j^{(i)} \left[y^{(i)} - \sum_{k \neq j}^{n} \beta_k x_k^{(i)} \right] + \beta_j \sum_{i=1}^{m} \left(x_j^{(i)} \right)^2 \\
&\triangleq -\rho_j + \beta_j z_j
\end{aligned}$$

where we denote $\rho_j = \sum_{i=1}^{m} x_j^{(i)} \left[y^{(i)} - \sum_{k \neq j}^{n} \beta_k x_k^{(i)} \right]$ and $z_j = \sum_{i=1}^{m} \left(x_j^{(i)} \right)^2$.

Because the objective function is convex function, the necessary and sufficient condition is that zero is contained in the subdifferential of J [Theorem 9.5.2]. Note that the subdifferential of L1 penalty is given by

$$\partial_{\theta_j} \lambda \sum_{j=0}^{n} \left| \beta_j \right| = \partial_{\beta_j} \lambda \left| \beta_j \right| = \begin{cases} \{-\lambda\} & \text{if } \beta_j < 0 \\ [-\lambda, \lambda] & \text{if } \beta_j = 0 \\ \{\lambda\} & \text{if } \beta_j > 0 \end{cases}$$

Our goal is to determine β_j such that $0 \in \partial_{\beta_j}$. After some algebra, we have

$$\begin{cases} \beta_j = \frac{\rho_j + \lambda}{z_j} & \text{if } \rho_j < -\lambda \\ \beta_j = 0 & \text{if } -\lambda \leq \rho_j \leq \lambda \\ \beta_j = \frac{\rho_j - \lambda}{z_j} & \text{if } \rho_j > \lambda \end{cases}$$

We can more compactly write $\beta_j = \frac{1}{z_j} S(\rho_j, \lambda)$, where $S(x, \lambda)$ is the soft thresholding operator defined by

$$S(z, \lambda) = \text{sgn}(z)(|z| - \lambda)_+ = \begin{cases} z - \lambda, & \text{if } z > \lambda \\ 0 & \text{if } |z| \leq \lambda \\ z + \lambda, & \text{if } z < -\lambda \end{cases}$$

The gradient descent algorithm can be summarized in the following. Similar to Rigid regression, each column of X data should be centered and divided by standard deviation as the data preprocessing step.

Algorithm 32: Coordinate descent for Lasso regression.

Input: Training data set $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(n)}, y^{(n)})\}$.

repeat

  for $j = 1, ..., p$ **do**

    Compute

$$\rho_j = \sum_{i=1}^{m} x_j^{(i)} \left[y^{(i)} - \sum_{k \neq j}^{n} \beta_k x_k^{(i)} \right]$$

    Compute

$$z_j = \sum_{i=1}^{m} \left(x_j^{(i)} \right)^2.$$

    Set

$$\beta_j = \frac{1}{z_j} S(\rho_j, \lambda).$$

  end

until *stopping criterion is met*;

Output: coefficients $\beta_1, ..., \beta_p$

Remark 23.2.5 (interpretation from Bayesian point of view). Assume $\epsilon \sim MN(0, \sigma^2 I)$. To perform an MAP estimation from Bayesian statistics, we would minimize the log function

$$\min_{\beta} -\log P(\beta | X, Y) = \min_{\beta} -\log P(Y | X, \beta) - \log P(\beta).$$

Specifically, if each component of β is assumed to have prior Laplace distribution of $Laplace(0, \sigma^2/\lambda) = \frac{\lambda}{2\sigma^2} e^{-\frac{\lambda|x|}{\sigma^2}}$ [Definition 12.1.7] , we have (after removing terms irrelevant to β)

$$\begin{aligned} &\min_{\beta} \left[\sum_{i=1}^{n} \frac{\left(y_i - x_i^T \beta\right)^2}{2\sigma^2} + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{2\sigma^2} \right] \\ =&\min_{\beta} \left[\sum_{i=1}^{n} \left(y_i - x_i^T \beta\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right] \end{aligned}$$

It is clear that optimization problem is the same the least square problem of the Lasso regression.

Prior Laplace distribution [Definition 12.1.7] has a sharp peak at the mean (here mean is zero), thus promoting sparsity.

23.2.3 Elastic net

The Elastic net regression has the mixed favor of Lasso regression L1 penalty and ridge regression L2 penalty.

Definition 23.2.2 (The Elastic net regression). *Consider **centered** data in the linear regression model. The ridge regression is to estimate $\hat{\beta}_{ridge}$ by minimizing*

$$\min_{\beta} \left\| Y - X\beta \right\|_2^2 + \lambda \sum_{i=1}^{p} (\alpha \beta_j^2 + (1 - \alpha \left| \beta_j \right|)),$$

where α is the mixing parameter between ridge ($\alpha = 0$) and Lasso ($\alpha = 1$).

The optimization of the Elastic net follows similar gradient descent algorithm like the Lasso [algorithm 32].

23.2.4 Shrinkage Comparison

Note that optimization of the form

$$\min_{\beta} \left\| Y - X\beta \right\|_2^2 + \lambda \left\| \beta \right\|_p,$$

is also equivalent to

$$\min_{\beta} \|Y - X\beta\|_2^2, s.t., \|\beta\|_p \leq t(\lambda).$$

where $t(\lambda)$ is a real-valued number depending on λ (derived from nonlinear constrained optimization optimality condition.) The shrinkage effects for different p norm optimization can be visualized in Figure 23.2.1.

All types of penalty will shrink down estimated coefficients, but in different manner:

- Lasso regression tend to set some coefficients to zero (particularly those features less correlated to outcome), thus performing feature selection.
- Rigid regression scales minimizers to smaller values but not zeros.
- The Elastic net scales estimates down and set coefficients to zeros in a less aggressive way compared to Lasso.

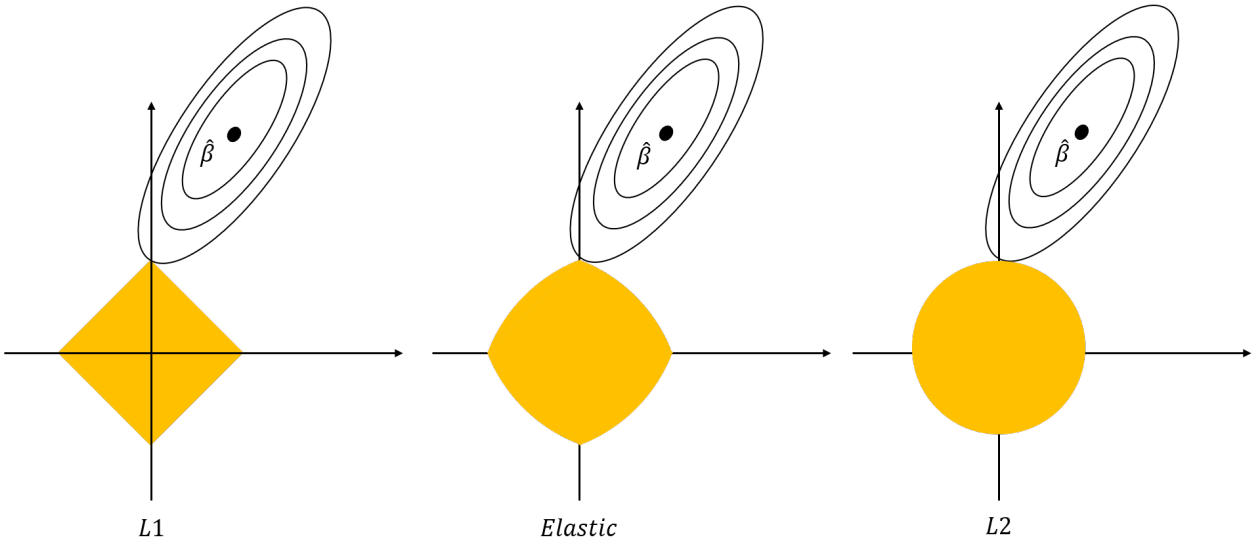
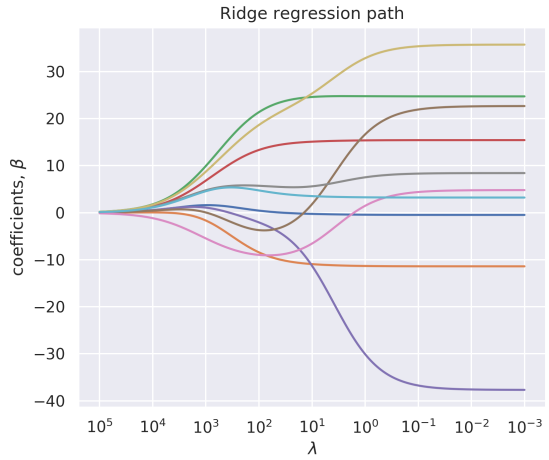


Figure 23.2.1: Comparison different penalization: Lasso $L1$, elastic net, and Ridge $L2$. Orange regions are admissible set for parameter β. Contours are objective function value as a function of parameter β. Black solid circles are the minimizers $\hat{\beta}$ when there are no penalties, and red solid circles are the minimizers when penalties are applied.

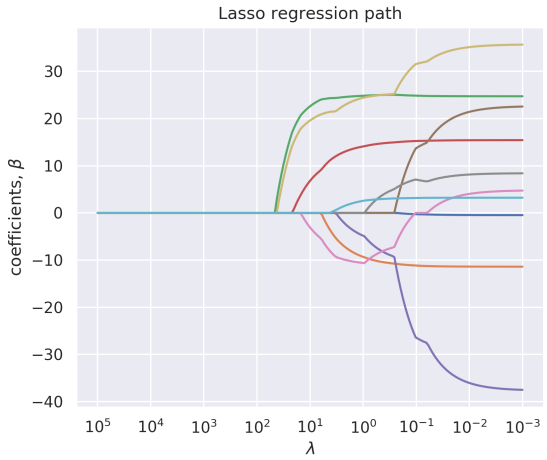
The difference of shrinkage effect can be further demonstrated in the regularization paths in a linear regression problem, as we showed in Figure 23.2.2. Clearly, L_1 penalty has the strongest effect on inducing sparsity, or setting coefficients to zeros; L_2 penalty, on the other hand, allows many non-zero coefficients even when employing large penalty terms; and elastic net falls between the middle.

In practice, some guideline on choices of different penalties are:

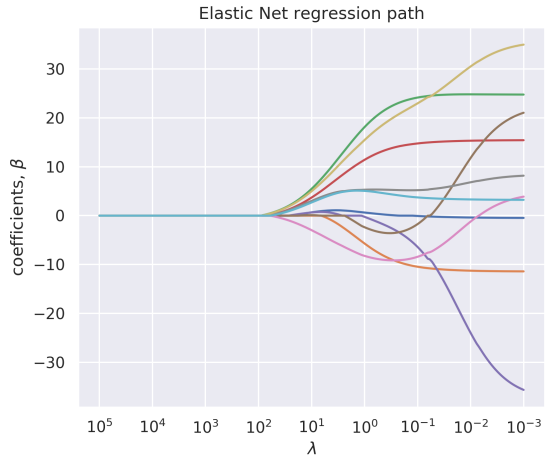
- Lasso tends to do well if there are a small number of significant parameters and the others are close to zero (ergo: when only a few predictors actually influence the response).
- Ridge works well if there are many large parameters of about the same value (ergo: when most predictors impact the response).
- Elastic Net lies on the middle ground and Lasso and Ridge.
- Since Elastic net includes the penalty of Lasso and Ridge, it is recommended using Cross Validation to determine the most suitable penalty.



(a) L2 regularization path.



(b) L1 regularization path.



(c) Elastic net (ratio = 0.5) regularization path.

Figure 23.2.2: Penalized regression path in a toy regression problem.

23.2.5 Effective degree of freedom

In our penalized linear regression, we also want capture the shrinkage effect via the effective number of free parameters, or effective degree of freedoms.

One measurement, introduced in [2], says:

Definition 23.2.3. *Given observation $y = (y_1, ..., y_n), y_i \in \mathbb{R}$ and model prediction $\hat{y} = (\hat{y}_1, ..., \hat{y}_n), \hat{y}_i \in \mathbb{R}$. The effective degree of freedom is defined by*

$$df(\hat{y}) = \frac{1}{\sigma^2} \sum_{i=1}^{n} Cov(y_i, \hat{y}_i).$$

Clearly, the higher the correlation between the ith fitted value and the ith data point, the more adaptive the estimate, and so the higher its degrees of freedom. Using established results in linear regression [Lemma 15.1.1] and ridge regression [Theorem 23.2.1], we have

Methodology 23.2.1 (effective degree of freedom estimation). *Let $X \in \mathbb{R}^{n \times p}$ be the data matrix for a linear regression problem. We have*

- *For linear regression, $\hat{y} = X\hat{\beta} = Hy$, where $H = X(X^TX)^{-1}X^T$,*

$$df(\hat{y}) = Tr(H).$$

- *For ridge regression, $\hat{y} = X\hat{\beta}^{ridge} = H^{ridge}y$, where $H^{ridge} = X(X^TX + \lambda I)^{-1}X^T$,*

$$df(\hat{y}) = Tr(H^{ridge}).$$

- *For lasso regression, $\hat{y} = X\hat{\beta}^{lasso}$,*

$$df(\hat{y}) = E[number\ of\ nonzero\ coefficients\ in \hat{\beta}^{lasso}].$$

23.3 Basis function extension

In this section, we extend our linear model to the nonlinear territory. The core idea is to augment the original feature vector X with additional features, which are usually the relatively simple nonlinear transformation X. Denoting these features by $h_m(X)$, where we call transformation functions h_m as basis function, the linear model framework based on these basis functions is then given by

$$y = \sum_{m=1}^{p} \beta_m h_m(X) + \epsilon.$$

Common basis functions or nonlinear transformations include

- $h(X) = X$, identify transformation that recovers linear regression.
- $h(X_1, X_2) = (X_1, X_2, X_1^2, X_2^2, 2X_1X_2)$, polynomial transformations. Here we use polynomial of degree 2.
- $h(X) = \log(X)$, log transformation.
- $h(X) = X^p$, power transformation.

The multiple linear regression with n samples using basis function features can be written as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & h_{11} & h_{12} & \dots & h_{1(p-1)} \\ 1 & h_{21} & h_{22} & \dots & h_{2(p-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & h_{n1} & h_{n2} & \dots & h_{n(p-1)} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{p-1} \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

with matrix form

$$Y = H\beta + \epsilon.$$

The task of finding coefficients β and σ can be formulated as a least square minimization problem [Theorem 15.1.1] given by

$$\min \left\| Y - H\beta \right\|^2$$

where the coefficient vector is $\beta = (\beta_0, \beta_1, ..., \beta_{p-1}]^T$, the minimizer β is given by

$$\hat{\beta} = (H^T H)^{-1} H^T Y.$$

Consider the linear regression example in Figure 23.3.1. The data sample are generated from nonlinear mapping given by $y = 2x_1 + x_2 - 0.8x_1x_2 + 0.5x_1^2 + \epsilon$ and will

yield poor regression results if we use model $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon$. Under the basis function regression framework, we can regress y on terms $x_1, x_2, x_1^2, x_1 x_2, x_2^2$ to improve the fitting results.

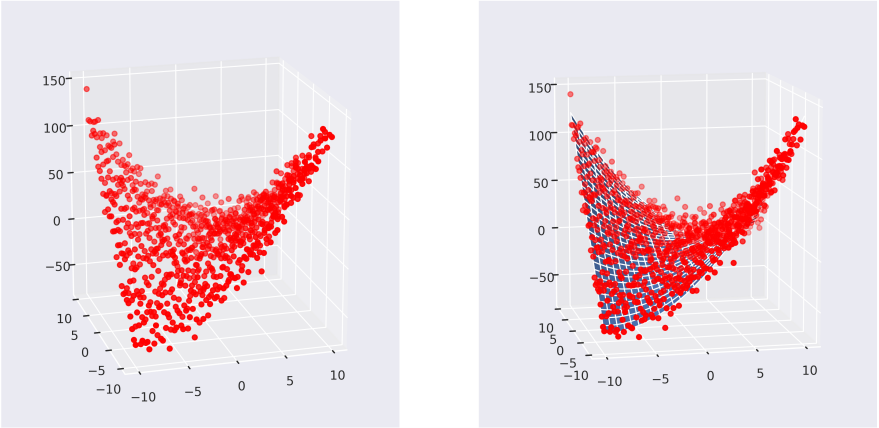


Figure 23.3.1: Linear regression with non-linear high order terms. The samples are generated via $y = 2x_1 + x_2 - 0.8x_1 x_2 + 0.5x_1^2 + \epsilon$, where ϵ is noise.

Ultimately, the basis function framework offer a systematic way to enhance linear regression models. Figure 23.3.2 illustrates a common workflow, where we start with the vanilla linear model with linear terms and then incrementally add cross terms and univariate high order terms.

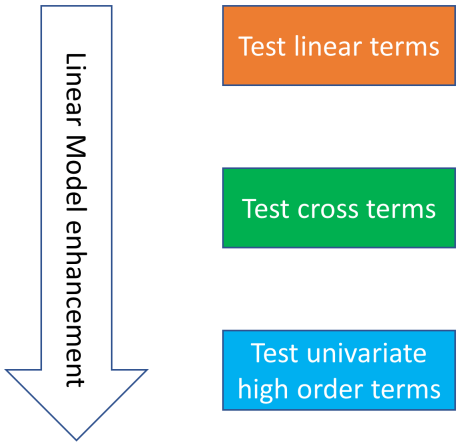


Figure 23.3.2: Linear model enhancement flowchart.

23.4 Note on bibliography

For linear regression models, see [3][4]. For, linear models with *R* resources, see [5]. For models with sparsity, see [6]

BIBLIOGRAPHY

1. Murphy, K. P. *Machine learning: a probabilistic perspective* (MIT press, 2012).
2. Friedman, J., Hastie, T. & Tibshirani, R. *The elements of statistical learning (2017 corrected version)* (Springer series in statistics Springer, Berlin, 2007).
3. Kutner, M., Nachtsheim, C. & Neter, J. *Applied Linear Regression Models* ISBN: 9780072955675 (McGraw-Hill Higher Education, 2003).
4. Seber, G. A. & Lee, A. J. *Linear regression analysis* (John Wiley & Sons, 2012).
5. Faraway, J. J. *Linear models with R* (CRC press, 2014).
6. Hastie, T., Tibshirani, R. & Wainwright, M. *Statistical learning with sparsity: the lasso and generalizations* (Chapman and Hall/CRC, 2015).