Linear model for regression

Peida Wu

June 5, 2025

Contents

1 Introduction

Hastie et al. [2009] A linear regression model assumes that the regression function $E(Y|X)$ is linear in the input $X_1, \ldots, X_p$. Linear models were largely developed in the precomputer age of statistics, but even in todays computer era there are still good reasons to use them. They are simple and often provide an adequate and interpretable description of how the inputs affect the output. For prediction purposes they can sometimes outperform fancier nonlinear models, especially in situations with small numbers of training cases, low signal-to-noise ratio or sparse data. Finally, linear methods can be applied to transformations of the inputs and this considerably expands their scope.

2 Linear Regression Models and Least Squares

2.1 Linear models

Suppose we have input vector $X^T = (X_1, X_2, \ldots, X_p)$, and we want to predict a real-valued output Y. The linear regression model has the form

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j \tag{1}$$

The linear model either consider the regression function $E(Y|X)$ is linear, or that the linear model is a reasonable approximation. Here the β_j are unknown parameters of coefficients, and the variables X_j can come from different sources:

1. quantitive inputs.
2. transformations of quantitive inputs, such as log, square-root, etc.
3. basis expansion, such as $X_2 = X_1^2, X_3 = X_1^3$, leading to a polynomial representation.
4. interactions between variables, for example, $X_3 = X_1 \cdot X_2$.

No matter the source of X_j, the model is linear in the parameters.

2.2 Least squares

2.2.1 Solution for least squares

Typically we have a set of training data $(x_1, y_1), \ldots, (x_N, y_N)$ from which to estimate the parameters β, Each $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$ is a is a vector of feature measurements for the ith case. The most popular estimation method is least squares, in which we pick the coefficients $\beta = (\beta_0, \ldots, \beta_p)$ to minimize the residue sum of squares

$$\begin{aligned} RSS(\beta) &= \sum_{i=1}^{N} (y_i - f(x_i))^2 \\ &= \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \end{aligned} \tag{2}$$

From statistical point of view, this criterion is reasonable if the training observations (x_i, y_i) represent indepent random draws from the population. Note that the equation (2) makes no assumptions about the validity of the model (1), it simply finds the best linear fit to the data. Least squares fitting is intuitively satisfying no matter how the data arise; the criterion measures the average lack of fit.

How to minimize equation (2)? Denote by $\mathbf{X}$ the $N \times (p+1)$ matrix with each row an input vector, and similarly let $\mathbf{y}$ be the N-vector of outputs in the training set. Then we can write the residue sum of squares as

$$RSS(\beta) = (y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta) \tag{3}$$

This is a quadratic function in the $p+1$ parameters. Differentiating with respect to β we obtain:

$$\begin{cases} \frac{\partial RSS}{\partial \beta} = & -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) \\ \frac{\partial^2 RSS}{\partial \beta \partial \beta^T} = & 2X^T X \end{cases} \tag{4}$$

Assuming that $\mathbf{X}$ has full column rank, and hence $\mathbf{X}^T\mathbf{X}$ is positive definite, we set the first derivative to zero

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0 \tag{5}$$

to obtain the unique solution

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{6}$$

Thus $\hat{y}$ is

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{7}$$

The matrix $\mathbf{H} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ appear in (7) is sometimes called the “hat” matrix because it puts the hat on $\mathbf{y}$.

`Images/projection.png`

Figure 1: Projection

The pecture shows the N-dimensional geometry of least squares regression with two predictors. The outcome vector y is orthogonally projected onto the hyperplane spanned by the input vectors x_1 and x_2. The projection $\hat{y}$ represents the vector of the least squares predictions.

2.2.2 Inference of coefficients

Up to now we have made minimal assumptions about the true distribution of the data. In order to pin down the sampling properties of $\hat{\beta}$, we now assume that the observations y_i are uncorrelated and have constant variance σ^2, and that the x_i are fixed. The variance- covariancew matrix of the least squares parameter estimates can be derived from (6) and is given by

$$\mathrm{Var}(\hat{\beta}) = (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2 \tag{8}$$

Typically one estimates the variance σ^2 by

$$\hat{\sigma}^2 = \frac{1}{N-p-1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{9}$$

The $N-p-1$ rather than N makes $\hat{\sigma}^2$ an unbiased estimate of σ^2.

To draw inferences about the parameters and the model, additional assumptions are needed. We now assume that (1) is the correct model for the mean; that is, the conditional expectation of Y is linear in $X_1, \dots, X_p$. We also assume that the deviations of Y around its expectation are additive and Gaussian. Hence

$$\begin{aligned} Y =&E(Y|X_1, \dots, X_p) + \epsilon \\ =&\beta_0 + \sum_{j=1}^{p} X_j\beta_j + \epsilon \end{aligned} \tag{10}$$

where $\epsilon \sim N(0, \sigma^2)$. Thus

$$\hat{\beta} \sim N(\beta, (X^TX)^{-1}\sigma^2), (N-p-1)\hat{\sigma}^2 \sim \sigma^2\chi^2_{N-p-1} \tag{11}$$

Since $\hat{\beta}$ and $\hat{\sigma}^2$ are independent, we can form tests of hypothesis and confidence intervals for parameters β_j.

To test the hypothesis that a particular coefficient $\beta_j = 0$, we form the standardized coefficient of Z-score

$$z_j = \frac{\hat{\beta}_j}{\hat{\sigma}\sqrt{v_j}} \tag{12}$$

where v_j is the diagonal element of $(\mathbf{X}^T\mathbf{X})^{-1}$. Under the null hypothesis that $\beta_j = 0$, $z_j \sim t_{N-p-1}$. If $\hat{\sigma}$ is replaced by a known value σ, then z_j would have a standard normal distribution. The difference between the tail quantiles of a t-distribution and a standard normal become negligible as the sample size increases, and so we typically use the normal quantiles.

Sometimes we need to test for the significance of groups of coefficients simultaneously. For example, to test if a categorical variables with k levels can be excluded from a model, we need to test whether the coefficients of the dummy variables used to represent the lvels can be all be set to 0. Here we use F statistic:

$$F = \frac{(RSS_0 - RSS_1)/(p_1 - p_0)}{(RSS_1)/(N - p_1 - 1)} \tag{13}$$

where RSS_1 is the residual sum-of squares for the least squares fit of the bigger model with $p_1 + 1$ parameters, and RSS_0 the same for the second smaller model with $p_0 + 1$ parameters.

2.3 The Gauss-Markov Theorem

One of the most famous results in statistics asserts that the least squares estimates of the parameters β have the smallest variance among all linear unbiased estimates.

Gauss Markov theorem: In linear regression model, if the error satisfies mean zero, variance are identitical and unrelated, then the regression coefficients β is best linear unbiased estimator(i.e. if $\theta*$ is any unbiased estimator of θ, then $\text{Var}(\theta*) \geq \text{Var}(\theta)$).

proof: First we prove unbiasedness:

$$\begin{aligned} E(\hat{\beta}) =&(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T E(\mathbf{y}) \\ =&(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T E(\mathbf{X}\beta + \epsilon) \\ =&(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}(\beta + E(\epsilon)) \\ =&\beta \end{aligned} \tag{14}$$

Then compute $\text{Var}(\hat{\beta})$, Since $\beta = \mathbf{X}^T E(\mathbf{y})$,

$$\begin{aligned} \text{Var}(\hat{\beta}) &= E[(\hat{\beta} - \beta)(\hat{\beta} - \beta)^T] \\ &= E[((X^TX)^{-1}X^T\epsilon)((X^TX)^{-1}X^T\epsilon)^T] \\ &= E[(X^TX)^{-1}X^T\epsilon\epsilon^T X(X^TX)^{-1}] \\ &= (X^TX)^{-1}X^T E[\epsilon\epsilon^T]X(X^TX)^{-1} \end{aligned} \tag{15}$$

Since $E(\epsilon) = 0$, we get

$$\text{Var}(\epsilon) = E[(\epsilon - E(\epsilon))(\epsilon - E(\epsilon)^T)] = E(\epsilon\epsilon^T) = \sigma^2 I_n \tag{16}$$

Thus

$$\begin{aligned}\text{Var}(\hat{\beta}) =&(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T E(\epsilon\epsilon^T)\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\\ =&(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\sigma^2 I_n)\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\\ =&\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}\end{aligned} \tag{17}$$

Finally we prove the least squares is the best unbiased linear estimate by contradiction. Suppose there is a better unbiased linear estimate $\tilde{\beta} = \mathbf{CY}$. Since C is arbitary, denote $\mathbf{C} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + D$, where D is a $(p+1) \times N$ matrix.
Since $\tilde{\beta}$ is unbiased. we must have $E[\tilde{\beta}] = \beta$.

$$\begin{aligned}E[\tilde{\beta}] &= E[\mathbf{CY}]\\ &= \mathrm{E}\left[\left((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}\right)(\mathbf{X}\beta + \varepsilon)\right]\\ &= \left((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}\right)\mathbf{X}\beta + \left((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}\right)\mathrm{E}[\varepsilon]\\ &= \left((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}\right)\mathbf{X}\beta\\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}\beta + \mathbf{DX}\beta\\ &= (\mathbf{I}_{p+1} + \mathbf{DX})\beta\end{aligned} \tag{18}$$

Thus $\tilde{\beta}$ is unbiased iff $D\mathbf{X} = 0$,

$$\begin{aligned}\text{Var}(\hat{\beta}) &= \text{Var}(\mathbf{CY})\\ &= \mathbf{C}\text{Var}(\mathbf{Y})\mathbf{C}^T\\ &= \sigma^2\mathbf{CC}^T\\ &= \sigma^2\left((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T + \mathbf{D}\right)\left(\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + \mathbf{D}^T\right)\\ &= \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1} + \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{D}^T + \sigma^2\mathbf{DX}(\mathbf{X}^T\mathbf{X})^{-1} + \sigma^2\mathbf{DD}^T\\ &= \sigma^2(\mathbf{X}^T\mathbf{X})^{-1} + \sigma^2\mathbf{DD}^T\\ &= \text{Var}(\hat{\beta}_{LSE}) + \sigma^2\mathbf{DD}^T\end{aligned} \tag{19}$$

For any matrix D, DD^T is positive difinite, which means least square estimate is the best unbiased estimator.

3 Subset selection

There are 2 reasons we don't often satisfied with the least square.

1. The first is prediction accuracy: the least squares estimates often have low bias but large variance. Prediction accuracy can sometimes be improved by shrinking or setting some coefficients to zero.

2. 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.

With subset selection we retain only a subset of the variables, and eliminate the rest from the model. Least squares regression is used to estimate the coefficients of the inputs that are retained. There are a number of different strategies for choosing the subset.

3.1 Best subset selection

To perform best subset selection, we fit a separate least squares regression for each possible combination of p predictors. We then look at all of the resulting models, with the goal of identifying the one that is best.

The problem of selecting the best model from among the 2^p possibilities considered by best subset selection is not trivial. This is usually broken up into two stages, as described in Algorithm 1.

Algorithm 1 Best Subset Selection

```
1: Let $\mathcal{M}_0$ denote the null model with no predictors (predicts sample mean for each observation)
2: for $k = 1$ to $p$ do
3:     Fit all $\binom{p}{k}$ models containing exactly $k$ predictors
4:     Select the best model among them and denote it as $\mathcal{M}_k$
       (chosen by smallest RSS or largest $R^2$)
5: Select the final best model from $\mathcal{M}_0, \dots, \mathcal{M}_p$
       (using cross-validated prediction error, $C_p$, BIC, or adjusted $R^2$)
```

In algorithm 1, step2 identifies the best model for each subset size, in order to reduce 2^p models to $p + 1$ possible models.

This task must be performed with care, because the RSS of these $p + 1$ models decreases monotonically, as the number of features included in the models increases. Therefore, if we use these statistics to select the best model, then we will always end up with a model involving all of the variables. The problem is that a low RSS indicates a model with a low training error, whereas we wish to choose a model that has a low test error. Therefore, in Step 3, we use cross-validated prediction error, C_p, BIC, or adjusted R^2 in order to select among $M_0, M_1, \dots, M_p$.

While best subset selection is a simple and conceptually appealing approach, it suffers from computational limitations. The number of possible models that must be considered grows rapidly as p increases.

3.2 Forward subset selection

Forward stepwise selection is a computationally efficient alternative to best subset selection. While the best subset selection procedure considers all 2^p possible models containing subsets of the p predictors, forward stepwise considers a much smaller set of models. Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model. In particular, at each step the variable that gives the greatest additional improvement to the fit is added to the model.

Algorithm 2 Forward Stepwise Selection

```
1: Let $\mathcal{M}_0$ denote the null model (with no predictors)
2: for $k = 0$ to $p - 1$ do
3:     Consider all $p - k$ models that add one predictor to $\mathcal{M}_k$
4:     Choose the best among these models and call it $\mathcal{M}_{k+1}$
       (chosen by smallest RSS or largest $R^2$)
5: Select a single best model from $\mathcal{M}_0, \dots, \mathcal{M}_p$
       (using cross-validated prediction error, $C_p$, BIC, or adjusted $R^2$)
```

Compared with best subset selection, forward subset selection only contains $1 + \frac{p(p+1)}{2}$ models.

Forward stepwise selection's computational advantage over best subset selection is clear. Though forward stepwise tends to do well in practice, it is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors.

3.3 Backward subset selection

Like forward stepwise selection, backward stepwise selection provides an efficient alternative to best subset selection. However, unlike forward stepwise selection, it begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time.

Algorithm 3 Backward Stepwise Selection

```
1: Let $\mathcal{M}_p$ denote the full model containing all $p$ predictors
2: for $k = p$ down to 1 do
3:     Consider all $k$ models that remove one predictor from $\mathcal{M}_k$
4:     Choose the best among these models and call it $\mathcal{M}_{k-1}$
       (chosen by smallest RSS or largest $R^2$)
5: Select a single best model from $\mathcal{M}_0, \dots, \mathcal{M}_p$
       (using cross-validated prediction error, $C_p$, BIC, or adjusted $R^2$)
```

Like stepwise forward selection, the backward selection approach searches through only $1 + \frac{p(p+1)}{2}$ models, and so can be applied in settings where p is too large to apply the best subset selection. Also like forward stepwise selection, backward stepwise selection is not guaranteed to yield the best model containing a subset of the p predictors.

Backward selection requires that the number of samples n is larger than the number of variables p (so that the full model can be fit).

3.4 Comparison among subset selection

In this section, we will use Hitter dataset to compare the three methods. In this experiment, we will use other factors to predict the linear model for salary.

Images/hitter dataset.png

Figure 2: Hitter dataset

We simulate the first 8 step by using best subset selection, The result is shown below Then we use different criteria to show how much input factor can best fit the model.

Images/subset selection.png

Figure 3: First 8 factors

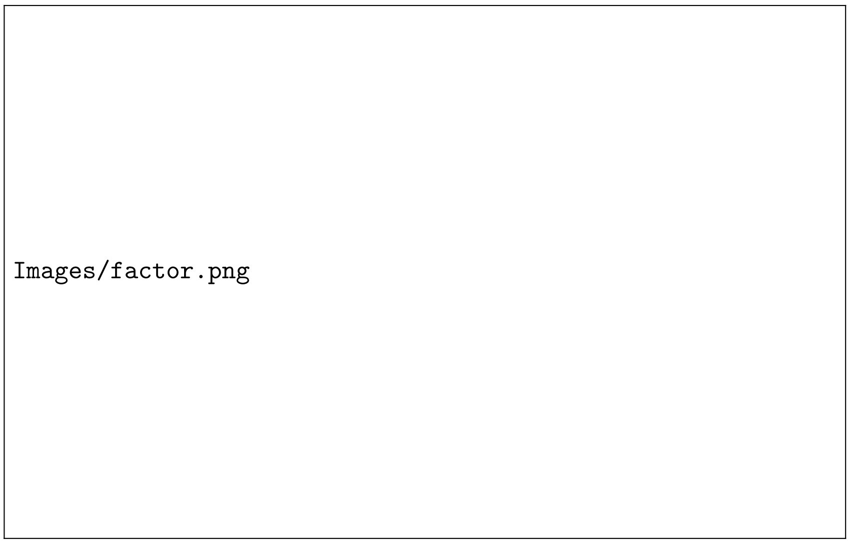

Figure 4: different criteria

Then compared with forward subset selection and backward subset selection. The result is not always the same, that is, sometimes best subset selection will delete some expected factors before. When the factor chosen is 7, we notice that:

	Full Model		**Forward Selection**		**Backward Selection**	
	Variable	Coefficient	Variable	Coefficient	Variable	Coefficient
1	(Intercept)	79.451	(Intercept)	109.787	(Intercept)	105.649
2	Hits	1.283	AtBat	-1.959	AtBat	-1.976
3	CHmRun	1.442	CWalks	-0.305	CWalks	-0.716
4	DivisionW	-129.987	DivisionW	-127.122	DivisionW	-116.169
5	Walks	3.227	Hits	7.450	Hits	6.757
6	CAtBat	-0.375	Walks	4.913	Walks	6.056
7	CHits	1.496	CRBI	0.854	CRuns	1.129
8	PutOuts	0.237	PutOuts	0.253	PutOuts	0.303

Table 1: Coefficients Comparison from Three Selection Methods (7 Variables Each)

4 Shinkage methods

By retaining a subset of the predictors and discarding the rest, subset selection produces a model that is interpretable and has possibly lower prediction error than the full model. However, because it is a discrete process— variables are either retained or discarded—it often exhibits high variance, and so doesn't reduce the prediction error of the full model. Shrinkage methods are more continuous, and don't suffer as much from high variability.

4.1 Ridge regression

Ridge regression shrinks the regression coefficients by imposing a penalty on their size. The ridge coefficients minimize a penalized esidual sum of squares,

$$\hat{\beta}^{ridge} = \text{argmin}_{\beta}\{\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{i=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{i=1}^{p} \beta_j^2\} \tag{20}$$

Here $\lambda \geq 0$ is a complexity parameter that controls the amount of shankage: the larger the value of λ, the greater the amount of shinkage. The coefficients are shrunk toward zero (and each other).

An equivalent way to write the ridge problem is

$$\begin{aligned} \hat{\beta}^{ridge} =& \text{argmin}_\beta \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{i=1}^{p} x_{ij}\beta_j)^2 \\ & \text{subject to} \sum_{i=1}^{p} \beta_j^2 \leq t \end{aligned} \tag{21}$$

which makes explicit the size constraint on the parameters. By imposing a size constraint on the coefficients, as in (21), this problem is alleviated.

Writing the solution inmmatrix form:

$$\text{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta \tag{22}$$

The ridge regression solutions can be seen to be

$$\hat{\beta}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda I)^{-1}\mathbf{X}^T\mathbf{y} \tag{23}$$

The solution adds a positive constant to the diagnal of $\mathbf{X^TX}$ before inversion. This makes the preoblem nonsingular, even if $\mathbf{X^TX}$ is not of full rank, and was the main motivation for ridge regression when it was first introduced in statistics.

The singular value decomposition of the input matrix $\mathbf{X}$ gives some insight to ridge regression. The SVD of the $N \times p$ matrix has the form

$$\mathbf{X} = \mathbf{UDV^T} \tag{24}$$

Using the singular value decomposition we can have the least square fitted vector is

$$\begin{aligned} \mathbf{X}\hat{\beta}^{ls} =& \mathbf{X}(\mathbf{X^TX})^{-1}\mathbf{X^Ty} \\ =& \mathbf{UU^Ty} \end{aligned} \tag{25}$$

The ridge solution is

$$\begin{aligned} \mathbf{X}\hat{\beta}^{ridge} =& \mathbf{X}(\mathbf{X^TX} + \lambda\mathbf{I})^{-1}\mathbf{X^Ty} \\ =& \mathbf{UD}(\mathbf{D^2} + \lambda\mathbf{I})^{-1}\mathbf{DU^Ty} \\ =& \sum_{j=1}^{p} \mathbf{u_j}\frac{d_j^2}{d_j^2 + \lambda}\mathbf{u_j^Ty} \end{aligned} \tag{26}$$

where u_j are the columns of $\mathbf{U}$. Like linear regression, ridge regression computes the coordinates of $\mathbf{y}$ with respect to the orthonormal basis $\mathbf{U}$. It then shrinks these coordinates by the factors $d_j^2/(d_j^2 + \lambda)$. This means that a greater amount of shrinkage is applied to the coordinates of basis vectors with smaller d_j^2.

4.2 Lasso regression

4.2.1 The lasso

The lasso estimate is defined by:

$$\hat{\beta}^{\text{lasso}} = \arg\min_\beta \left\{ \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \right\}. \tag{27}$$

We can also write the lasso problem in the equivalent Lagrangian form:

$$\begin{aligned}\hat{\beta}^{lasso} =& \text{argmin}_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \\ &\text{subject to} \sum_{j=1}^{p} |\beta_j| \leq t\end{aligned} \tag{28}$$

Since the latter constraints makes the solution nonlinear in the y_i, there is no closed form expression as in ridge regression.

Because of the nature of the constraint, making t sufficiently small will cause some of the coefficients to be exactly zero. Thus the lasso does a kind of continuous subset selection. If t is chosen larger than $t_0 = \sum_1^p |\hat{\beta}_j|$ (where $\hat{\beta}_j = \hat{\beta^{ls}}_j$, the least squares estimates), then the lasso estimates are the $\hat{\beta}_j$.

4.2.2 Pairwise coordinate optimization

Even though lasso problem doesn't have closed form solution, we can get the solution by iteration. The idea of pairwise coordinate optimization is to fix the penalty parameter λ in the Lagrangian form and optimize successively over each parameter, holding the other parameters fixed at their current values.

Suppose the predictors are all standardized to have mean zero and unit norm. Denote by $\tilde{\beta_k}(\lambda)$ the current estimate for β_k at penalty parameter λ. We can rearrange (27) to isolate β_j,

$$R(\tilde{\beta}(\lambda), \beta_j) = \frac{1}{2} \sum_{i=1}^{N} \left(y_i - \sum_{k \neq j} x_{ik}\tilde{\beta}_k(\lambda) - x_{ij}\beta_j \right)^2 + \lambda \sum_{k \neq j} |\tilde{\beta}_k(\lambda)| + \lambda |\beta_j| \tag{29}$$

where we have suppressed the intercept and introduce a factor $\frac{1}{2}$ for convenience. This can be viewed as a univariate lasso problem with response variable the partial residual $y_i - \tilde{y}_i^{(j)} = y_i \sum_{k \neq j} x_{ik}\tilde{\beta}_k(\lambda)$. This has n explicit solution, resulting in the update

$$\tilde{\beta}_j \leftarrow S(\sum_{i=1}^{N} x_{ij}(y_i - \tilde{y}_i^{(j)}), \lambda) \tag{30}$$

Here $S(t, \lambda) = \text{sign}(|t| - \lambda)_+$ is the soft-threshold operator.

4.3 Least angle regression

Least angle regression(LAR) is a relative newcomer, and can be viewed as a kind of "democratic" version of forward stepwise regression . As we will see, LAR is intimately connected with the lasso, and in fact provides an extremely efficient algorithm for computing the entire lasso path.

Forward stepwise regression adding one variable, which is the best variable, at a time.

Least angle regression uses a similar strategy, but only eners as much a predictor as it deserves. At the first step it identifies the variable most correlated with the response. Rather than fit this variable completely, LAR moves the coefficient of this variable continuously toward its leastsquares value (causing its correlation with the evolving residual to decrease in absolute value). As soon as another variable "catches up" in terms of correlation with the residual, the process is paused. The second variable then joins the active set, and their coefficients are moved together in a way that keeps their correlations tied and decreasing. This process is continued until all the variables are in the model, and ends at the full least-squares fit.

Suppose A_k is the active set of variables at the beginning of the kth step, and let β_{A_k} be the coefficient vector for these variables at this step; there will be $k - 1$ nonzero values, and the

Algorithm 4 Least Angle Regression (LARS)

```
1: Standardize the predictors to have mean zero and unit norm.
2: Initialize residual $\mathbf{r} = \mathbf{y} - \bar{\mathbf{y}}$, and set $\beta_1, \beta_2, \dots, \beta_p = 0$
3: while not all predictors entered do
4:     Find predictor $\mathbf{x}_j$ most correlated with $\mathbf{r}$
5:     Move $\beta_j$ from 0 toward its least-squares coefficient $(\mathbf{x}_j^T\mathbf{r})$
6:     while less than all $p$ predictors have been entered do
7:         if some other predictor $\mathbf{x}_k$ has as much correlation with $\mathbf{r}$ as $\mathbf{x}_j$ then
8:             Move $\beta_j$ and $\beta_k$ in their joint least squares direction
9: Continue until all $p$ predictors are entered or until $\min(N-1, p)$ steps reached
```

onejust enter will be zero. If $\mathbf{r_k} = \mathbf{y} - \mathbf{X_{A_k}}\beta_{A_k}$ is the recurrent residue, then the direction for this step is

$$\delta_k = (\mathbf{X^T_{A_k} X_{A_k}})^{-1}\mathbf{X_{A_k} r_k} \tag{31}$$

The coefficient profile then evolves as $\beta_{A_k}(\alpha) = \beta_{A_k} + \alpha \cdot \delta_k$. If the fit vector at the begging of this step is $\hat{\mathbf{f}}_k(\alpha) = \hat{\mathbf{f}}_k + \alpha \cdot \mathbf{u}_k$, where $\mathbf{u_k} = X_{A_k}\delta_k$ is the new direction.

The name "least angle" arises from a geometrical interpretation of this process; $\mathbf{u}_k$ makes the smallest (and equal) angle with each of the predictors in A_k.

proof: We first prove that the direction keeps the correlations tied and decreasing.
Denote $\mathbf{c} = \mathbf{X^T r}$ is the correlation, $\forall j \in A_k$, c_j are the same. When we update the path, the residue becomes

$$\mathbf{r}(\alpha) = \mathbf{X}^T(\mathbf{r_k} - \alpha\mathbf{u_k}) \tag{32}$$

Then the correlation vector is

$$\begin{aligned}\mathbf{c}(\alpha) = \mathbf{X^T r}(\alpha) = \mathbf{X^T}(\mathbf{r_k} - \alpha\mathbf{u_k}) =& \mathbf{X^T r_k} - \alpha\mathbf{X^T u_k} \\ =& \mathbf{X_{A_k} r_k} - \alpha\mathbf{X^T_{A_k} X_{A_k}}\boldsymbol{\delta}_\mathbf{k}\end{aligned} \tag{33}$$

By definition $\delta_k = (\mathbf{X^T_{A_k} X_{A_k}})^{-1}\mathbf{X_{A_k} r_k}$, we get

$$c_{A_k}(\alpha) = c_{A_k}(0) - \alpha c_{A_k}(0) \tag{34}$$

which means the direction keeps the correlations tied and decreasing.

Then we want to prove $\mathbf{u}_k$ makes the smallest (and equal) angle with each of the predictors in A_k. By the last step, we get

$$x_j^T r_k = C_k, \forall j \sin A_k \tag{35}$$

where C_k is a constant. For $x_j^T u_k$ we get

$$\begin{aligned} x_j^T u_k =& x_j^T X_{A_k}\delta_k \\ =& x_j^T X_{A_k}(X_{A_k}^T X_{A_k})^{-1} X_{A_k} r_k \\ =& x_j^T X_{A_k}(X_{A_k}^T X_{A_k})^{-1}(C_k 1_{A_k}) \\ =& C_k \end{aligned} \tag{36}$$

which means that the inner product is a constant. As a result, $\mathbf{u}_k$ makes the smallest (and equal) angle with each of the predictors in A_k.

Back to Lasso regression, we can observe there is some connection between the Lasso and LAR. By LAR, we can get an easy way to get lasso solution.

Suppose A is the active set of variables at some stage in the algorithm, tied in their absolute inner-product with the current residuals $\mathbf{y} - \mathbf{X}\beta$. We can express this as

$$\mathbf{x_j^T}(\mathbf{y} - \mathbf{X}\beta) = \gamma \cdot s_j, \forall j \in A \tag{37}$$

where $s_j = \pm 1$ indicates the sign of inner product.

Algorithm 5 Least Angle Regression with Lasso Modification

```
1: Standardize the predictors to have mean zero and unit norm. Start with the residual $\mathbf{r} = \mathbf{y} - \bar{\mathbf{y}}$, $\beta_1, \beta_2, \ldots, \beta_p = 0$.
2: Find the predictor $\mathbf{x}_j$ most correlated with $\mathbf{r}$.
3: Move $\beta_j$ from 0 towards its least-squares coefficient $(\mathbf{x}_j, \mathbf{r})$, until some other competitor $\mathbf{x}_k$ has as much correlation with the current residual as does $\mathbf{x}_j$.
4: Move $\beta_j$ and $\beta_k$ in the direction defined by their joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until some other competitor $\mathbf{x}_l$ has as much correlation with the current residual.
5: while not all predictors have been entered do
6:     if a non-zero coefficient hits zero then
7:         Drop its variable from the active set of variables.
8:         Recompute the current joint least squares direction.
9:     Continue with the next step of the algorithm.
10: Continue in this way until all $p$ predictors have been entered. After $\min(N-1, p)$ steps, we arrive at the full least-squares solution.
```

For lasso regression, we write it in vector form:

$$R(\beta) = \frac{1}{2}||\mathbf{y} - \mathbf{X}\beta||_2^2 + \lambda||\beta||_1 \tag{38}$$

Let B be the active set of variables in the solution for a given value of λ. For these variables $R(\beta)$ is differentiable, and the stationarity conditions give

$$\mathbf{x_j^T}(\mathbf{y} - \mathbf{X}\beta) = \lambda \cdot \text{sign}(\beta_j), \forall j \in B \tag{39}$$

Compare (38) with (39), we see that they are identical only if the sign of β_j matches the sign of the inner product. That is why the LAR algorithm and lasso start to differ when an active coefficient passes through zero.

4.4 Experiment

In this part, we still use Hitter dataset to show lasso and ridge regression.

4.4.1 ridge regression

First, we use ridge regression. Setting $\lambda = 5$, we get the coeffcients that are shown below: Then separate them into 2 set, one is training set and another is testing set, we also set $\lambda = 5$.

Table 2: Regression Coefficients

Variable	Coefficient	Variable	Coefficient
(Intercept)	150.1571	AtBat	-1.6300
Hits	5.7031	HmRun	0.7439
Runs	-0.4024	RBI	0.0369
Walks	5.2699	Years	-10.1602
CAtBat	-0.0604	CHits	0.2047
CHmRun	0.7103	CRuns	0.7219
CRBI	0.3727	CWalks	-0.6151
LeagueN	61.5045	DivisionW	-122.5148
PutOuts	0.2797	Assists	0.2936
Errors	-3.7510	NewLeagueN	-27.8742

Finally we get training loss is 142199.2 and testing loss is 224669.9. By cross-validation, we can get the best λ is 326, wise the testing MSE 139856.6. The coefficients are

Table 3: Regression Model Coefficients

Variable	Coefficient	Variable	Coefficient
(Intercept)	15.44383120	AtBat	0.07715547
Hits	0.85911582	HmRun	0.60103106
Runs	1.06369007	RBI	0.87936105
Walks	1.62444617	Years	1.35254778
CAtBat	0.01134999	CHits	0.05746654
CHmRun	0.40680157	CRuns	0.11456224
CRBI	0.12116504	CWalks	0.05299202
LeagueN	22.09143197	DivisionW	-79.04032656
PutOuts	0.16619903	Assists	0.02941950
Errors	-1.36092945	NewLeagueN	9.12487765

4.4.2 lasso regression

Similarly, we use the dataset to do cross-validation, get the best $\lambda = 9.28$. Then we get the final MSE IS 143673, and the result is where we notice there are many zeros, which means lasso

Table 4: Regression Model Coefficients

Variable	Coefficient	Variable	Coefficient
(Intercept)	1.27479059	AtBat	-0.05497143
Hits	2.18034583	HmRun	0.00000000
Runs	0.00000000	RBI	0.00000000
Walks	2.29192406	Years	-0.33806109
CAtBat	0.00000000	CHits	0.00000000
CHmRun	0.02825013	CRuns	0.21628385
CRBI	0.41712537	CWalks	0.00000000
LeagueN	20.28615023	DivisionW	-116.16755870
PutOuts	0.23752385	Assists	0.00000000
Errors	-0.85629148	NewLeagueN	0.00000000

regression can always produce sparse solution, leading to dimension reduction.

5 Dimension reduction methods

The methods that we have discussed so far in this chapter have controlled variance in two different ways, either by using a subset of the original variables, or by shrinking their coefficients toward zero. We now explore a method that transform variables. We refer to these methods as dimension reduction methods.

Let $Z_1, \dots, Z_M$ represent $M < p$ linear combinations of our original p predictors.

$$Z_m = \sum_{j=1}^{p} \phi_{jm} X_j \tag{40}$$

for some constants $\phi_{1m}, \dots, \phi_{pm}, m = 1, \dots, M$. We can fit the linear regression model

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i, i = 1, \dots, n \tag{41}$$

If the constants $\phi_{1m}, \dots, \phi_{pm}$ are chosen wisely, then such dimension reduction approached can often outperform least squares regression.

The term dimension reduction comes from the fact that the method reduces $p+1$ coefficients $\beta_0, \ldots, \beta_p$ to $M+1$ parameters $\theta_0, \ldots, \theta_M$. Notice that from (40)

$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{jm} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{jm} x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij}, \tag{42}$$

where

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{jm} \tag{43}$$

Hence (41) can be thought of as a special case of the original linear regression model given by (1).

5.1 Principal component regression

Principal Component regression is the most widely used data dimensionality reduction algorithm. The main idea of PCA is to map n-dimensional features onto k dimensions. The work of PCR is to sequentially find a set of mutually orthogonal coordinate axes in the original space, and the selection of new coordinate axes is closely related to the data itself. Among them, the first new coordinate axis is chosen as the direction with the largest variance in the original data. The second new coordinate axis is selected as the plane perpendicular to the first coordinate axis that has the largest variance. The third axis is the plane perpendicular to the first and second axes that has the largest variance. By analogy, we can obtain n such coordinate axes. Through this method, we find that most of the variance is contained in the first k coordinate axes, while the variance of the subsequent coordinate axes is almost zero. Therefore, we can ignore the remaining coordinate axes and only retain the first k coordinate axes that contain most of the variance. In fact, this is equivalent to retaining only the dimensional features that contain most of the variance and ignoring the feature dimensions whose variance is almost zero, thereby achieving dimensionality reduction processing of data features.

Algorithm 6 Principal Component Analysis (PCA)

```
1: Input: Data matrix $\mathbf{X} \in R^{n\times p}$
2: Output: Principal components $\mathbf{Z}$ and loadings $\mathbf{V}$
3: Center the data: subtract the column mean from each variable in $\mathbf{X}$
4: Compute the covariance matrix $\mathbf{\Sigma} = \frac{1}{n}\mathbf{X}^T\mathbf{X}$
5: Perform eigen decomposition: $\mathbf{\Sigma} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$
6: Sort the eigenvalues in descending order, and reorder columns of $\mathbf{V}$ accordingly
7: Select the top $k$ eigenvectors $\mathbf{V}_k$ corresponding to the top $k$ eigenvalues
8: Project the data onto the top $k$ principal components: $\mathbf{Z} = \mathbf{X}\mathbf{V}_k$
```

5.2 Partial least square

The PCR approach that we just described involves identifying linear combinations, or directions, that best represent the predictors $X_1, \ldots, X_p$. These directions are identified in an unsupervised way, since the response Y is not used to help determine the principal component directions. That is, the response does not supervise the identification of the principal components. Consequently, PCR suffers from a drawback: there is no guarantee that the directions that best explain the predictors will also be the best directions to use for predicting the response.

Unlike PCR, PLS identifies these new features in a supervised way—that is, it makes use of the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response. Roughly speaking, the PLS approach attempts to find directions that help explain both the response and the predictors.

Images/pcr.png

Figure 5: Principal component regression

Algorithm 7 Partial Least Squares Regression

```
 1: Input: Centered predictor matrix $\mathbf{X} \in R^{n\times p}$, centered response vector $\mathbf{y} \in R^n$
 2: Output: Weight vectors $\mathbf{w}_1, \dots, \mathbf{w}_k$, score vectors $\mathbf{t}_1, \dots, \mathbf{t}_k$, loading vectors $\mathbf{p}_1, \dots, \mathbf{p}_k$,
    regression coefficients $\boldsymbol{\beta}$
 3: for $i = 1$ to $k$ do
 4:     Initialize score vector $\mathbf{t}_i = \mathbf{X}_{i-1}\mathbf{w}_{i-1}$ or use $\mathbf{y}$ as initial guess
 5:     repeat
 6:         $\mathbf{w}_i = \frac{\mathbf{X}_{i-1}^T\mathbf{t}_i}{\|\mathbf{X}_{i-1}^T\mathbf{t}_i\|}$
 7:         $\mathbf{t}_i = \mathbf{X}_{i-1}\mathbf{w}_i$
 8:         $c_i = \frac{\mathbf{y}_{i-1}^T\mathbf{t}_i}{\|\mathbf{t}_i\|^2}$
 9:         $\mathbf{u}_i = c_i\mathbf{t}_i$
10:     until convergence of $\mathbf{t}_i$
11:     $\mathbf{p}_i = \frac{\mathbf{X}_{i-1}^T\mathbf{t}_i}{\|\mathbf{t}_i\|^2}$
12:     Deflate: $\mathbf{X}_i = \mathbf{X}_{i-1} - \mathbf{t}_i\mathbf{p}_i^T$, $\mathbf{y}_i = \mathbf{y}_{i-1} - c_i\mathbf{t}_i$
13: Compute regression coefficients: $\boldsymbol{\beta} = \sum_{i=1}^k \mathbf{w}_i \left(\frac{c_i}{\mathbf{w}_i^T\mathbf{p}_i}\right)$
```

5.3 Comparison of PCR and PLS

What optimization problem is this 2 methods are solving? For PLR, the mth principal component direction v_m solves:

$$\begin{gathered}\max_{\alpha} \operatorname{Var}(\mathbf{X}\alpha) \\ \text{subject to } \|\alpha\| = 1, \quad \alpha^\top S v_l = 0,\ l = 1, \dots, m-1\end{gathered} \tag{44}$$

where S is the sample covariance mstrix of the $\mathbf{x_j}$.
The mth PLS direction $\hat{\varphi}_m$ solves:

$$\begin{gathered}\max_\alpha \operatorname{Corr}^2(y, X\alpha)\operatorname{Var}(X\alpha) \\ \text{subject to} \|\alpha\| = 1, \quad \alpha^T S\hat{\varphi}_\ell = 0, \ell = 1, \dots, m-1\end{gathered} \tag{45}$$

Further analysis reveals that the variance aspect tends to dominate, and so partial least squares behaves much like ridge regression and principal components regression.

If the input matrix $\mathbf{X}$ is orthogonal, then partial least squares finds the least squares estimates after $m = 1$ steps. Subsequent steps have no effect since the $\hat{\varphi}_{mj}$ are zero for m ¿ 1

5.4 Experiment

In this section we will compare this 2 dimension reduction methods, and show how the princial component explain the variance of inputs and outputs.
Here we use principal component regression we show how variance is explained by the number of principal components.

Table 5: Percentage of Variance Explained by Principal Components (First 10 Components)

	Number of Components									
	1	2	3	4	5	6	7	8	9	10
X	38.31	60.16	70.84	79.03	84.29	88.63	92.26	94.96	96.28	97.26
Salary	40.63	41.58	42.17	43.22	44.90	46.48	46.69	46.75	46.86	47.76

We notice that only 7 principal components can explain 90% of the input variance, which means PCR is efficient dimension reduction method.
Then we use partial least square to do the similar thing.

Table 6: Percentage of Variance Explained by PLS Components (First 10 Components)

	Number of Components									
	1	2	3	4	5	6	7	8	9	10
X	38.08	51.03	65.98	73.93	78.63	84.26	88.17	90.12	92.92	95.00
Salary	43.05	46.40	47.72	48.71	50.53	51.66	52.34	53.26	53.52	53.77

We notice that PLS use 8 principal components to explain 90% of the input variance, but for the output, the PLS can do better because it maximiae the correlation of input and output.

6 Conclusion

In this semester, I have systematically explored various linear regression techniques, progressing from fundamental concepts to advanced methodologies. The study began with the basic least squares (LS) model, then extended to discrete approaches such as subset selection methods (best, forward, backward stepwise selection). Subsequently, shrinkage methods including ridge regression and LASSO were investigated, followed by dimension reduction techniques such as principal component regression (PCR) and partial least squares (PLS).

The core principles, mathematical formulations, and algorithmic implementations of each method were thoroughly examined. Through theoretical analysis and empirical validation using the Hitters dataset, the comparative performance of these methods was evaluated in terms of prediction accuracy, model interpretability. Key findings demonstrate the trade-offs between bias and variance across different approaches, highlighting how shrinkage and dimension reduction methods can effectively address multicollinearity while maintaining model simplicity.

This comprehensive study provides valuable insights into selecting appropriate regression techniques based on dataset characteristics and modeling objectives. Future work could explore ensemble methods and nonlinear extensions to further enhance predictive performance.

A Code for all experiment

James et al. [2013]

```
##Use Hitter dataset to do linear regression
library (ISLR)

##preprocessing
fix(Hitters)
names(Hitters)
dim (Hitters)
sum(is.na(Hitters$Salary))
Hitters = na.omit(Hitters)
dim(Hitters)
sum(is.na(Hitters))

##BSS
library(leaps)
regfit.full = regsubsets(Salary~.,Hitters)
summary(regfit.full)
regfit.full = regsubsets(Salary~.,data=Hitters,nvmax=19)
reg.summary = summary(regfit.full)
names(reg.summary )
reg.summary$rsq

par(mfrow=c(2,2))
plot(reg.summary$rss,xlab ="Number␣of␣Variables",ylab ="␣RSS␣",type ="l")
plot(reg.summary$adjr2,xlab ="Number␣of␣Variables",ylab ="Adjusted␣RSq",
    type ="l")

which.max(reg.summary$adjr2)
points(11,reg.summary$adjr2[11],col="␣red␣",cex =2,pch =20)

plot(reg.summary$cp,xlab ="Number␣of␣Variables", ylab ="Cp",type="l")
which.min(reg.summary$cp)
points(10,reg.summary$cp[10],col ="red",cex =2,pch =20)

which.min(reg.summary$bic)
plot(reg.summary$bic,xlab ="Number␣of␣Variables",ylab ="BIC",type="l")
points (6,reg.summary$bic[6],col ="red",cex =2,pch =20)

plot(regfit.full,scale ="r2")
plot(regfit.full,scale ="adjr2")
plot(regfit.full,scale ="Cp")
plot(regfit.full,scale ="bic")
coef(regfit.full,6)

##FSS
regfit.fwd = regsubsets(Salary~.,data= Hitters,nvmax =19, method ="forward
    ")
summary(regfit.fwd)

#backSS
regfit.bwd=regsubsets(Salary~.,data= Hitters,nvmax =19,method ="backward")
summary(regfit.bwd)

##ridge
x = model.matrix(Salary~., Hitters)[,-1]
y = Hitters$Salary
```

```
library(glmnet)
grid = 10^seq(10,-2,length=100)
ridge.mod = glmnet(x,y,alpha=0, lambda=grid)

dim(coef(ridge.mod))

predict(ridge.mod, s = 4, type = "coefficients")[1:20,]

set.seed(1)
train = sample(1:nrow(x), nrow(x)/2)
test = (-train)
y.test = y[test]

ridge.mod=glmnet(x[train,], y[train], alpha =0, lambda =grid ,thresh =1e
    -12)
ridge.pred=predict(ridge.mod,s=4,newx=x[test,])
mean((ridge.pred-y.test)^2)
mean((mean(y[train])-y.test)^2)

set.seed(1)
cv.out =cv.glmnet (x[train,], y[train], alpha=0)
plot(cv.out)
bestlam =cv.out$lambda.min
bestlam

ridge.pred= predict(ridge.mod ,s=bestlam , newx=x[test,])
mean((ridge.pred -y.test)^2)
out = glmnet(x,y,alpha =0)
predict(out , type ="coefficients",s= bestlam) [1:20,]

##lasso
lasso.mod =glmnet(x[train,], y[train], alpha=1, lambda=grid)
plot(lasso.mod)
set.seed (1)
cv.out =cv.glmnet (x[train ,], y[train], alpha=1)
plot(cv.out)
bestlam =cv.out$lambda.min
lasso.pred=predict(lasso.mod ,s=bestlam , newx=x[test,])
mean ((lasso.pred-y.test)^2)
out = glmnet(x,y, alpha =1, lambda = grid)
lasso.coef= predict (out ,type ="coefficients",s= bestlam ) [1:20,]
lasso.coef
bestlam

##pcr
library(pls)
set.seed(2)
pcr.fit=pcr(Salary~., data = Hitters, scale = TRUE, validation ="CV")
summary(pcr.fit)
##pls
set.seed (1)
pls.fit =plsr(Salary~.,dat =Hitters, scale=TRUE , validation ="CV")
summary (pls.fit )
```

References

T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

G. James, D. Witten, T. Hastie, R. Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.