

Advanced Computational Econometrics: Machine Learning

Chapter 3: Model Selection

Thierry Denœux

Spring 2023



Need for model selection

- Consider, for instance, a regression problem with a response variable Y and 3 predictors X_1, X_2, X_3 .
- We can consider many (an infinity of) models, such as

$$Y = \beta_0 + \beta_1 X_1 + \epsilon$$

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon$$

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_1^2 + \beta_5 X_2^2 + \\ \beta_6 X_1 X_2 + \beta_7 X_3^2 + \beta_8 X_1 X_3 + \beta_9 X_2 X_3 + \epsilon$$

\vdots

Which model to choose?



Bias-variance trade-off

- We have seen that a more complex model will not always have a smaller error when applied to test data.
- This is due to the **bias-variance trade-off**: when the number of parameters increases, the bias of the model decreases, but the variance increases.
- Furthermore, a simpler model often has a distinct advantage in terms of its interpretability.
- In this chapter, we discuss some tools to select models that will be
 - Complex enough to fit the data, but
 - Not too complex to avoid overfitting and to be interpretable.
- We focus mainly on **linear regression**, but the tools can be adapted to classification.



Three classes of methods

Subset Selection: We identify a **subset** of the p predictors that we believe to be related to the response. We then fit a model using the reduced set of variables.

Regularization: We fit a model involving all p predictors, but the estimated coefficients are shrunk towards zero to obtain a smoother prediction function. This **regularization** (also known as **shrinkage**) has the effect of reducing variance and can also perform variable selection.

Feature extraction: We project the p predictors onto a **q -dimensional subspace**, where $q < p$. This is achieved by computing q different linear combinations, or projections, of the variables. Then these q projections (features) are used as predictors to fit a linear model.



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Best subset selection

- ① Let \mathcal{M}_0 denote the null model, which contains no predictors. This model simply predicts the sample mean for each observation.
- ② For $k = 1, 2, \dots, p$:
 - ① Fit all $\binom{p}{k}$ models that contain exactly k predictors.
 - ② Pick the best among these $\binom{p}{k}$ models, and call it \mathcal{M}_k . Here “best” is defined as having the smallest RSS, or equivalently the largest R^2 .
- ③ Select a single best model from among $\mathcal{M}_0, \dots, \mathcal{M}_p$. (How? to be seen later).



Example: air pollution and mortality

- Data are from McDonald and Schwing (1973), “Instabilities of Regression Estimates Relating Air Pollution to Mortality”, *Technometrics*, 15, 463-481.
- This data set of 15 predictors and a measure of mortality in 60 US metropolitan areas in 1959-1961.



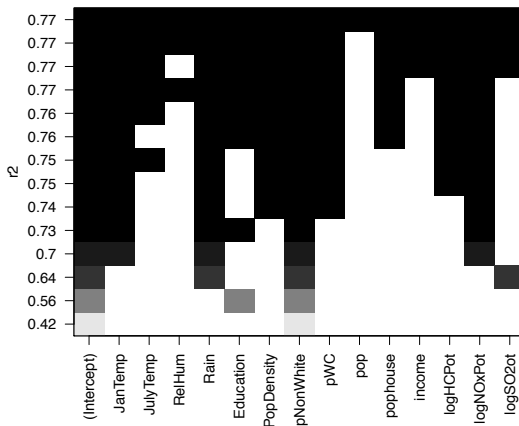
Variables

- Response: Total Age Adjusted Mortality Rate
- Predictors:
 - 1 Mean annual precipitation in inches
 - 2 Mean January temperature in degrees Fahrenheit
 - 3 Mean July temperature in degrees Fahrenheit
 - 4 Percent of 1960 SMSA population that is 65 years of age or over
 - 5 Population per household, 1960 SMSA
 - 6 Median school years completed for those over 25 in 1960 SMSA
 - 7 Percent of housing units that are found with facilities
 - 8 Population per square mile in urbanized area in 1960
 - 9 % of 1960 urbanized area population that is non-white
 - 10 % employment in white-collar occupations in 1960 urbanized area
 - 11 % of families with income under 3,000 in 1960 urbanized area
 - 12 Relative population potential of hydrocarbons, HC
 - 13 Relative pollution potential of oxides of nitrogen, NO_x
 - 14 Relative pollution potential of sulfur dioxide, SO₂
 - 15 Percent relative humidity, annual average at 1 p.m.



Best subset selection in R

```
library('leaps')
reg.fit<-regsubsets(Mortality~.-logNOx,data=pollution,method='exhaustive',nvmax=15)
plot(reg.fit,scale="r2")
```



Extension to other models

- Although we have presented best subset selection here for least squares regression, the same ideas apply to other types of models, such as logistic regression.
- The **deviance**, $-2\ell(\hat{\theta})$, plays the role of RSS for a broader class of models.



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Stepwise selection

- For computational reasons, best subset selection cannot be applied with very large p .
- Best subset selection may also suffer from statistical problems when p is large: larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data.
- Thus an enormous search space can lead to overfitting and high variance of the coefficient estimates.
- For both of these reasons, **stepwise methods**, which explore a far more restricted set of models, are attractive alternatives to best subset selection.



Forward stepwise selection

- **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.



Forward stepwise selection in detail

- 1 Let \mathcal{M}_0 denote the null model, which contains no predictors.
- 2 For $k = 0, \dots, p - 1$:
 - 1 Consider all $p - k$ models that augment the predictors in \mathcal{M}_k with one additional predictor.
 - 2 Choose the best among these $p - k$ models, and call it \mathcal{M}_{k+1} . Here “best” is defined as having smallest RSS or highest R^2 .
- 3 Select a single best model from among $\mathcal{M}_0, \dots, \mathcal{M}_p$. (How? to be seen later).



More on forward stepwise selection

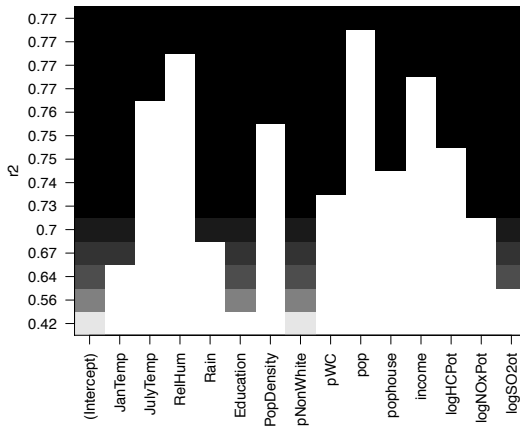
- Computational advantage over best subset selection is clear.
- It is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors.
- In contrast to best subset selection, the models are nested:

$$\mathcal{M}_0 \subset \dots \subset \mathcal{M}_p$$



Forward stepwise selection in R

```
reg.fit<-regsubsets(Mortality~.-logNOx,data=pollution,method='forward',nvmax=15)  
plot(reg.fit,scale="r2")
```



Backward stepwise 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.



Backward stepwise selection in detail

- 1 Let \mathcal{M}_p denote the full model, which contains all p predictors.
- 2 For $k = p, p - 1, \dots, 1$:
 - 1 Consider all k models that contain all but one of the predictors in \mathcal{M}_k , for a total of $k - 1$ predictors.
 - 2 Choose the best among these k models, and call it \mathcal{M}_{k-1} . Here “best” is defined as having smallest RSS or highest R^2 .
- 3 Select a single best model from among $\mathcal{M}_0, \dots, \mathcal{M}_p$. (How? to be seen later).



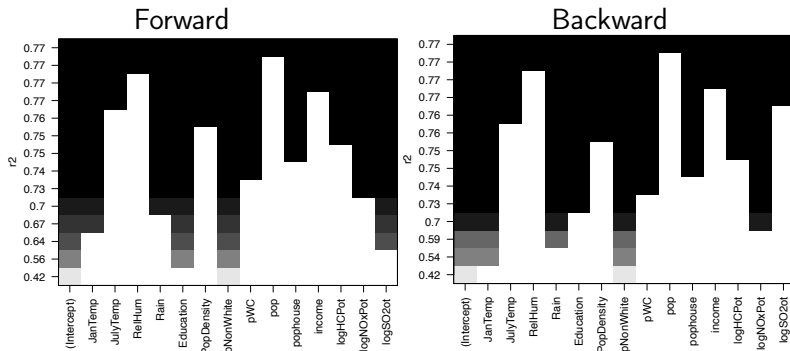
More on backward stepwise selection

- Like forward stepwise selection, the backward selection approach searches through only $1 + p(p+1)/2$ models, and so can be applied in settings where p is too large to apply best subset selection
- Like forward stepwise selection, backward stepwise selection is **not guaranteed to yield the best model** containing a subset of the p predictors.
- Backward stepwise selection requires that the number of samples n is larger than the number of variables p (so that the full model can be fit). In contrast, forward stepwise selection can be used even when $n < p$, and so is the only viable subset method when p is very large.



Backward stepwise selection in R

```
reg.fit<-regsubsets(Mortality~.-logNOx,data=pollution,method='backward',nvmax=15)
plot(reg.fit,scale="r2")
```



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 **Choosing the optimal model**
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Choosing the optimal model

- Each of the subset selection procedures returns a sequence of models \mathcal{M}_k indexed by model size $k = 0, 1, 2, \dots, p$. Our job here is to select \hat{k} . Once selected, we will return model $\mathcal{M}_{\hat{k}}$.
- Which criterion for model selection?
- The model containing all of the predictors will always have the smallest RSS and the largest R^2 , since these quantities are related to the training error.
- We wish to **choose a model with low prediction error**, not a model with low training error. Recall that training error is usually a poor estimate of prediction error.
- Therefore, RSS and R^2 are not suitable for selecting the best model among a collection of models with different numbers of predictors.



Estimating prediction error: two approaches

- We can
 - 1 Indirectly estimate prediction error by making an **adjustment** to the training error to account for the bias due to overfitting, or
 - 2 Directly estimate the prediction error, using either the **hold-out** method or **cross-validation**.
- We illustrate both approaches next.



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



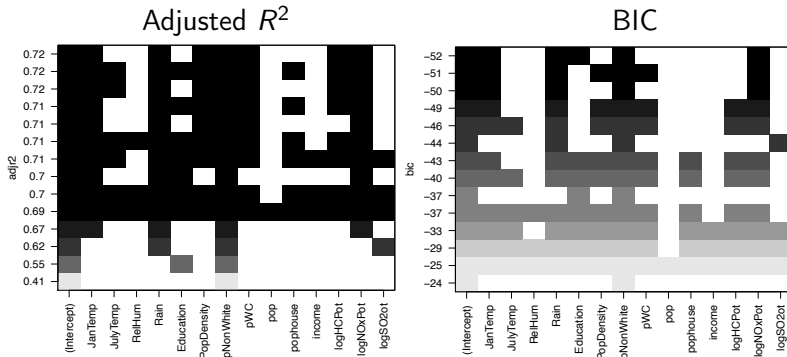
Training error adjustment techniques

- These techniques **adjust the training error for the model size**, and can be used to select among a set of models with different numbers of variables.
- Three criteria:
 - 1 Adjusted R^2
 - 2 Akaike information criterion (AIC)
 - 3 Bayesian information criterion (BIC)
- The next figure displays BIC, and adjusted R^2 for the best model of each size produced by best subset selection on the pollution data set.



Example

```
reg.fit<-regsubsets(Mortality~.-logNOx,data=pollution,method='exhaustive',nvmax=15)
plot(reg.fit,scale="adjr2") plot(reg.fit,scale="bic")
```



Adjusted R -squared

- The usual R^2 is

$$R^2 = 1 - \frac{RSS/n}{TSS/n}$$

- It can be seen as an estimate of the “population R^2 ” defined as

$$R_{pop}^2 = 1 - \frac{\sigma^2}{\text{Var}(Y)}$$

- The R^2 uses biased estimates of the residual and total variances. The **adjusted R^2** is based on unbiased estimates:

$$\bar{R}^2 = 1 - \frac{RSS/(n - p - 1)}{TSS/(n - 1)}$$

where p is the number of predictors used.

- This criterion is specific to regression.



AIC

- The **AIC criterion** is defined for a large class of models fit by maximum likelihood:

$$AIC = -2\ell(\hat{\theta}) + 2r$$

where $\ell(\hat{\theta})$ is the maximized value of the log-likelihood function for the estimated model, and r is the number of parameters.

- The best model has the smallest AIC value.
- For linear regression with p variables and a constant term, $r = p + 2$ ($p + 1$ coefficients and the variance σ^2 or the error term).



BIC

- Definition:

$$BIC = -2\ell(\hat{\theta}) + r \log(n)$$

where r is the number of parameters.

- Like AIC, BIC will tend to take on a small value for a model with a low prediction error, and so generally we select the model that has the lowest BIC value.
- Notice that BIC replaces the $2r$ used by AIC with a $r \log(n)$ term, where n is the number of observations.
- Since $\log n > 2$ for any $n > 7$, the BIC statistic generally places a heavier penalty on models with many variables, and hence results in the selection of smaller models than AIC.



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Direct estimation of the prediction error

- We compute an **estimate of the prediction error** for each model \mathcal{M}_k under consideration, and then select the k for which the resulting estimated prediction error is smallest.
- This procedure has an advantage relative to AIC, BIC, and adjusted R^2 , in that it provides a **direct estimate of the prediction error**.
- It can also be used in a wider range of model selection tasks, even in cases where it is hard to pinpoint the model degrees of freedom.



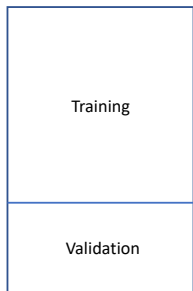
Direct estimation of the prediction error

Two methods:

- 1 Validation-set (hold-out) approach
- 2 Cross-validation



Validation-set approach



- Here we randomly divide the available set of samples into two parts:

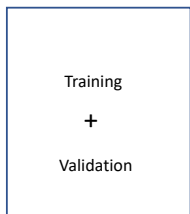
- 1 a training set and
- 2 a **validation** set

- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set.
- The resulting validation-set error provides an estimate of the prediction error. This is typically assessed using MSE in the case of regression and error rate in the case of classification.

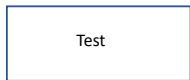


Hold-out approach (continued)

- After the best model has been selected, it is usually **fit on the whole data (training+validation)**.
- The validation error for the best model is biased (optimistic).



+



- The error of the best model has to be estimated using an independent **test set**.



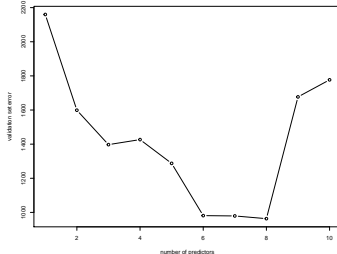
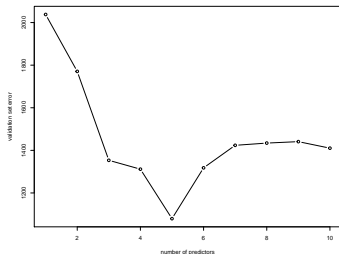
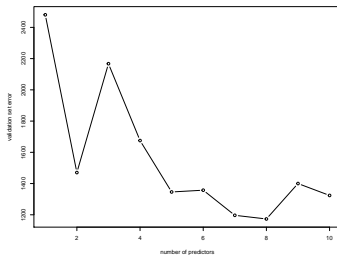
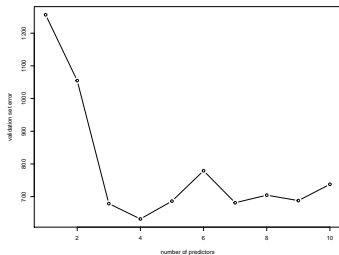
Example

```
n <- nrow(pollution)
ntrain <- 45
nval <- n-ntrain
train <- sample(1:n,ntrain)
pollution.train <- pollution[train,]
pollution.val <- pollution[-train,]
X <- model.matrix(Mortality~.-logNOx,data=pollution)
X.test <- X[-train,]

reg.fit<-regsubsets(Mortality~.-logNOx,data=pollution.train,method='forward',
nvmax=15)
res <- summary(reg.fit)
for(i in 1:10){
model <- res$which[i,]
ypred<-X.test[,model]*%coef(reg.fit,i)
err[i]<-mean((pollution.test$Mortality-ypred)^2)
}
plot(err,type='b',xlab='number of predictors',ylab='validation set error')
```



Results with 4 different splits (Air pollution data)



Limitations of the hold-out approach

- The validation estimate of the prediction error can be **highly variable**, depending on which observations are included in the training set.
- In the hold-out approach, only a subset of the observations – those that are included in the training set – are used to fit the model.
- Consequently, the validation-set error tends to **overestimate the prediction error** for the model fit on the entire data set.



K-fold cross-validation

- Widely used approach for estimating prediction error.
- Estimates can be used to select the best model, and to give an idea of the prediction error of the final chosen model.
- Idea is to randomly divide the data into K equal-sized subsets. We leave out subset k , fit the model to the other $K - 1$ subsets (combined), and then obtain predictions for the left-out k -th subset.
- This is done in turn for each subset $k = 1, 2, \dots, K$, and then the results are combined.

1	2	3	4	5
Validation	Train	Train	Train	Train



K -fold cross-validation in detail

- Let the K subsets be C_1, C_2, \dots, C_K , where C_k denotes the indices of the observations in subset k . There are n_k observations in subset k : if n is a multiple of K , then $n_k = n/K$.
- Compute

$$CV_{(K)} = \frac{1}{K} \sum_{k=1}^K \text{MSE}_k,$$

where

$$\text{MSE}_k = \frac{1}{n_k} \sum_{i \in C_k} \left(y_i - \hat{y}_i^{(-k)} \right)^2$$

and $\hat{y}_i^{(-k)}$ is the fitted value for observation i , obtained from the data with subset k removed.

- Setting $K = n$ yields n -fold or **leave-one-out** cross-validation (LOOCV).



Special case

- With least-squares linear regression, a shortcut makes the cost of LOOCV the same as that of a single model fit!
- The following formula holds:

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{1 - h_i} \right)^2,$$

where \hat{y}_i is the i th fitted value from the original least squares fit, and h_i is the **leverage** (diagonal term of the “hat” matrix). (Reminder: $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, and $\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}$).

- This is like the ordinary MSE, except the i -th residual is divided by $1 - h_i$.



Choice of K

- Since each training set is only $(K - 1)/K$ as big as the original training set, the estimates of prediction error will typically be **biased upward**.
- This bias is minimized when $K = n$ (LOOCV), but this estimate has high variance, because the estimates from each fold are highly correlated.
- $K = 5$ or 10 provides a good compromise for this bias-variance tradeoff.



Standard error of the CV estimate

- The variance of the MSE computed for one subset can be estimated by

$$\widehat{\text{Var}}(\text{MSE}) = \frac{1}{K-1} \sum_{k=1}^K (\text{MSE}_k - \overline{\text{MSE}})^2$$

- An estimate of the variance of the mean of K independent estimates is, thus, $\widehat{\text{Var}}(\text{MSE})/K$. Consequently, the **standard error** of the CV error can be estimated by

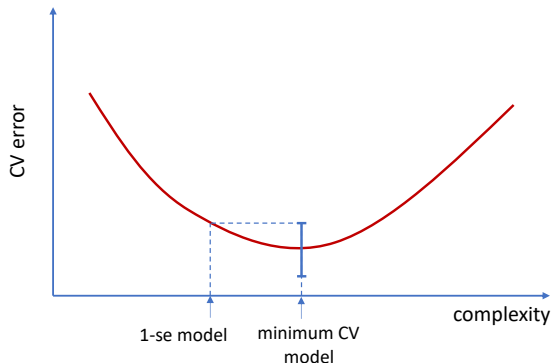
$$\widehat{\text{se}}(\text{CV}_{(K)}) = \sqrt{\frac{\widehat{\text{Var}}(\text{MSE})}{K}}$$

- This is not completely rigorous because the MSE_k , $k = 1, \dots, K$ are not independent.



One-standard-error rule

- Calculate the standard error of the estimated test MSE for each model size
- Select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve



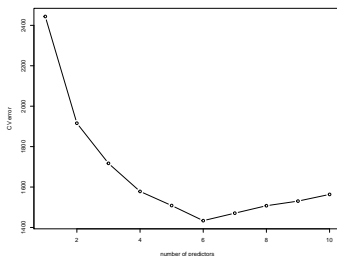
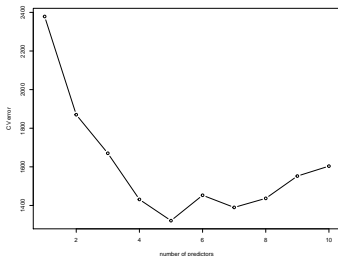
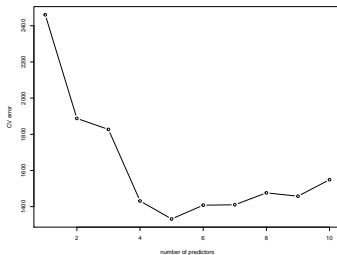
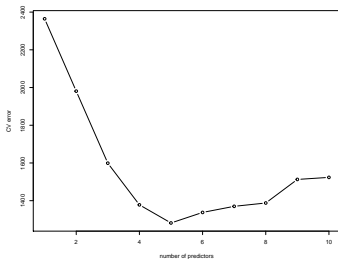
Example of 10-fold cross-validation

```
K <- 10
folds <- sample(1:K,n,replace=TRUE)
MSE <- matrix(0,10,K)

for(k in (1:K)){
  reg <- regsubsets(Mortality~.-logNOx,data=pollution[folds!=k,],
  method='forward',nvmax=15)
  res <- summary(reg)
  for(i in (1:10)){
    model <- res$which[i,] ypred <- X[folds==k,model]%*%coef(reg,i)
    MSE[i,k] <- mean((pollution$Mortality[folds==k]-ypred)^2)
  }
}
CV <- rowMeans(MSE)
```



Result (4 trials)



Final remarks on cross-validation

- The CV error estimates can be averaged over r repetitions of K -fold cross-validation with different random partitions, to reduce the variance of the CV error estimates.
- After the best model has been selected, we usually re-estimate the model parameters using the whole training set.
- To obtain an unbiased estimate of the best model's error, we need an **independent test set**, or **nested cross-validation**.



Nested cross-validation

- Two nested loops: an outer loop of K folds and an inner loop of K' folds.
- The data is first split into K outer subsets.
- One by one, an outer subset is selected (outer loop); the remaining $K - 1$ outer subsets are pooled and split into K' inner subsets.
- Model selection is performed by K' -fold CV (inner loop).
- The best model is fit on $K - 1$ outer subsets and its performance is evaluated using the outer test set.
- After the outer loop has been completed, the error is averaged over the K outer subsets.



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Shrinkage methods

- By retaining only a subset of the predictors, 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 does not always reduce the prediction error of the full model.
- **Shrinkage methods** are more continuous, and do not suffer as much from high variability.
- Two main methods:
 - 1 Ridge regression
 - 2 Lasso



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Ridge regression

- **Ridge regression** shrinks the regression coefficients by imposing a penalty on their size. The ridge coefficients minimize a penalized residual sum of squares:

$$\hat{\beta}^{\text{ridge}} = \arg \min_{\beta} \left\{ \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 \right\}$$

- Here $\lambda \geq 0$ is a **regularization coefficient (hyperparameter)**, which controls the amount of shrinkage: the larger the value of λ , the greater the amount of shrinkage. The parameters β_j are shrunk toward zero (and each other), i.e., to the simplest model (with only the constant term).
- Selecting a good value for λ is critical; cross-validation can be used for this.



Equivalent form

- An equivalent way to write the ridge problem is

$$\hat{\beta}^{\text{ridge}} = \arg \min_{\beta} \left\{ \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 \right\}$$

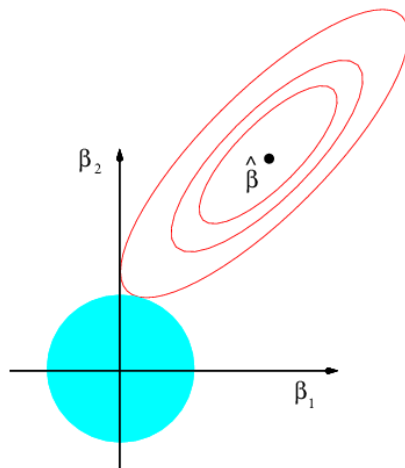
$$\text{subject to } \sum_{j=1}^p \beta_j^2 \leq t,$$

which makes explicit the size constraint on the parameters. (See next slide).

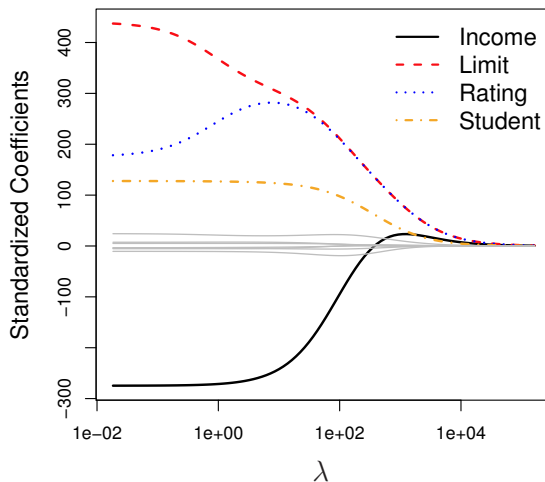
- There is a one-to-one correspondence between parameters t and λ in the previous formulation.



Ridge regression as a constrained optimization problem



The effect of ridge regression



Derivation of the ridge regression estimates

- We can show that $\hat{\beta}^{\text{ridge}}$ can be found by separating the minimization problem into two parts, after centering the inputs (replacing x_{ij} by $x_{ij} - \bar{x}_j$):
 - 1 We estimate β_0 by $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$
 - 2 The remaining coefficients get estimated by a ridge regression without intercept, using the centered x_{ij} and the centered y_i .
- We assume that both the inputs and the output have been centered, so that the input matrix \mathbf{X} has p (rather than $p + 1$) columns, and \mathbf{y} is the n -vector of centered outputs.
- The criterion can be written in matrix form

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



Derivation of the ridge regression estimates (continued)

- The criterion can be rewritten as

$$\text{RSS}_\lambda(\beta) = \mathbf{y}^T \mathbf{y} - 2\beta^T \mathbf{X}^T \mathbf{y} + \beta^T (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}_p) \beta$$

- Differentiating with respect to β we obtain

$$\frac{\partial \text{RSS}_\lambda(\beta)}{\partial \beta} = -2\mathbf{X}^T \mathbf{y} + 2(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}_p) \beta$$

- The solution of the equation $\frac{\partial \text{RSS}_\lambda(\beta)}{\partial \beta} = 0$ is

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



Effective degrees of freedom

- As with the least-squares method, the fitted values are linear functions of the y_i

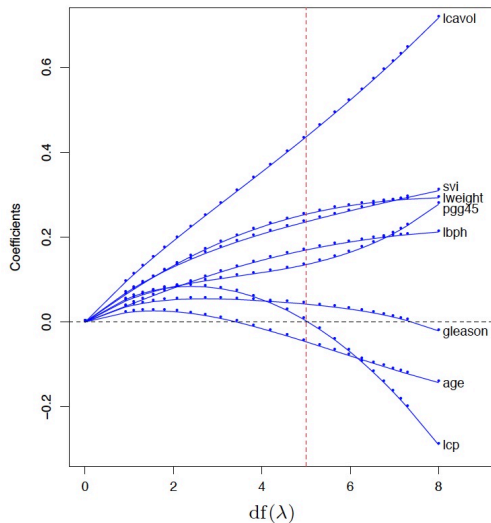
$$\hat{\mathbf{y}}^{\text{ridge}} = \mathbf{X} \hat{\boldsymbol{\beta}}^{\text{ridge}} = \underbrace{\mathbf{X} (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}_p)^{-1} \mathbf{X}^T}_{\mathbf{S}_\lambda} \mathbf{y}$$

- When $\lambda = 0$, $\mathbf{S}_\lambda = \mathbf{H}$ and $\text{tr}(\mathbf{S}_\lambda) = p$, i.e., the degrees of freedom of the model. (Reminder: the trace of a projection matrix is equal to its rank).
- By analogy, when $\lambda > 0$, we can define the **effective degrees of freedom** as

$$\text{df}(\lambda) = \text{tr}(\mathbf{S}_\lambda).$$



Example

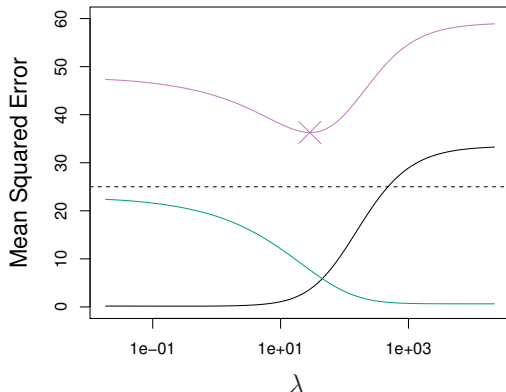


Ridge regression: scaling of predictors

- The standard least squares coefficient estimates are scale **equivariant**: multiplying X_j by a constant c simply leads to a scaling of the least squares coefficient estimates by a factor of $1/c$. In other words, regardless of how the j th predictor is scaled, $X_j\hat{\beta}_j$ will remain the same.
- In contrast, the ridge regression coefficient estimates can change substantially when multiplying a given predictor by a constant, due to the sum of squared coefficients term in the penalty part of the ridge regression objective function.
- Therefore, it is best to apply ridge regression after **standardizing the predictors** (dividing each centered variable by its standard deviation).



Why does ridge regression improve over least squares?



Simulated data with $n = 50$ observations, $p = 45$ predictors, all having nonzero coefficients. Squared bias (black), variance (green), and test MSE (purple) for the ridge regression predictions, as a function of λ . The horizontal dashed lines indicate the minimum possible MSE.



Ridge regression in R

```
library(glmnet)

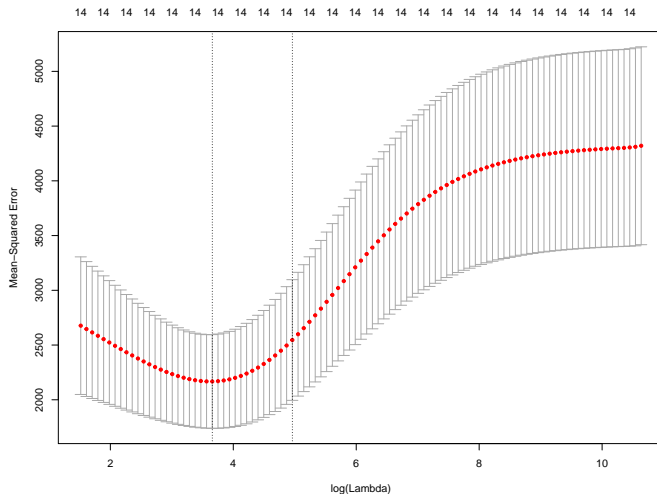
x<-model.matrix(Mortality~.-logNOx,pollution)
y<-pollution$Mortality[-21] # obs 21 has 2 missing values
n<-nrow(x)
ntrain=45
ntst=n-45
train<-sample(1:n,ntrain)
xtrain<-x[train,]
ytrain<-y[train]
xtst<-x[-train,]
ytst<-y[-train]

cv.out<-cv.glmnet(xtrain,ytrain,alpha=0)
plot(cv.out)

fit<-glmnet(xtrain,ytrain,lambda=cv.out$lambda.min,alpha=0)
ridge.pred<-predict(fit,s=cv.out$lambda.min,newx=xtst)
print(mean((ytst-ridge.pred)^2))
2421.136
```



CV error as a function of λ



Coefficients

```
fit$beta
s0
(Intercept) .
JanTemp -2.641635e-01
JulyTemp 7.231499e-01
RelHum -1.443636e-01
Rain 9.618201e-01
Education -1.154417e+01
PopDensity 2.066547e-03
pNonWhite 1.478269e+00
pWC -1.105875e+00
pop 2.629839e-06
pophouse 3.057905e+01
income -1.008305e-03
logHCPot 2.311552e+00
logNOxPot 6.616369e+00
logSO2ot 3.966114e+00
```



The lasso

- Ridge regression has one obvious disadvantage: unlike subset selection, it includes all p predictors in the final model
- The **lasso** is a relatively recent alternative to ridge regression that overcomes this disadvantage. The lasso coefficients, $\hat{\beta}^{\text{lasso}}$ minimize a penalized residual sum of squares:

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

where the L_2 norm used in ridge regression is replaced by the L_1 norm in the penalty term.

(Reminder: the L_p norm is defined as $\|\beta\|_p = \left(\sum_j |\beta_j|^p \right)^{1/p}$).

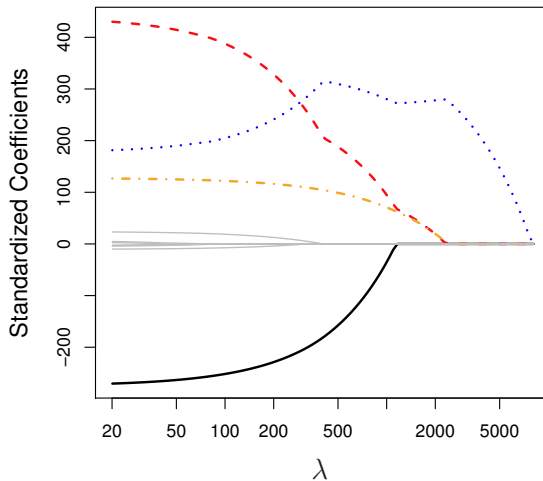


The lasso (continued)

- As with ridge regression, the lasso **shrinks the coefficient estimates towards zero**.
- However, in the case of the lasso, the L_1 penalty has the effect of forcing some of the coefficient estimates to be **exactly equal to zero** when the tuning parameter λ is sufficiently large.
- Hence, much like best subset selection, the lasso performs **variable selection**.
- We say that the lasso yields **sparse models** – that is, models that involve only a subset of the variables.
- As in ridge regression, selecting a good value of λ for the lasso is critical; cross-validation is again the method of choice.



Example



Equivalent form

- As in the case of ridge regression, the previous unconstrained optimization problem is equivalent to the following constrained one:

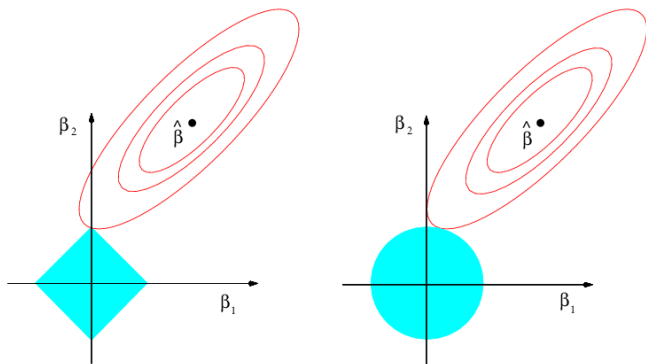
$$\hat{\beta}^{\text{lasso}} = \arg \min_{\beta} \left\{ \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 \right\}$$

subject to $\sum_{j=1}^p |\beta_j| \leq t,$

- This problem can be solved using a quadratic programming algorithm.
- Remark: this time, **the solution $\hat{\beta}^{\text{lasso}}$ is a nonlinear function of y .** There is no obvious notion of effective degrees of freedom.



Why does the lasso eliminate variables?

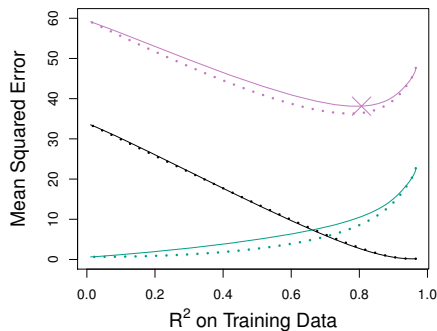
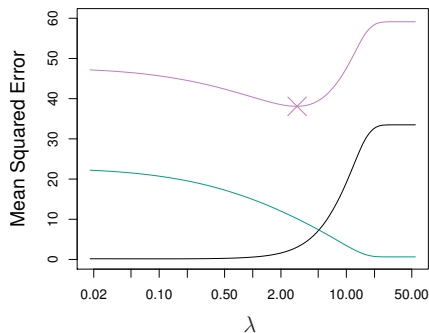


When $p = 2$, the feasibility region is a diamond, which has corners; if the solution occurs at a corner, then it has one parameter β_j equal to zero.

When $p > 2$, the feasibility region has many corners, flat edges and faces; there are many more opportunities for the estimated parameters to be zero.



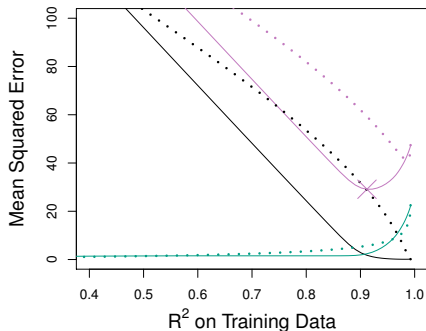
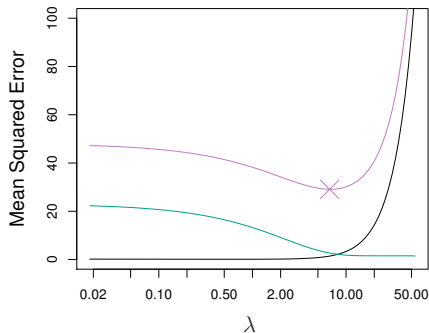
Comparing the lasso and ridge regression



Left: Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso on simulated data set of Slide 61. Right: Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed). Both are plotted against their R^2 on the training data, as a common form of indexing.



Comparing the lasso and ridge regression (continued)



Left: Plots of squared bias (black), variance (green), and test MSE (purple) for the lasso. The simulated data are similar to those in the previous slide, except that now **only two predictors are related to the response**. Right: Comparison of squared bias, variance and test MSE between lasso (solid) and ridge (dashed). Both are plotted against their R^2 on the training data, as a common form of indexing.



The lasso in R

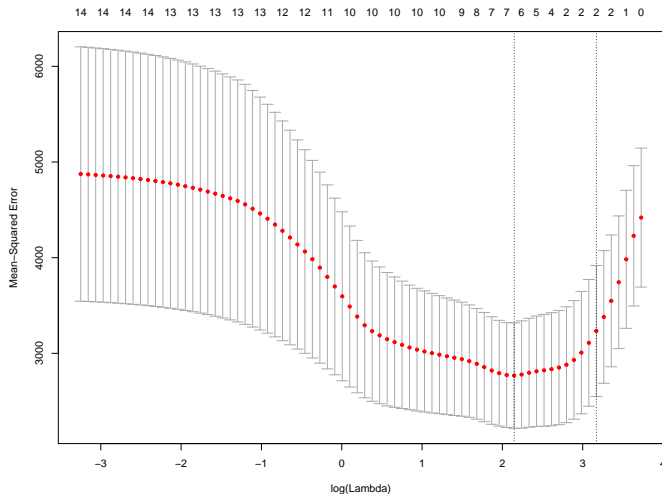
```
cv.out<-cv.glmnet(xtrain,ytrain,alpha=1)
plot(cv.out)

fit.lasso<-glmnet(xtrain,ytrain,lambda=cv.out$lambda.min,alpha=1)

lasso.pred<-predict(fit.lasso,s=cv.out$lambda.min,newx=xtst)
print(mean((ytst-lasso.pred)^2))
1946.667
```



CV error as a function of λ (lasso)



Coefficients

```
> print(fit.lasso$beta)
s0
(Intercept) .
JanTemp -1.157095e+00
JulyTemp .
RelHum .
Rain 1.404239e+00
Education -1.796084e+01
PopDensity .
pNonWhite 2.880287e+00
pWC -9.421496e-01
pop 2.141275e-06
pophouse .
income -4.655832e-04
logHCPot .
logNOxPot 1.392387e+01
logSO2ot 3.461564e-01
```



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Bayesian inference

- In **Bayesian inference**, the parameter β is treated as a random variable.
- Inference consists in computing the conditional probability distribution of the parameter given the data, obtained by the Bayes Theorem as

$$p(\beta \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \beta)p(\beta)}{p(\mathbf{y})} \propto \underbrace{p(\mathbf{y} \mid \beta)}_{\text{likelihood}} \underbrace{p(\beta)}_{\text{prior}}$$

- The marginal distribution $p(\beta)$ is called the **prior distribution** of β . It encodes prior knowledge about β , i.e., information that we have about β before observing the data.



Ridge regression corresponds to Gaussian errors and prior

- Assumptions:

① **Gaussian errors:** $\mathbf{Y} = \mathbf{X}\beta + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_n)$, so

$$p(\mathbf{y} \mid \beta) \propto \exp \left\{ -\frac{1}{2\sigma^2} \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij}\beta_j \right)^2 \right\}$$

② **Gaussian prior:** $p(\beta) \propto \exp \left(-\frac{1}{2\sigma_0^2} \sum_{j=1}^p \beta_j^2 \right)$,

- Then the log-posterior density of β is

$$\log p(\beta \mid \mathbf{y}) = -\frac{1}{2\sigma^2} \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij}\beta_j \right)^2 - \frac{1}{2\sigma_0^2} \sum_{j=1}^p \beta_j^2 + c$$

- Ridge regression searches for the **mode of the posterior distribution**.



Lasso correspond to a Laplace prior

- With the Gaussian error model as before and an independent **Laplace prior**

$$p(\beta) \propto \exp \left(-\frac{1}{\tau} \sum_{j=1}^p |\beta_j| \right),$$

we get

$$\log p(\beta \mid \mathbf{y}) = -\frac{1}{2\sigma^2} \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij}\beta_j \right)^2 - \frac{1}{\tau} \sum_{j=1}^p |\beta_j| + c$$

- Thus, the lasso corresponds to a Laplace prior.



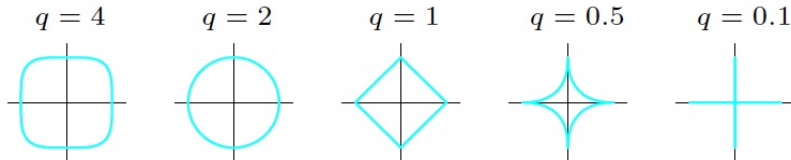
Generalization

- More generally, a prior of the form

$$p(\beta) \propto \exp \left(-\gamma \sum_{j=1}^p |\beta_j|^q \right)$$

leads to minimizing the MSE under a constraint $\sum_{j=1}^p |\beta_j|^q \leq t$.

- The case $q = 1$ (lasso) is the smallest q such that the constraint region is convex; nonconvex constraint regions make the optimization problem more difficult.
- The case $q = 0$ corresponds to subset selection.



Finding a compromise between lasso and ridge regression

- We might try using other values of q besides 0, 1, or 2. Values of $q \in (1, 2)$ suggest a **compromise between the lasso and ridge regression**.
- Although this is the case, with $q > 1$, $|\beta_j|^q$ is differentiable at 0, and so does not share the ability of lasso ($q = 1$) for setting coefficients exactly to zero.
- The **elastic net** penalty

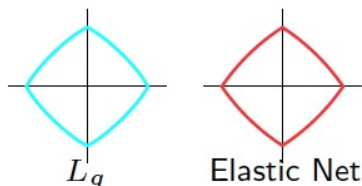
$$\lambda \sum_{j=1}^p (\alpha \beta_j^2 + (1 - \alpha) |\beta_j|)$$

with $0 \leq \alpha \leq 1$ is a different compromise between ridge and lasso.

- The elastic-net selects variables like the lasso, and shrinks together the coefficients of correlated predictors like ridge.



Elastic net penalty



Contours of constant value of $\sum_{j=1}^p |\beta_j|^q$ for $q = 1.2$ (left plot), and the elastic-net penalty $\sum_{j=1}^p (\alpha \beta_j^2 + (1 - \alpha) |\beta_j|)$ for $\alpha = 0.2$ (right plot). Although visually very similar, the elastic-net has sharp (non-differentiable) corners, while the $q = 1.2$ penalty does not.



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Principle

- Given p variables (features) $X = (X_1, \dots, X_p)$, **feature extraction** consists in finding q new features

$$Z_1 = \Phi_1(X_1, \dots, X_p)$$

$$\vdots$$

$$Z_q = \Phi_q(X_1, \dots, X_p),$$

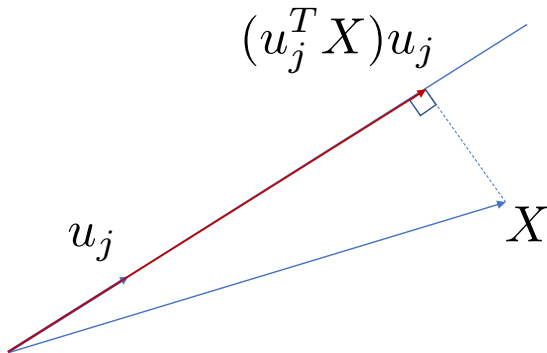
where Φ_1, \dots, Φ_q are functions from \mathbb{R}^p to \mathbb{R} . These functions may be **linear** or **nonlinear**.

- When functions Φ_j are linear, we can write $Z_j = u_j^T X$, where $u_j \in \mathbb{R}^p$ and, without loss of generality, $\|u_j\| = 1$. Only this case will be considered in this chapter.



Geometrical representation

Geometrically, Z_j can be seen as the coordinate of the **projection** of X onto an axis directed by u_j .



Objectives of feature extraction

- Feature extraction is useful for
 - Representing high-dimensional data in a lower-dimensional feature space
 - Reducing the input dimension (and hence the number of parameters) in prediction (regression or classification) problems
- Vectors u_1, \dots, u_q are determined in such a way that the new features Z_1, \dots, Z_q contain **as much useful information as possible**.
- Feature extraction methods can be supervised, or unsupervised.
- Here, we consider an unsupervised method: Principal Component Analysis (PCA)



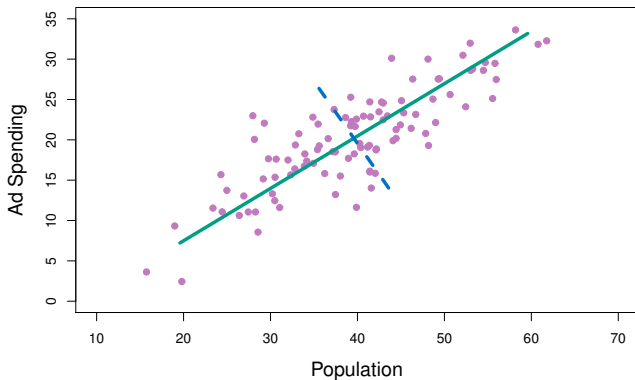
Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Basic idea

Idea: find **orthogonal directions** u_j in input space in which the projected data has **maximal variance**. These directions correspond to features $Z_j = u_j^T X$ (called principal components) that have maximum variance.



Finding the first component

- Let $X = (X_1, \dots, X_p)$ be a random vector with variance matrix Σ .
- The first feature $Z_1 = u_1^T X$, called the **first component**, is chosen such that

$$\text{Var}(Z_1) = \max_{u_1} \text{Var}(u_1^T X) = \max_{u_1} u_1^T \Sigma u_1$$

subject to $u_1^T u_1 = 1$.

- To solve this constrained optimization problem, we write the Lagrange function as

$$L(u_1, \lambda) = u_1^T \Sigma u_1 - \lambda(u_1^T u_1 - 1)$$



Finding the first component (continued)

- The solution must verify

$$\frac{\partial L}{\partial u_1} = 2\mathbf{\Sigma}u_1 - 2\lambda u_1 = 0 \Leftrightarrow \mathbf{\Sigma}u_1 = \lambda u_1$$
$$u_1^T u_1 = 1$$

- Vector u_1 is thus the **eigenvector** of $\mathbf{\Sigma}$ with unit norm and eigenvalue λ_1 . (We recall that a symmetric and positive definite $p \times p$ matrix has p orthogonal eigenvectors with real and positive eigenvalues).

- Now,

$$\text{Var}(u_1^T X) = u_1^T \mathbf{\Sigma} u_1 = u_1^T (\lambda_1 u_1) = \lambda_1 u_1^T u_1 = \lambda_1.$$

so λ_1 must be the **largest eigenvalue**.

- Vector u_1 is called the **loading vector** of the first principal component.



Finding the second component

- The **second component** $Z_2 = u_2^T X$ is chosen such that

$$\text{Var}(Z_2) = \max_{u_2} \text{Var}(u_2^T X) = \max_{u_2} u_2^T \Sigma u_2$$

subject to $u_2^T u_2 = 1$ and $\text{Cov}(Z_1, Z_2) = 0$.

- Now,

$$\text{Cov}(Z_1, Z_2) = \text{Cov}(u_1^T X, u_2^T X) = \underbrace{u_1^T \Sigma}_{(\lambda_1 u_1)^T} u_2 = \lambda_1 u_1^T u_2,$$

so the second constraint can be written $u_1^T u_2 = 0$.

- The Lagrange function is

$$L(u_2, \lambda, \mu) = u_2^T \Sigma u_2 - \lambda(u_2^T u_2 - 1) - \mu u_2^T u_1$$



Finding the second component (continued)

- We solve:

$$\frac{\partial L}{\partial u_2} = 2\mathbf{\Sigma}u_2 - 2\lambda u_2 - \mu u_1 = 0 \quad (1)$$

- Left-multiplying (1) by u_1^T , we get

$$\underbrace{2 u_1^T \mathbf{\Sigma} u_2}_0 - \underbrace{2 u_1^T \lambda u_2}_0 - \mu u_1^T u_1 = 0 \Rightarrow \mu = 0$$

- So, (1) reduces to

$$\frac{\partial L}{\partial u_2} = 2\mathbf{\Sigma}u_2 - 2\lambda u_2 = 0 \Leftrightarrow \mathbf{\Sigma}u_2 = \lambda u_2$$

- The solution is an eigenvector of $\mathbf{\Sigma}$. We choose the one with the second largest eigenvalue λ_2 .



Finding the next components

- Continuing the same line of reasoning, we obtain p **uncorrelated components** $Z = (Z_1, \dots, Z_p)$ corresponding to the eigenvalues $\lambda_1 \geq \dots \geq \lambda_p$ of Σ .
- We can write

$$Z = \mathbf{U}^T X,$$

where $\mathbf{U} = (u_1, \dots, u_p)$ is the $p \times p$ matrix (called the **loading matrix**) whose columns are the p eigenvectors.



Properties

- The variance matrix of Z is

$$\text{Var}(Z) = \mathbf{U}^T \mathbf{\Sigma} \mathbf{U} = \mathbf{\Lambda},$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_p)$ is the diagonal matrix containing the p eigenvalues of $\mathbf{\Sigma}$.

- Matrix \mathbf{U} verifies $\mathbf{U}^T \mathbf{U} = \mathbf{I}$, i.e., $\mathbf{U}^{-1} = \mathbf{U}^T$: it is an **orthogonal matrix**, corresponding to a **rotation**.



Properties (continued)

- Consequently,

$$\text{tr}(\mathbf{\Lambda}) = \text{tr}[\mathbf{U}^T(\mathbf{\Sigma U})] = \text{tr}[(\mathbf{\Sigma U})\mathbf{U}^T] = \text{tr}(\mathbf{\Sigma})$$

- Hence, the sum of the eigenvalues is the **total variance**

$$\sum_{j=1}^p \lambda_j = \sum_{j=1}^p \text{Var}(X_j)$$

- The **proportion of the variance explained by the first q components** is

$$\sum_{j=1}^q \lambda_j \bigg/ \sum_{j=1}^p \lambda_j$$



Practical application

- In practice, we center the data, and we estimate Σ by the empirical variance matrix

$$\hat{\Sigma} = \frac{1}{n} \sum_{i=1}^n x_i x_i^T$$

- If we also standardize the data, then matrix $\hat{\Sigma}$ is actually the **correlation matrix** R (its diagonal elements equal 1, and its off-diagonal elements are correlation coefficients).
- Typically, we keep only q components Z_1, \dots, Z_q such that the cumulative proportion of explained variance is close enough to 1.



Example: Wine data

- Results of a chemical analysis of 178 wines grown in the same region in Italy but derived from three different cultivars.
- The analysis determined the quantities of 13 constituents found in each of the three types of wines.



PCA in R

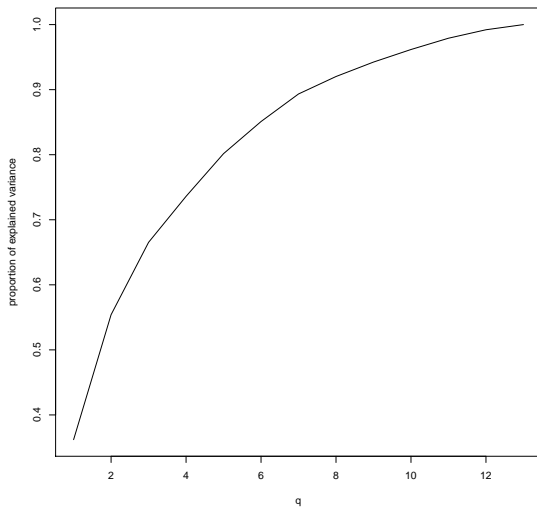
```
wine<-read.csv('wine.data',header=FALSE)
X<-wine[,2:14]
X<-scale(X)
pca<-princomp(X)
Z<-pca$scores
lambda<-pca$sdev^2

plot(cumsum(lambda)/sum(lambda),type="l",xlab="q",
     ylab="proportion of explained variance")

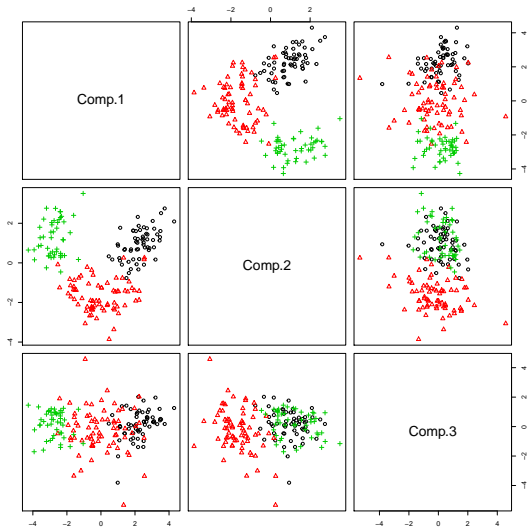
pairs(Z[,1:3],col=wine[,1],pch=wine[,1])
```



Proportion of explained variance



First 3 principal components



Overview

- 1 Subset selection methods
 - Best subset selection
 - Stepwise selection
- 2 Choosing the optimal model
 - Training error adjustment
 - Direct error estimation
- 3 Ridge regression and lasso
 - Ridge regression and Lasso
 - Bayesian interpretation
- 4 Feature extraction
 - Principal component analysis
 - Principal component regression



Principal component regression (PCR)

- The idea is to fit a regression model using least squares, taking as predictors $q < p$ principal components:

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

- We have

$$\sum_{m=1}^q \theta_m z_{im} = \sum_{m=1}^q \theta_m \sum_{j=1}^p u_{mj} x_{ij} = \sum_{j=1}^p \underbrace{\sum_{m=1}^q \theta_m u_{mj}}_{\beta_j} x_{ij}$$



Principal component regression (continued)

- Hence, the PCR model can be thought of as a special case of the original linear regression model.
- Dimension reduction serves to constrain the estimated β_j coefficients, which can yield a **good bias-variance tradeoff**.
- As with ridge regression, principal components depend on the scaling of the inputs, so typically we first standardize them.
- The value of q can be determined by cross-validation.



Principal component regression in R

```
library(pls)

pcr.fit<-pcr(Mortality ~.-logNOx,data=pollution,scale=TRUE,
             validation="CV")

summary(pcr.fit)
validationplot(pcr.fit,val.type = "MSEP",
               legendpos = "topright")
```



Cross-validation MSE as a function of M

