6.3 Dimesion Reduction Methods

Dimension Reduction?

- So far, we've thought about controlling variance by
  - using a subset of the original input variables
  - shrinking coefficients toward zero
- These methods were all defined using their original predictors X
- We now explore a class of approaches that transform the predictors and then fit a least squares model using the transformed variables. We will refer to these techniques as dimension reduction methods.

Dimension Reduction

- Let $Z_1, Z_2, \ldots, Z_M$ represent $M < p$ linear combinations of the original p predictors:

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$$

  for some constants $\phi_{m1}, \ldots, \phi_{mp}$.
- We then fit the linear regression model

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i$$

  ($i \in \{1, \ldots, n\}$) using ordinary least squares.
- If the constants $\phi_{m1}, \ldots, \phi_{mp}$ are chosen wisely, dimension reduction can often outperform OLS regression.

Dimension Reduction

Based on $Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$, we can write

$$\sum_{m=1}^{M} \theta_m z_{im} = \sum_{m=1}^{M} \theta_m \sum_{j=1}^{p} \phi_{mj} x_{ij} = \sum_{j=1}^{p} \sum_{m=1}^{M} \theta_m \phi_{mj} x_{ij} = \sum_{j=1}^{p} \beta_j x_{ij}$$

where $\beta_J = \sum_{m=1}^{M} \theta_m \phi_{mj}$

Dimension Reduction

So the model

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i$$

can be thought of as a special case of the original linear regression model.

- Dimension reduction serves to constrain the estimated β coefficients, since they must now take this form.
- Can win in the bias-variance trade off.
- We will consider two approaches: principal components and partial least squares.

Principal Components Regression

- Principal components analysis (PCA) is a way to derive a low-dimensional set of features from a large set of variables.
  - We will fill out some of the details when we get to Chapter 12 (Unsupervised Learning).
  - Here, we consider PCA as a dimension reduction technique.

An Overview of PCA

- This is a technique for reducing the dimension of an $n \times p$ matrix X
- The first principal component is the (normalized) linear combination of the variables with the largest variance.
  - (We want to capture as much of the variability as possible.)

The First Principal Component

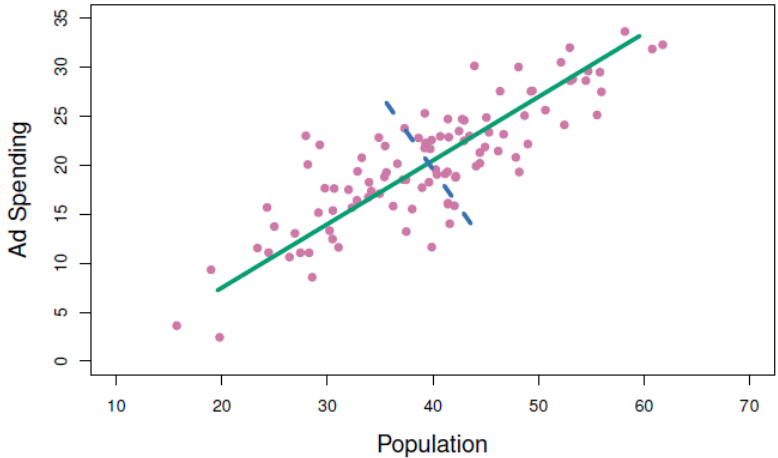


FIGURE 6.14. *The population size (*pop*) and ad spending (*ad*) for* 100 *different cities are shown as purple circles. The green solid line indicates the first principal component, and the blue dashed line indicates the second principal component.*

The First Principal Component

This first principal component can be summarized mathematically using

$$Z_1 = 0.839 \times (\text{pop} - \overline{\text{pop}}) + 0.544 \times (\text{add} - \overline{\text{add}})$$

where ϕ_{11} and ϕ_{21} are the principal component loadings, which define the direction referred to above.

- Idea: for every possible linear combination of pop and ad where $\phi_{11} + \phi_{12} = 1$
- We constrain this sum to 1 to avoid arbitrarily increasing ϕ_{11} and ϕ_{21} in order to blow up the variance.

The First Principal Component

- The first principal component looks a lot like a linear regression line, but its derivation is different!
  - It also does *not* use the outcome variable y in its construction, instead examining only the predictor space.
  - (Recall for the `credit` data, the outcome is `balance`.)
- It defines the line that minimizes the sum of squared *perpendicular distances* between each point and the line.
  - So we end up with a line that is overall as close as possible to the data.

The First Principal Component

- This is a length n vector Z_1, the values of which can be thought of as single-number summaries of the joint `pop` and `ad` for each location.
  - For example, if $z_{i1} < 0$, this is a city with below average population size and ad spending.
- How well does this work? If there's an approximately linear relationship, it tends to work pretty well.

The First Principal Component

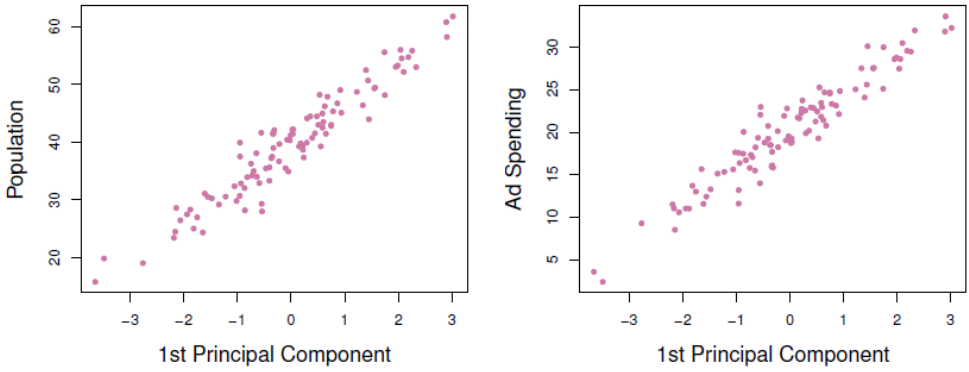


FIGURE 6.16. *Plots of the first principal component scores z_{i1} versus* pop *and* ad. *The relationships are strong.*

Further Principal Components

- We can construct up to p distinct principal components.
- The second Z_2 is the linear combination of the variables with the largest variance, under the constraint that it is uncorrelated with Z_1.
  - This turns out to be equivalent to requiring that Z_2 be orthogonal to Z_1.
- Each successive principal component maximizes the variability, such that it is uncorrelated with all preceding components.

$$Z_2 = 0.544 \times (\text{pop} - \overline{\text{pop}}) + 0.839 \times (\text{add} - \overline{\text{add}})$$

Further Principal Components

- In the advertising data, there are only two predictors, so the first two principal components contain all of the information in `pop` and `ad`.
- By construction, the first principal component contains the most information.

Further Principal Components

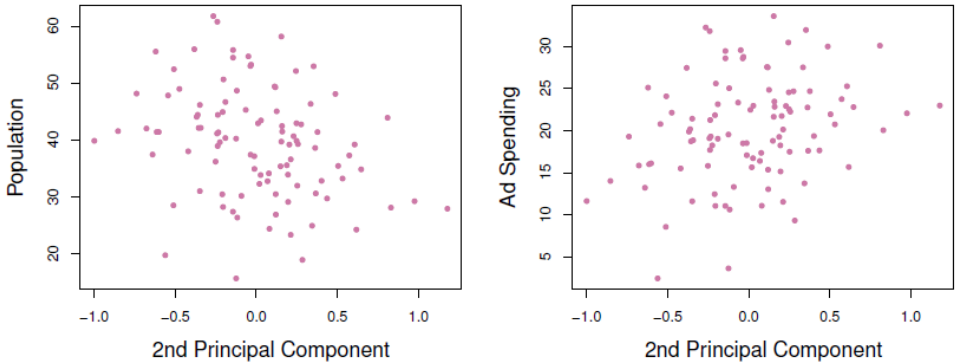


FIGURE 6.17. *Plots of the second principal component scores z_{i2} versus* pop *and* ad*. The relationships are weak.*

Principal Components Regression

PCR involves constructing the first M principal components and then using these in a linear regression model fit using least squares.

Idea: often a small number of principal components are sufficient to explain most of the variability in the data, as well as the relationship with the response.

- We assume the directions in which X show the most variation are the directions associated with Y.
  - Not guaranteed to be true, but turns out to be reasonable enough to work well.
- As long as $M << p$, this can be used to help mitigate overfitting.
- Tends to work best when relatively few principal components caption most of the variation in the predictors.

Principal Components Regression

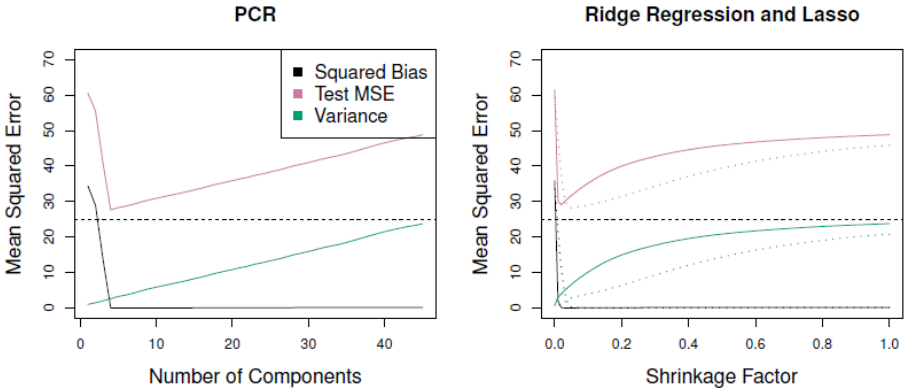


FIGURE 6.19. PCR, ridge, and lasso applied to simulated data in which the first five principal components of X contain all the information about the response y. The irreducible error $Var(\epsilon)$ is shown as a horizontal dashed line. Left: PCR. Right: Lasso (solid) and ridge (dotted). The x-axis displays the shrinkage factor of the coef estimates (l_2 norm of shrunken coef ests / l_2 norm of least squares ests).

Principal Components Regression

A few notes

- This is not a feature selection method!
  - All p original features are used in constructing each principal component.
- M is typically chosen by cross-validation.
- We typically standardize each predictor prior to generating the principal components.
  - This prevents variables with high variance (on their original scale) from dominating.

PCR: Example

```
library(pls)
data("Credit")
set.seed(2)
pcr.fit <- pcr(Balance ~ ., data = Credit,
               scale = TRUE,
               validation = "CV")
summary(pcr.fit)
```

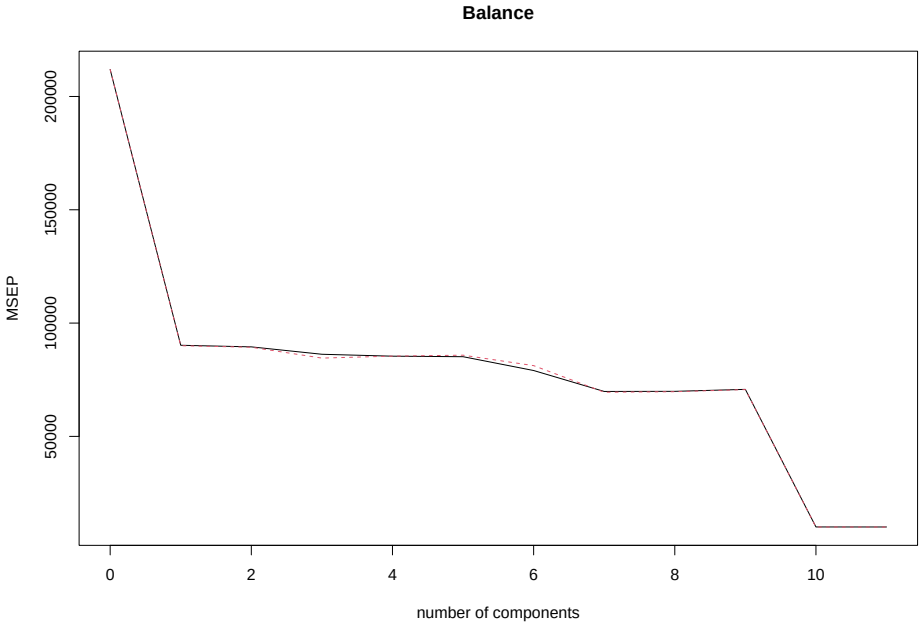
```
validationplot(pcr.fit, val.type = "MSEP")
```

PCR: Example

```
Data:   X dimension: 400 11
        Y dimension: 400 1

Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7
CV           460.3    300.3    299.2    293.7    292.2    291.8    281.2
adjCV        460.3    300.0    298.9    290.8    292.3    293.0    285.0

TRAINING: % variance explained
         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 c
X          25.05    39.64    49.73    59.74    68.89    77.73    86.43    9
Balance    58.07    58.37    60.78    60.90    61.46    63.11    68.70    6
```


Balance
MSEP
200000
150000
100000
50000
0
2
4
6
8
10
number of components

Partial Least Squares

- PCR identifies linear combinations, or *directions*, that best represent the predictors X.
- These are identified in an *unsupervised* way, since the response Y is not used in their construction.
- Drawback: in PCR, there is no guarantee that the directions that best explain the predictors will also best predict the response.

Partial Least Squares

- Like PCR, PLS is a dimension reduction method that identifies a set of M new features Z which are used to fit a linear model.
- Unlike PCR, PLS identifies these in a *supervised* way, using y to identify features which are also related to the response.
- The idea is to find directions that help explain both the response and the predictors.

Partial Least Squares

The dimension reduction process is essentially the same.

(The following copied exactly from a previous slide:)

- Let $Z_1, Z_2, \ldots, Z_M$ represent $M < p$ linear combinations of the original p predictors:

$$Z_m = \sum_{j=1}^{p} \phi_{mj} X_j$$

  for some constants $\phi_{m1}, \ldots, \phi_{mp}$.
- We then fit the linear regression model

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i$$

  ($i \in \{1, \ldots, n\}$) using ordinary least squares.

Partial Least Squares

- PLS computes the first direction Z_1 by setting each ϕ_{1j} equal to the coefficient from the simple linear regression of Y onto the (standardized) X_j.
- It can be shown that this cofficient is proportional to the correlation between y and X_j.
- So in computing $Z_1 = \sum_{j=1}^{p} \phi_{1j} X_j$, PLS places the highest weight on the variables that are most strongly related to the response.

Partial Least Squares

- Subsequent directions are found by taking the residuals and then repeating this process on them.
- We can think of the residuals as the leftover information that went unexplained by the first PLS direction.
- This process can be repeated M times to identify all PLS components.
  - M again found using cross-validation.
- Finally, we use least squares to fit a linear model to predict y using Z.

PLS: Example

```
set.seed(2)
pls.fit <- plsr(Balance ~ ., data = Credit,
                scale = TRUE,
                validation = "CV")
summary(pls.fit)
```

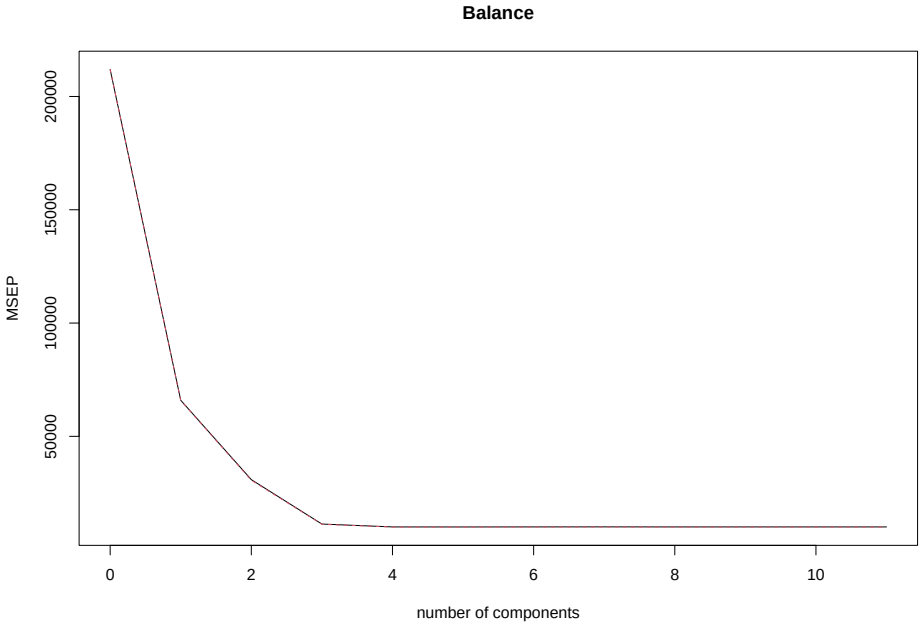
```
validationplot(pls.fit, val.type = "MSEP")
```

PLS: Example

```
Data:   X dimension: 400 11
          Y dimension: 400 1

Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           460.3    256.8    175.8    106.3   100.13   100.08    100.2
adjCV        460.3    256.5    174.9    105.7    99.96    99.94    100.0

TRAINING: % variance explained
         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8
X          24.58    32.53    37.84    50.55    60.80    65.92    73.20
Balance    69.67    86.53    94.95    95.46    95.48    95.48    95.48
```


Balance
MSEP
200000
150000
100000
50000
0
2
4
6
8
10
number of components

PLS: Example

```
pls.fit <- plsr(Balance ~ ., data = Credit,
                scale = TRUE,
                ncomp = 3)
summary(pls.fit)
```

```
## Data:    X dimension: 400 11
##  Y dimension: 400 1
## Fit method: kernelpls
## Number of components considered: 3
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps
## X          24.58    32.53    37.84
## Balance    69.67    86.53    94.95
```

PCR and PLS

- Models can be difficult to interpret as they do not perform any kind of variable selection or directly produce coefficient estimates.
- Supervised dimension reduction of PLS can reduce bias, but may increase variance.
- Overall benefit of PLS vs PCR is a wash.