Dimension Reduction Methods

And Bayesian Machine Learning

Marek Petrik

2/28

Previously in Machine Learning

- How to choose the right features if we have (too) many options

- Methods:
  1. Subset selection
  2. Regularization (shrinkage)
  3. Dimensionality reduction (next class)

Best Subset Selection

- Want to find a subset of p features
- The subset should be small and predict well
- Example: credit $\sim$ rating $+$ income $+$ student $+$ limit

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
  Fit all $\binom{p}{k}$ models that contain k features ;
  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Algorithm 1: Best Subset Selection

Achieving Scalability

- Complexity of *Best Subset Selection*?

Achieving Scalability

- Complexity of *Best Subset Selection*?
- Examine all possible subsets? How many?

Achieving Scalability

- Complexity of *Best Subset Selection*?
- Examine all possible subsets? How many?
- $O(2^p)$!

Achieving Scalability

- Complexity of *Best Subset Selection*?
- Examine all possible subsets? How many?
- $O(2^p)$!

- Heuristic approaches:
  1. **Stepwise selection**: Solve the problem approximately: greedy
  2. **Regularization**: Solve a different (easier) problem: relaxation

Which Metric to Use?

$\mathcal{M}_0 \leftarrow$ *null model* (no features);

for $k = 1, 2, \ldots, p$ **do**

  Fit all $\binom{p}{k}$ models that contain k features ;

  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)

end

return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Algorithm 2: Best Subset Selection

Which Metric to Use?

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
  Fit all $\binom{p}{k}$ models that contain k features ;
  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Algorithm 3: Best Subset Selection

1. **Direct error estimate**: Cross validation, precise but computationally intensive

Which Metric to Use?

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
  Fit all $\binom{p}{k}$ models that contain k features ;
  $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Algorithm 4: Best Subset Selection

1. **Direct error estimate**: Cross validation, precise but computationally intensive
2. **Indirect error estimate**: Mellow's C_p:

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2) \text{ where } \hat{\sigma}^2 \approx \mathrm{Var}[\epsilon]$$

   Akaike information criterion, BIC, and many others. Theoretical foundations

Which Metric to Use?

$\mathcal{M}_0 \leftarrow$ *null model* (no features);
for $k = 1, 2, \ldots, p$ **do**
    Fit all $\binom{p}{k}$ models that contain k features ;
    $\mathcal{M}_k \leftarrow$ best of $\binom{p}{k}$ models according to a metric (CV error, R^2, etc)
end
return Best of $\mathcal{M}_0, \mathcal{M}_1, \ldots, \mathcal{M}_p$ according to metric above

Algorithm 5: Best Subset Selection

1. **Direct error estimate**: Cross validation, precise but computationally intensive
2. **Indirect error estimate**: Mellow's C_p:

$$C_p = \frac{1}{n}(\mathrm{RSS} + 2d\hat{\sigma}^2) \text{ where } \hat{\sigma}^2 \approx \mathrm{Var}[\epsilon]$$

   Akaike information criterion, BIC, and many others. Theoretical foundations
3. **Interpretability Penalty**: What is the cost of extra features

Regularization

1. **Stepwise selection**: Solve the problem approximately

2. **Regularization**: Solve a different (easier) problem: <u>relaxation</u>
   - Solve a machine learning problem, but penalize solutions that use “too much” of the features

Regularization

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j \beta_j^2 =$$

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

- **Lasso** (parameter λ), ℓ_1 penalty

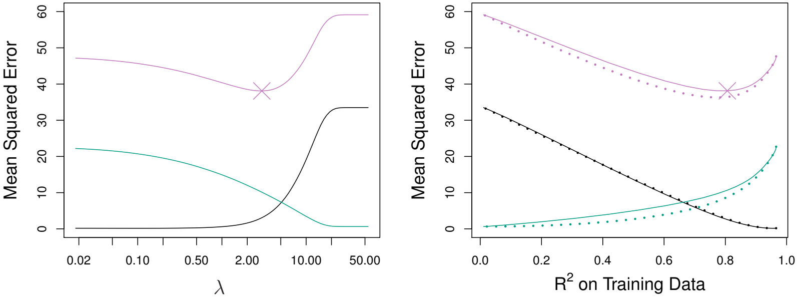
$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j |\beta_j| =$$

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

- Approximations to the ℓ_0 solution

Why Lasso Works

- Bias-variance trade-off
- Increasing λ increases bias
- Example: all features relevant

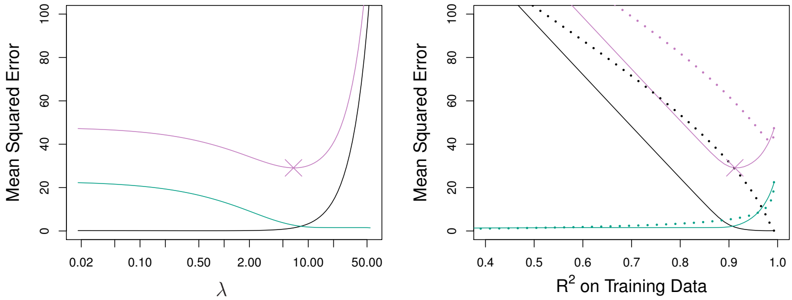


purple: test MSE, black: bias, green: variance

dotted (ridge)

Why Lasso Works

- Bias-variance trade-off
- Increasing λ increases bias
- Example: some features relevant



purple: test MSE, black: bias, green: variance
dotted (ridge)

Regularization

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j \beta_j^2 =$$

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

- **Lasso** (parameter λ), ℓ_1 penalty

$$\min_\beta \mathrm{RSS}(\beta) + \lambda \sum_j |\beta_j| =$$

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

- Approximations to the ℓ_0 solution

Regularization: Constrained Formulation

- **Ridge regression** (parameter λ), ℓ_2 penalty

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \text{ subject to } \sum_j \beta_j^2 \leq s$$

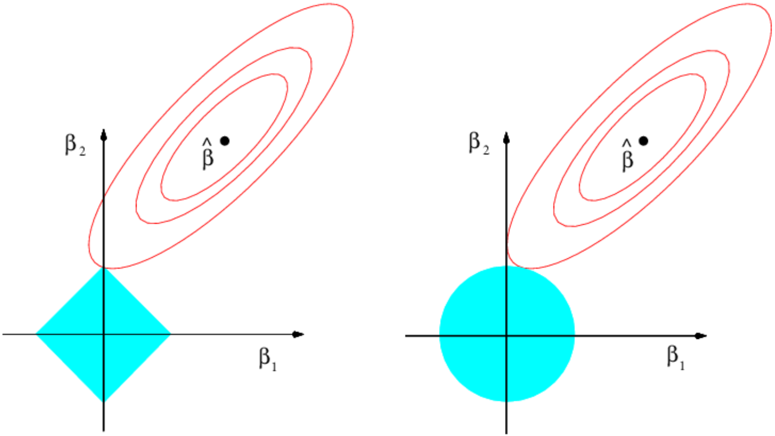
- **Lasso** (parameter λ), ℓ_1 penalty

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \text{ subject to } \sum_j |\beta_j| \leq s$$

- Approximations to the ℓ_0 solution

Lasso Solutions are Sparse

Constrained Lasso (left) vs Constrained Ridge Regression (right)



Constraints are blue, red are contours of the objective

Today

- Dimension reduction methods
  - Principal component regression
  - Partial least squares
- Interpretation in high dimensions
- Bayesian view of ridge regression and lasso

Dimension Reduction Methods

- Different approach to model selection
- We have many features: $X_1, X_2, \ldots, X_p$
- Transform features to a *smaller* number $Z_1, \ldots, Z_M$

Dimension Reduction Methods

- Different approach to model selection
- We have many features: $X_1, X_2, \ldots, X_p$
- Transform features to a *smaller* number $Z_1, \ldots, Z_M$

- **Find** constants ϕ_{jm}
- New features Z_m are **linear combinations** of X_j:

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

Dimension Reduction Methods

- Different approach to model selection
- We have many features: $X_1, X_2, \ldots, X_p$
- Transform features to a *smaller* number $Z_1, \ldots, Z_M$

- **Find** constants ϕ_{jm}
- New features Z_m are **linear combinations** of X_j:

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

- **Dimension reduction**: M is much smaller than p

Using Transformed Features

- New features Z_m are **linear combinations** of X_j:

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

- Fit linear regression model:

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

- Run plain linear regression, logistic regression, LDA, or anything else

Recovering Coefficients for Original Coefficients

- Prediction using transformed features

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

- New features Z_m are **linear combinations** of X_j:

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

- Consider prediction for data point i

$$\sum_{m=1}^{M} \theta_m z_{im}$$

Recovering Coefficients for Original Coefficients

- Prediction using transformed features

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

- New features Z_m are **linear combinations** of X_j:

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

- Consider prediction for data point i

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

Recovering Coefficients for Original Coefficients

- Prediction using transformed features

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

- New features Z_m are **linear combinations** of X_j:

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

- Consider prediction for data point i

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

Recovering Coefficients for Original Coefficients

- Prediction using transformed features

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

- New features Z_m are **linear combinations** of X_j:

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

- Consider prediction for data point i

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

Dimension Reduction

1. Reduce dimensions of features Z from X

2. Fit prediction model to compute ϕ

3. Compute weights for the original features β

Dimension Reduction

1. Reduce dimensions of features Z from X

2. Fit prediction model to compute ϕ

3. Compute weights for the original features β

Dimension Reduction

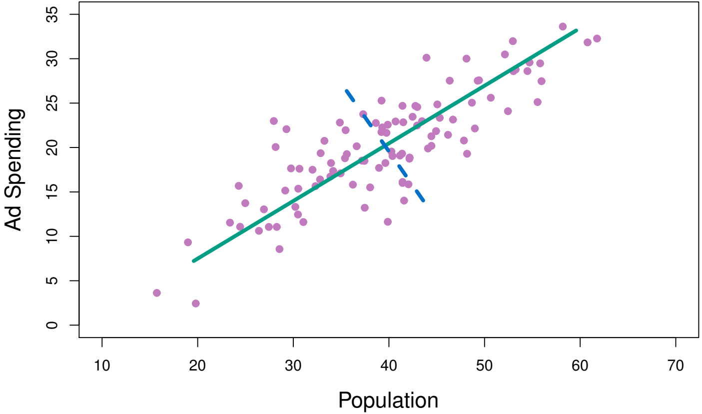
1. Reduce dimensions of features Z from X
2. Fit prediction model to compute ϕ
3. Compute weights for the original features β

How (and Why) Reduce Feaures?

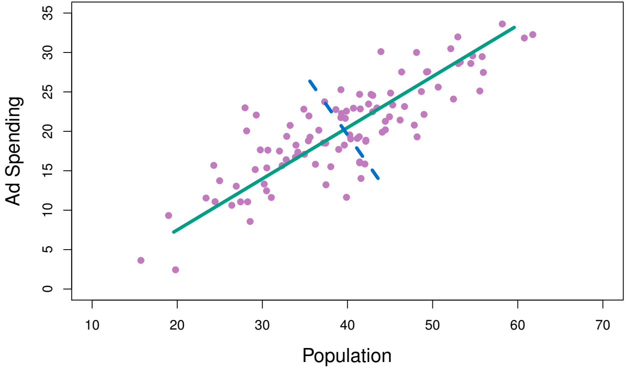
1. Principal Component Analysis (PCA)
2. Partial least squares
3. Also: many other non-linear dimensionality reduction methods

Principal Component Analysis

- **Unsupervised** dimensionality reduction methods
- Works with $n \times p$ *data matrix* $\mathbf{X}$ (no labels)
- Correlated features: pop and ad



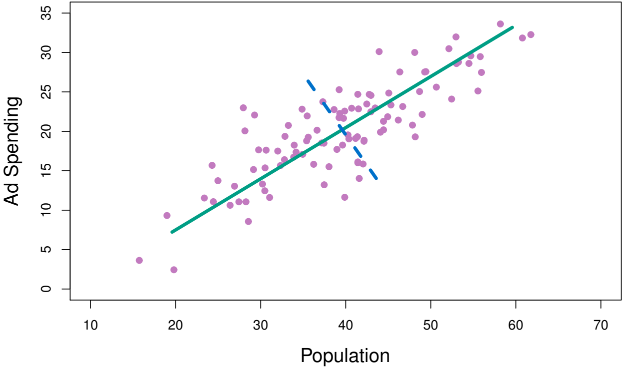
1st Principal Component



- **1st Principal Component**: Direction with the largest variance

$$Z_1 = 0.839 \times (\mathsf{pop} - \overline{\mathsf{pop}}) + 0.544 \times (\mathsf{ad} - \overline{\mathsf{ad}})$$

1st Principal Component

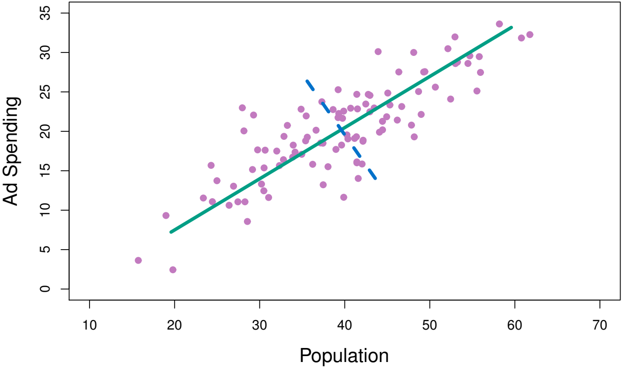


- **1st Principal Component**: Direction with the largest variance

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

- Is this linear?

1st Principal Component

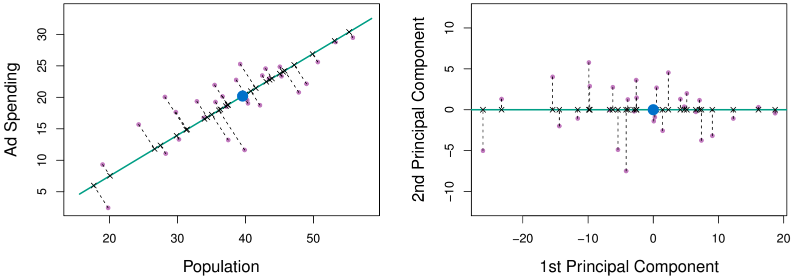


- **1st Principal Component**: Direction with the largest variance

$$Z_1 = 0.839 \times (\mathsf{pop} - \overline{\mathsf{pop}}) + 0.544 \times (\mathsf{ad} - \overline{\mathsf{ad}})$$

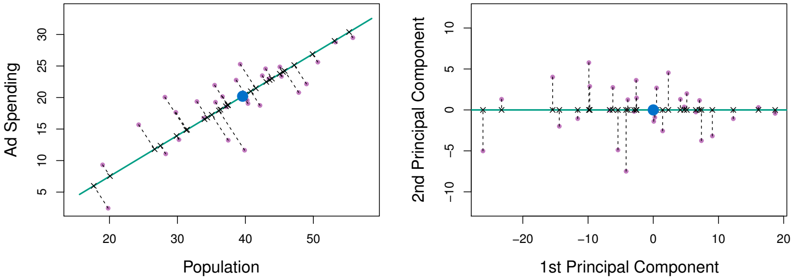
- Is this linear? Yes, after *mean centering*.

1st Principal Component



green line: 1st principal component, minimize distances to all points

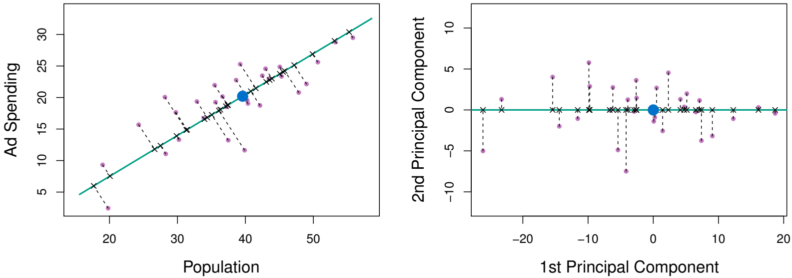
1st Principal Component



green line: 1st principal component, minimize distances to all points

Is this the same as linear regression?

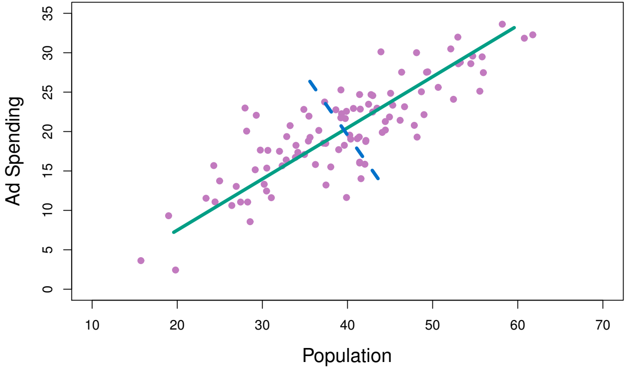
1st Principal Component



green line: 1st principal component, minimize distances to all points

Is this the same as linear regression? **No**, like *total least squares*.

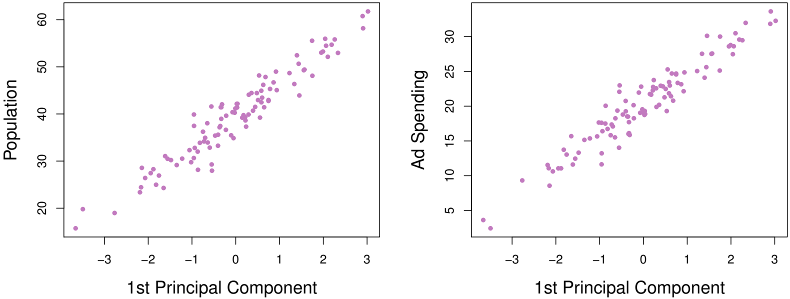
2nd Principal Component



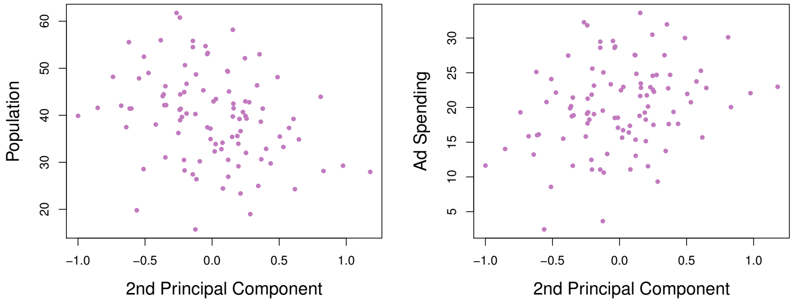
- **2nd Principal Component**: Orthogonal to 1st component, largest variance

$$Z_2 = 0.544 \times (\mathsf{pop} - \overline{\mathsf{pop}}) - 0.839 \times (\mathsf{ad} - \overline{\mathsf{ad}})$$

1st Principal Component



2nd Principal Component

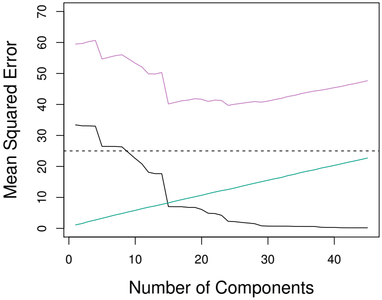


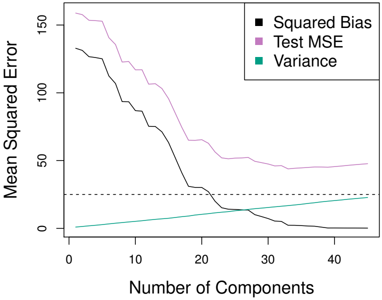
Properties of PCA

- No more principal components than features
- Principal components are perpendicular
- Principal components are eigenvalues of $\mathbf{X}^\top\mathbf{X}$
- Assumes normality, can break with heavy tails
- PCA depends on the scale of features

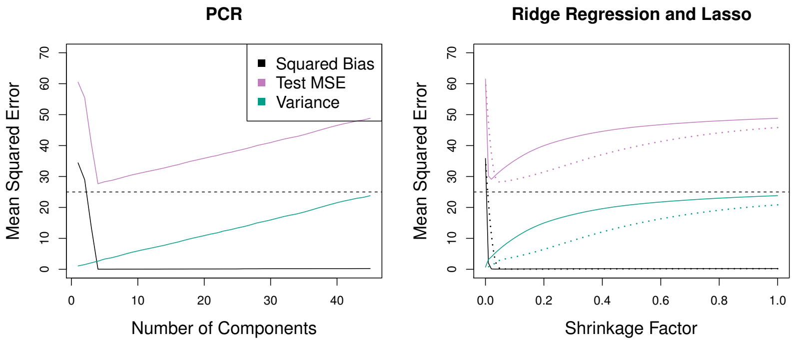
Principal Component Regression

1. Use PCA to reduce features to a small number of principal components
2. Fit regression using principal components



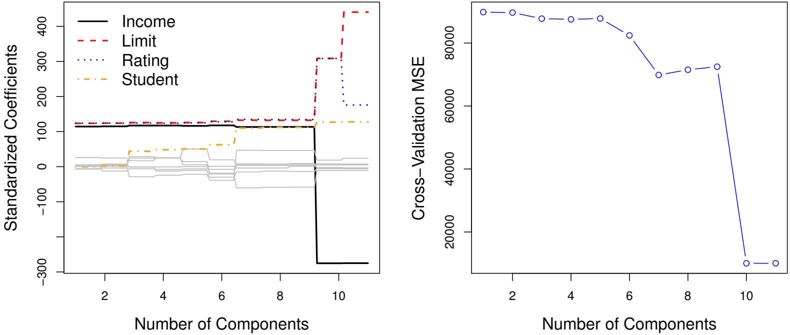


PCR vs Ridge Regression & Lasso



- PCR selects combinations of all features (not feature selection)
- PCR is closely related to ridge regression

PCR Application



Standardizing Features

- Regularization and PCR depend on scales of features
- Good practice is to *standardize* features to have **same variance**

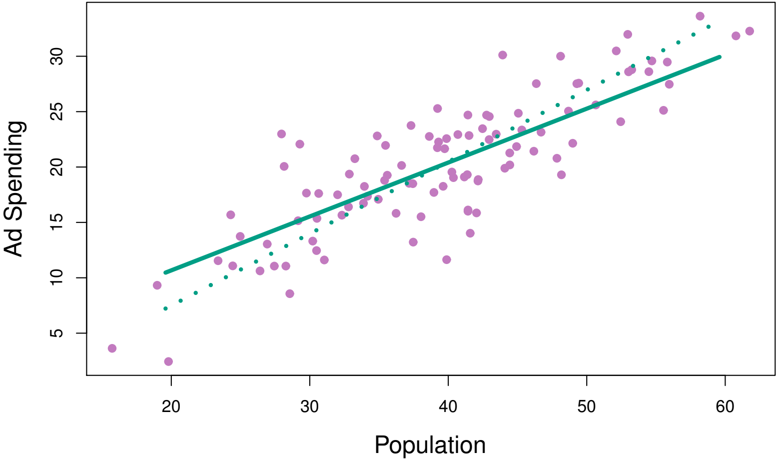
$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}}$$

- Do not standardize features when they have the same units
- PCA needs mean-centered features

$$\tilde{x}_{ij} = x_{ij} - \bar{x}_j$$

Partial Least Squares

- Supervised version of PCR



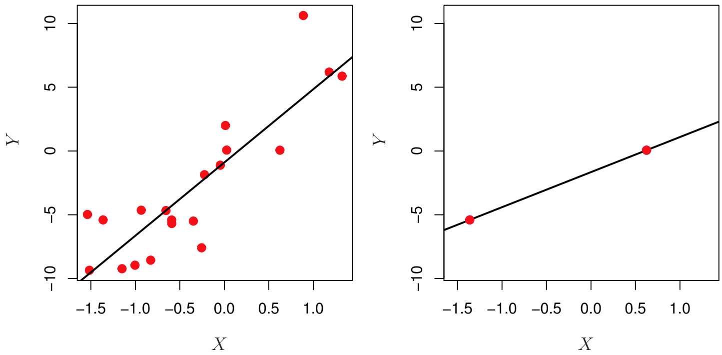
High-dimensional Data

1. Predict blood pressure from DNA: $n = 200, p = 500\,000$

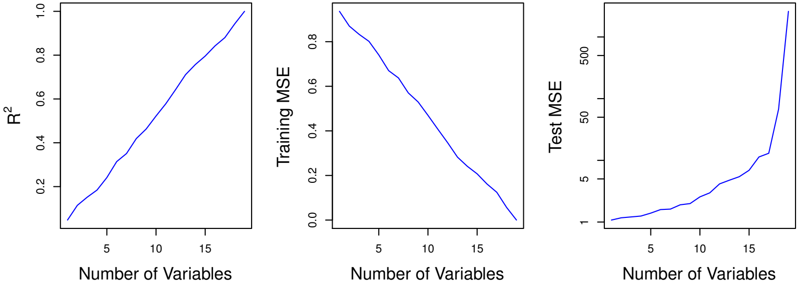
2. Predicting user behavior online: $n = 10\,000, p = 200\,000$

Problem With High Dimension

- Computational complexity
- Overfitting is a problem



Overfitting with Many Variables



Interpreting Feature Selection

1. Solutions may not be unique
2. Must be careful about how we report solutions
3. Just because one combination of features predicts well, does not mean others will not

Bayesian Machine Learning

- Maximum likelihood
- What if we have prior knowledge?
- Improve on maximum likelihood

Estimating Coefficients: Maximum Likelihood

- **Likelihood**: Probability that data is generated from a model

$$\ell(\text{model}) = \Pr[\text{data} \mid \text{model}]$$

- Find the most likely model:

$$\max_{\text{model}} \ell(\text{model}) = \max_{\text{model}} \Pr[\text{data} \mid \text{model}]$$

- Likelihood function is difficult to maximize
- Transform it using $\log$ (strictly increasing)

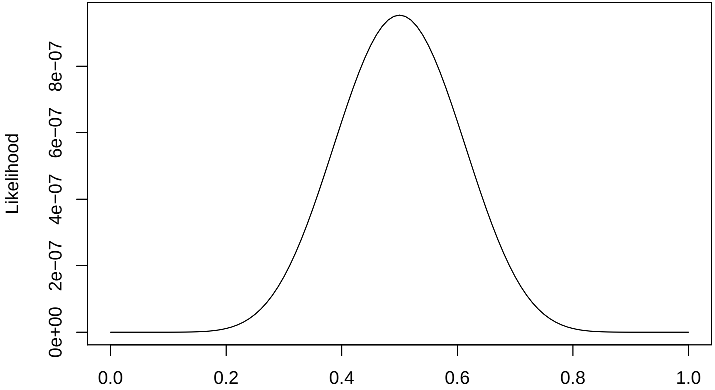
$$\max_{\text{model}} \log \ell(\text{model})$$

- Strictly increasing transformation does not change maximum

Example: Maximum Likelihood

- Assume a coin with p as the probability of *heads*
- **Data**: h heads, t tails
- The likelihood function is:

$$\ell(p) = \binom{h+t}{h} p^h (1-p)^t \approx p^h (1-p)^t \,.$$



Problems with Maximum Likelihood

- Example!

Bayes Theorem

- Classification from label distributions:

$$\Pr[Y = k \mid X = x] = \frac{\Pr[X = x \mid Y = k] \Pr[Y = k]}{\Pr[X = x]}$$

- Example:

$$\Pr[\text{default} = \text{yes} \mid \text{balance} = \$100] = \frac{\Pr[\text{balance} = \$100 \mid \text{default} = \text{yes}] \Pr[\text{default} = \text{yes}]}{\Pr[\text{balance} = \$100]}$$

- Notation:

$$\Pr[Y = k \mid X = x] = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}$$

Better Options

1. **Maximum likelihood**

$$\max_{\text{model}} \Pr[\text{data} \mid \text{model}]$$

2. **Maximum a posteriori estimate (MAP)**

$$\max_{\text{model}} \Pr[\text{model} \mid \text{data}]$$

Better Options

1. **Maximum likelihood**

$$\max_{\text{model}} \Pr[\text{data} \mid \text{model}]$$

2. **Maximum a posteriori estimate (MAP)**

$$\max_{\text{model}} \Pr[\text{model} \mid \text{data}] = \max_{\text{model}} \Pr[\text{data} \mid \text{model}] \frac{\Pr[\text{model}]}{\Pr[\text{data}]}$$

Better Options

1. **Maximum likelihood**

$$\max_{\text{model}} \Pr[\text{data} \mid \text{model}]$$

2. **Maximum a posteriori estimate (MAP)**

$$\max_{\text{model}} \Pr[\text{model} \mid \text{data}] = \max_{\text{model}} \Pr[\text{data} \mid \text{model}] \frac{\Pr[\text{model}]}{\Pr[\text{data}]}$$

Identical when the prior is normal

Maximum a Posteriori Estimate

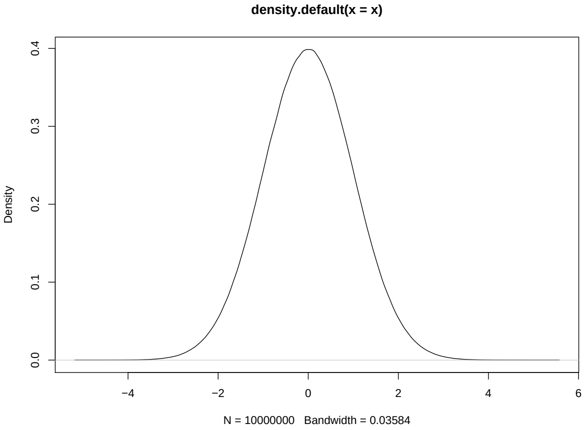
$$\Pr[\beta \mid X, Y] = \alpha f(Y \mid X, \beta) p(\beta \mid X) = \alpha f(Y \mid X, \beta) p(\beta)$$

- *Prior*: $p(\beta)$
- *Likelihood*: $f(Y \mid X, \beta)$

Normal Distribution

- Normal density function:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



Maximum Likelihood For OLS

- Assume $\epsilon_i \sim \mathcal{N}(0, 1)$
- Likelihood of a single data point

$$f(y_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(y_i - \hat{y}_i)^2}{2}}$$

- Recall

$$\hat{y}_i = \beta_0 + \beta_1 x_i$$

- Likelihood of all data

$$\prod_{i=1}^{n} f(y_i)$$