General Linear Models

Ridge and Lasso Regression

– O.Örsan Özener

Improving on the Least Squares Regression Estimates?

- We want to improve the Linear Regression model, by replacing the least square fitting with some alternative fitting procedure, i.e., the values that minimize the mean square error (MSE)

- There are 2 reasons we might not prefer to just use the ordinary least squares (OLS) estimates
  1. Prediction Accuracy
  2. Model Interpretability

1. Prediction Accuracy

- The least squares estimates have relatively low bias and low variability especially when the relationship between Y and X is linear and the number of observations n is way bigger than the number of predictors p
- But, when n=p (almost), then the least squares fit can have high variance and may result in over fitting and poor estimates on unseen observations,
- And, when n<p, then the variability of the least squares fit increases dramatically, and the variance of these estimates in infinite

Why can shrinking towards zero be a good thing to do?

- It turns out that the OLS estimates generally have low bias but can be highly variable. In particular when n and p are of similar size or when $n < p$, then the OLS estimates will be extremely variable

- The penalty term makes the ridge regression estimates biased but can also substantially reduce variance

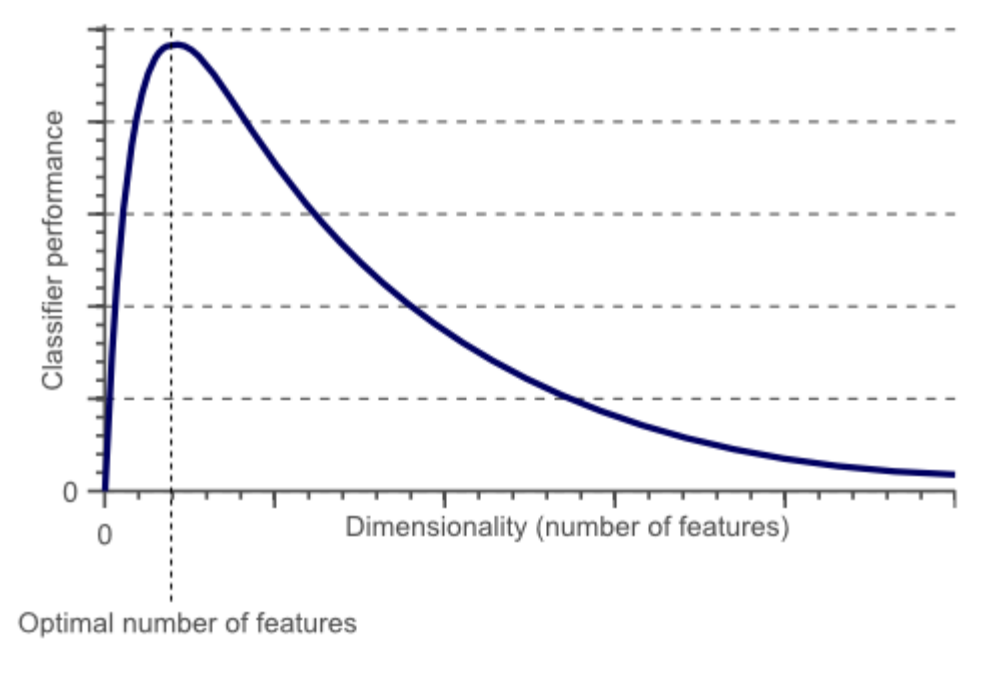
- Thus, there is a bias/variance trade-off

Curse of Dimensionality

- In most applications we observe a high dimensional data set
- It is inadvisable to use all the features
  - Redundant
  - Model complexity ~ tendency to overfit
  - Computational difficulty
  - Correlation
  - Anything else ?

Curse of Dimensionality

- Hughes Phenomenon: As the number of features increases, the classifier's performance increases until the optimal number of features. Adding more features based on the same size as the training set will then degrade the classifier's performance.

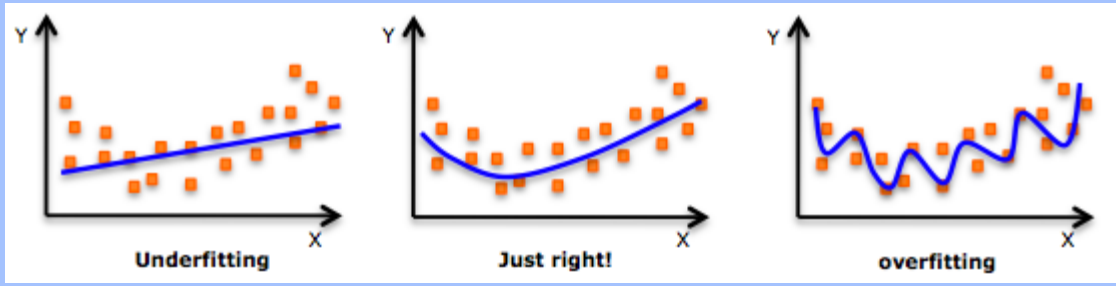


Bias and Variance Tradeoff

- In general, good estimators should, on average have, small prediction errors
- As model becomes more complex (more terms included), local structure/curvature can be picked up
- But coefficient estimates suffer from high variance as more terms are included in the model
- Therefore, introducing a little bias in our estimate for β might lead to a substantial decrease in variance, and hence to a substantial decrease in prediction error

Bias and Variance Tradeoff

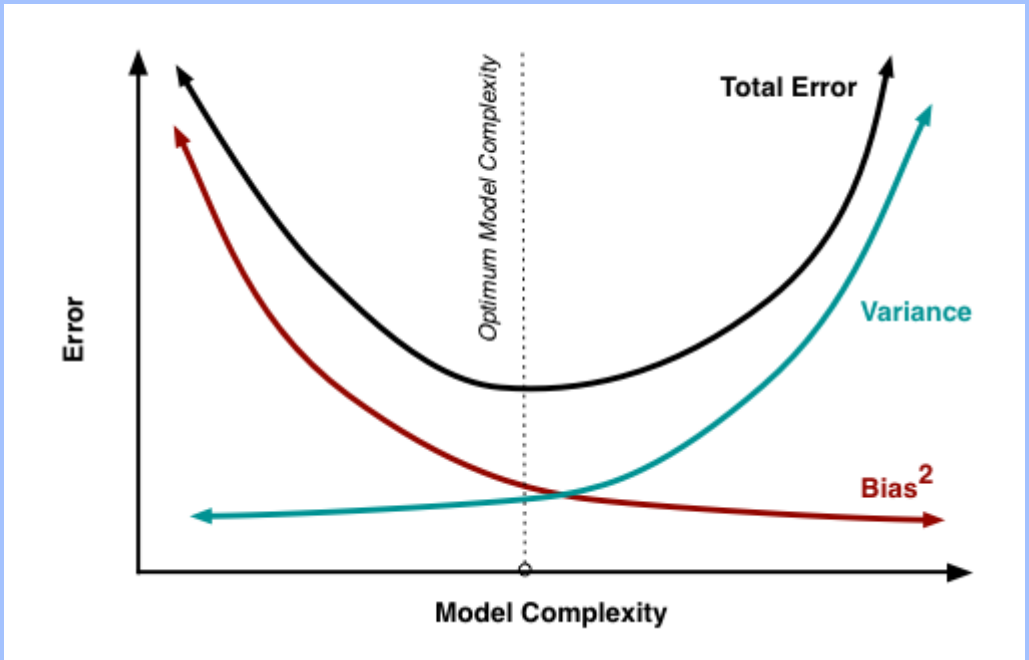
- Suppose we use a simple polynomial regression. As we increase the degree of the polynomial function, we observe



- Underfitting ~ high bias and low variance
- Overfitting ~ low bias and high variance

Bias and Variance Tradeoff

- We need to determine the optimum point in model complexity to avoid both underfitting and overfitting



2. Model Interpretability

- When we have a large number of variables X in the model there will generally be many that have little or no effect on Y
- Leaving these variables in the model makes it harder to see the “big picture”, i.e., the effect of the “important variables”
- The model would be easier to interpret by removing (i.e. setting the coefficients to zero) the unimportant variables

Solution

- Subset Selection
  - Identifying a subset of all p predictors X that we believe to be related to the response Y, and then fitting the model using this subset
  - E.g. best subset selection and stepwise selection
- Shrinkage
  - Involves shrinking the estimates coefficients towards zero
  - This shrinkage reduces the variance
  - Some of the coefficients may shrink to exactly zero, and hence shrinkage methods can also perform variable selection
  - E.g. Ridge regression and the Lasso
- Dimension Reduction
  - Involves projecting all p predictors into an M-dimensional space where $M < p$, and then fitting linear regression model
  - E.g. Principle Components Regression

Feature Selection

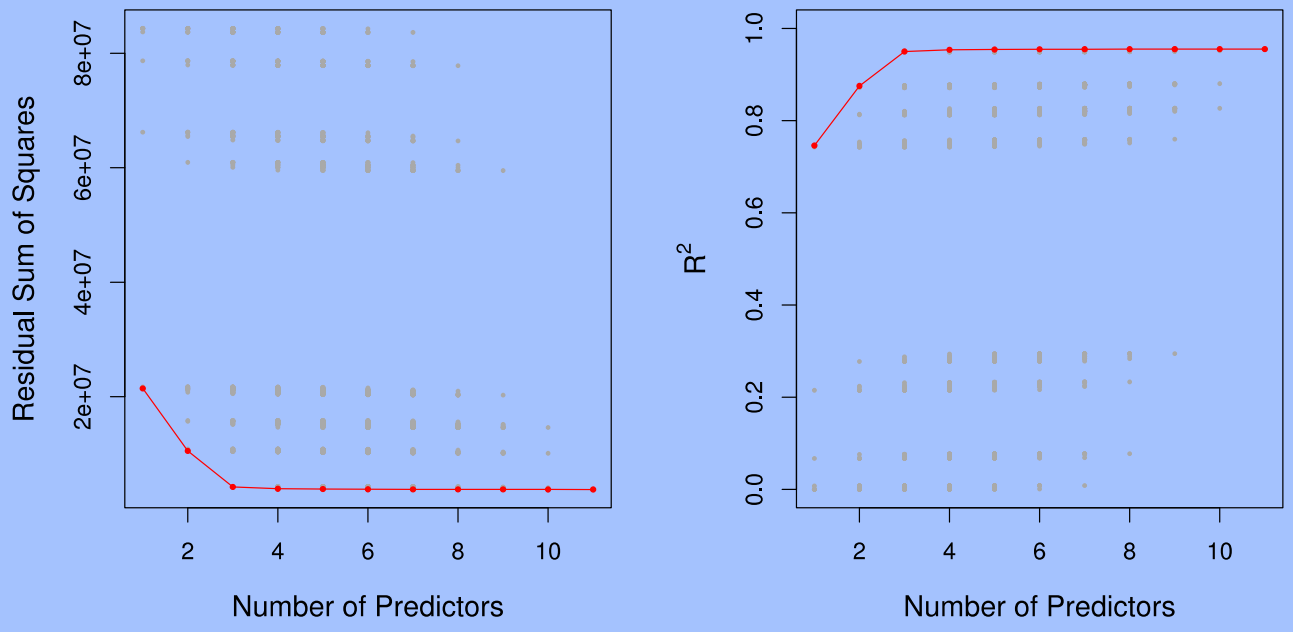
- Business (domain) knowledge
  - Select the features that are relevant and important
  - Minimize the correlation among the selected features
- Automated selection
  - Forward selection: Start with most significant predictor in the model and adds features in each step.
  - Backward elimination: Starts with all predictors in the model and removes the least significant feature in each step.
- Any reason not to use either method?

Best Subset Selection

- In this approach, we run a linear regression for each possible combination of the X predictors
- How do we judge which subset is the “best”?
- One simple approach is to take the subset with the smallest RSS or the largest R^2
- Unfortunately, one can show that the model that includes all the variables will always have the largest R^2 (and smallest RSS)

Credit Data: R^2 vs. Subset Size

- The RSS/R^2 will always decline/increase as the number of variables increase so they are not very useful
- The red line tracks the best model for a given number of predictors, according to RSS and R^2

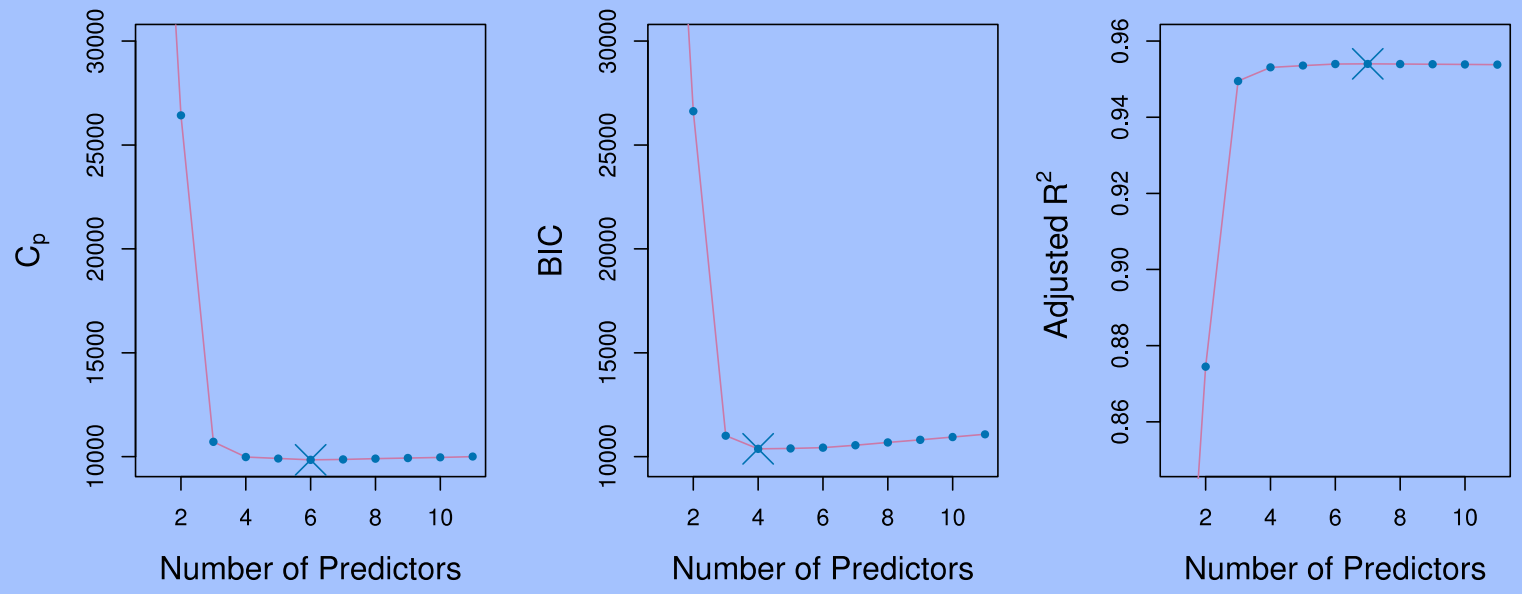


Other Measures of Comparison

- To compare different models, we can use other approaches:
  - Adjusted R^2
  - AIC (Akaike information criterion)
  - BIC (Bayesian information criterion)
  - C_p (equivalent to AIC for linear regression)
- These methods add penalty to RSS for the number of variables (i.e. complexity) in the model
- None are perfect

Credit Data: C_p, BIC, and Adjusted R^2

- A small value of C_p and BIC indicates a low error, and thus a better model
- A large value for the Adjusted R^2 indicates a better model



Stepwise Selection

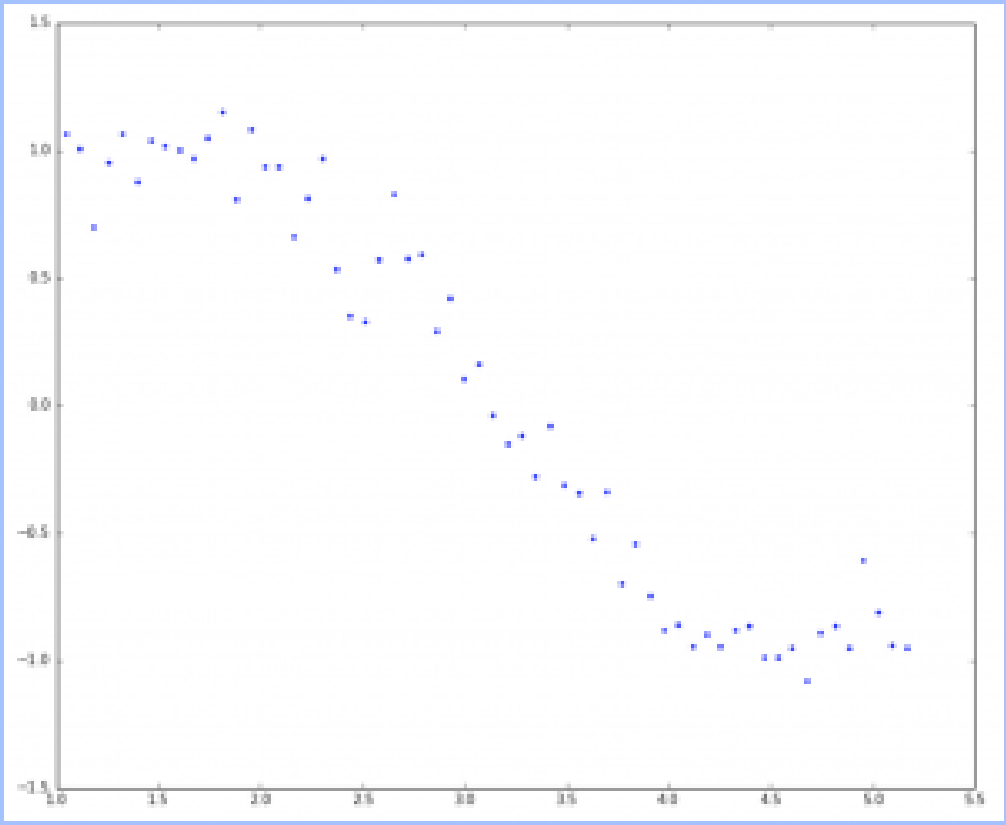
- Best Subset Selection is computationally intensive especially when we have a large number of predictors (large p)
- More attractive methods:
  - Forward Stepwise Selection: Begins with the model containing no predictor, and then adds one predictor at a time that improves the model the most until no further improvement is possible
  - Backward Stepwise Selection: Begins with the model containing all predictors, and then deleting one predictor at a time that improves the model the most until no further improvement is possible

Regularization

- To overcome overfitting, we should either reduce the complexity of the model or use regularization
- In regularization we reduce the magnitude of the coefficients by penalizing them
- Because if the βj's are unconstrained
  - They can explode
  - Hence are susceptible to very high variance
- To control variance, we might regularize the coefficients
  - Control how large the coefficients grow

Motivation for Penalizing Coefficients

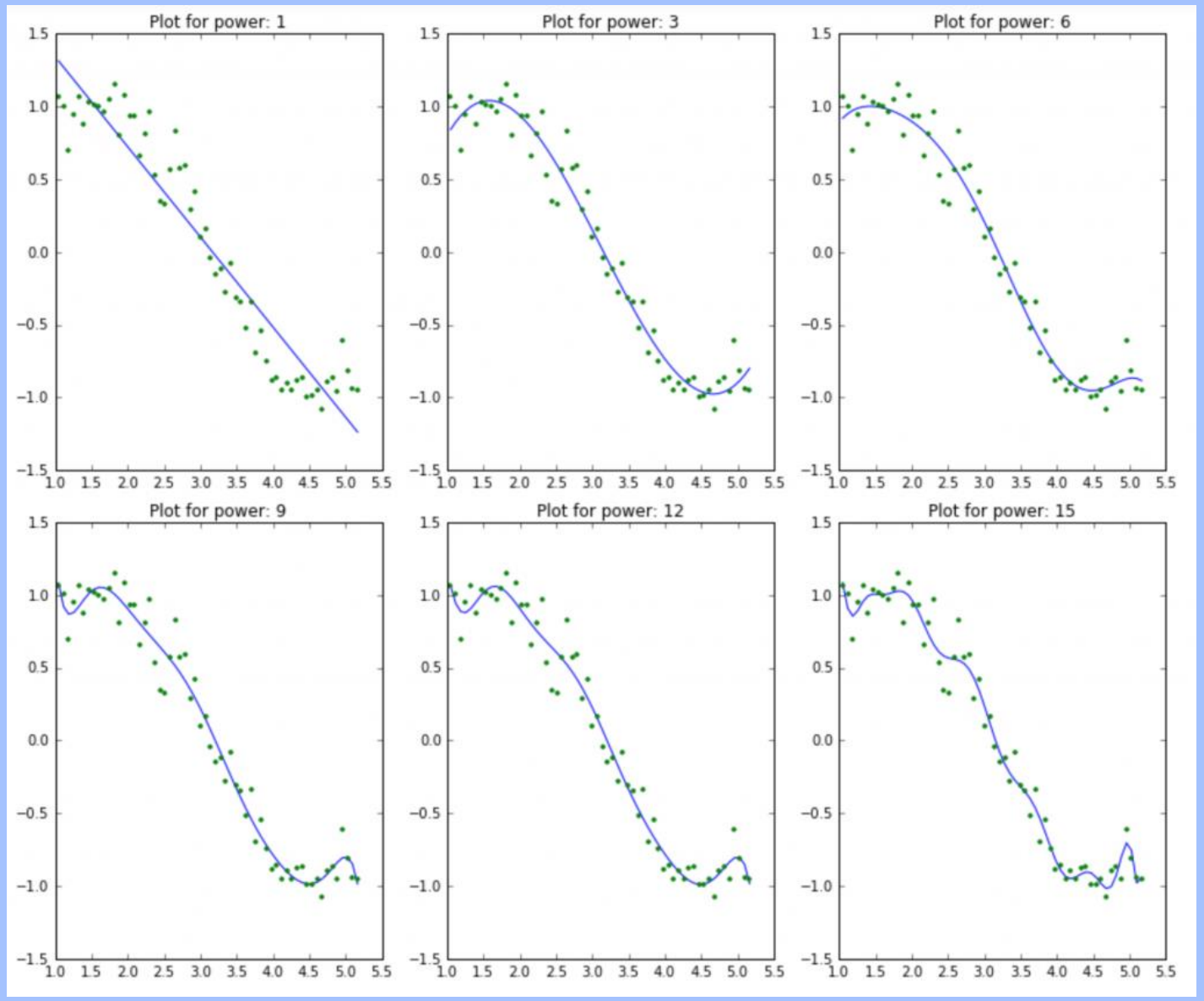
- Plot a sine curve with random noise



Motivation for Penalizing Coefficients

- Estimate the sine function using **polynomial regression** with powers of x up to 15.
- We will have 15 different polynomial regression starting with linear to polynomial with degree 15.

Motivation for Penalizing Coefficients



Motivation for Penalizing Coefficients

- The sizes of coefficients increase exponentially with increase in model complexity.
- Check the Table Poly in Lecture5a.xlxs
- Large coefficient ~ high emphasis on feature.
- Too large coefficient ~ algorithm starts overfitting

Ridge Regression

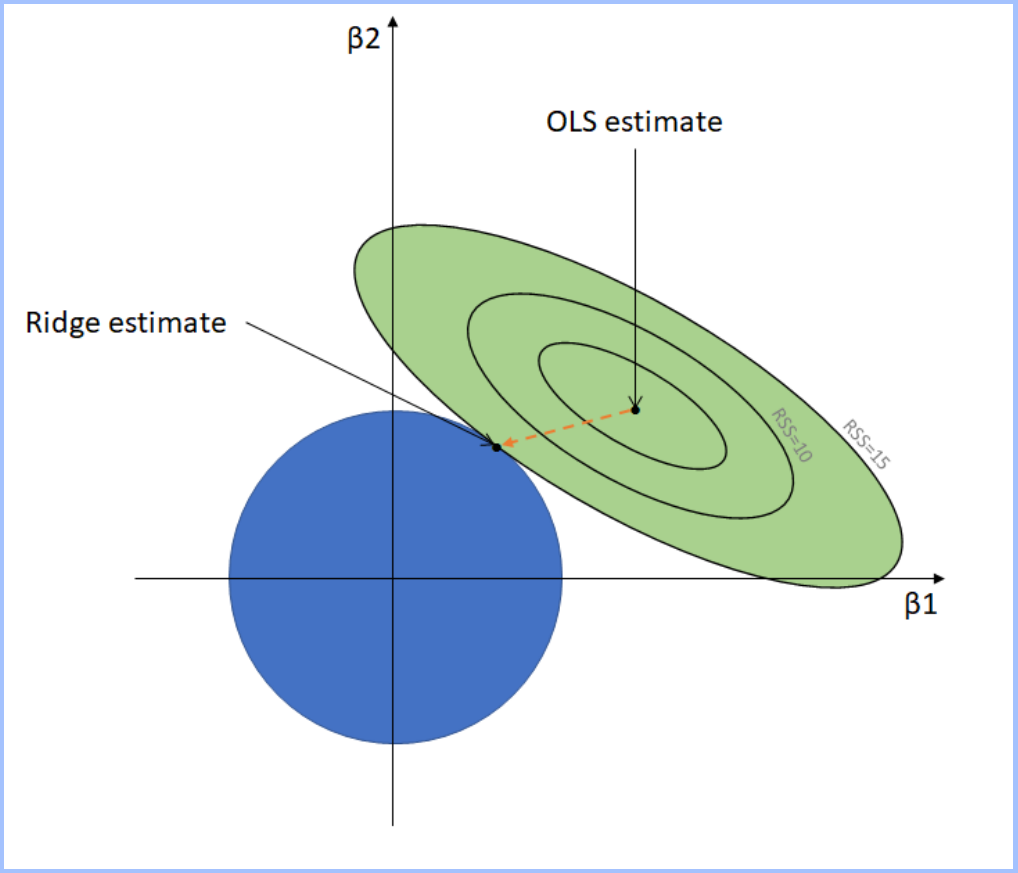
- **Ridge Regression:**
  - Performs L2 regularization, i.e. adds penalty equivalent to **square of the magnitude** of coefficients
  - Minimization objective = LS Obj + λ * (sum of square of coefficients)
- In ridge constraint form
  - minimize $\Sigma(y_i - \beta^T z_i)^2$
  - s.t. $\Sigma\ \beta_j^2 \leq t$
- In function form
  - minimize $\Sigma(y_i - \beta^T z_i)^2 + \lambda\ (\Sigma\ \beta_j^2 - t)$

Penalty Coefficient

- λ (or alpha parameter extra term) is the penalty term in the ridge function.
- Changing the values of λ: controlling the importance on the penalty term/cost
- Higher the values of λ leads to bigger total penalty cost and as a result the magnitude of coefficients are reduced.
- It shrinks the parameters, therefore it is mostly used to prevent multi-collinearity.
- It reduces the model complexity by coefficient shrinkage.

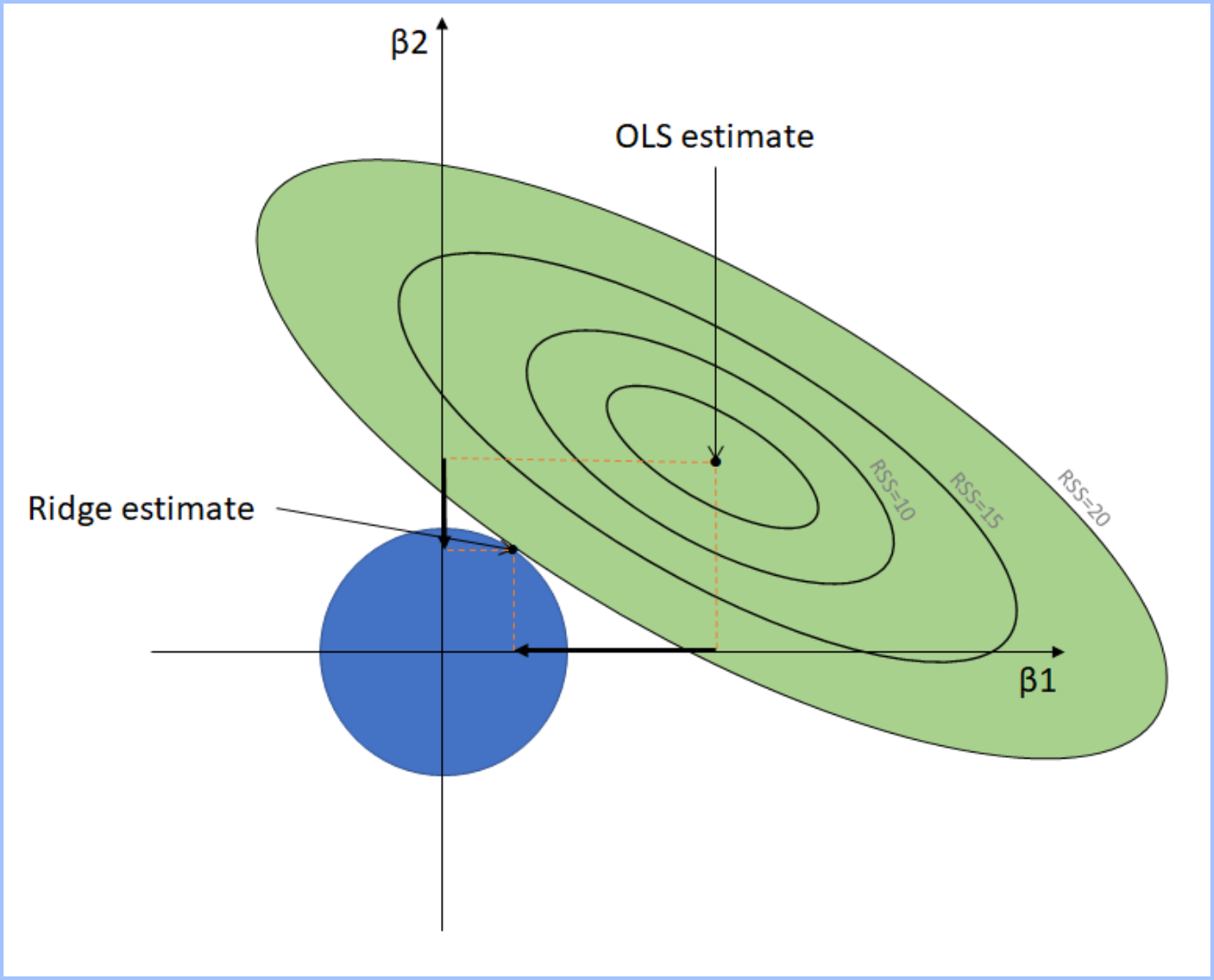
Ridge Regression vs OLS

- "Iso-RSS" lines



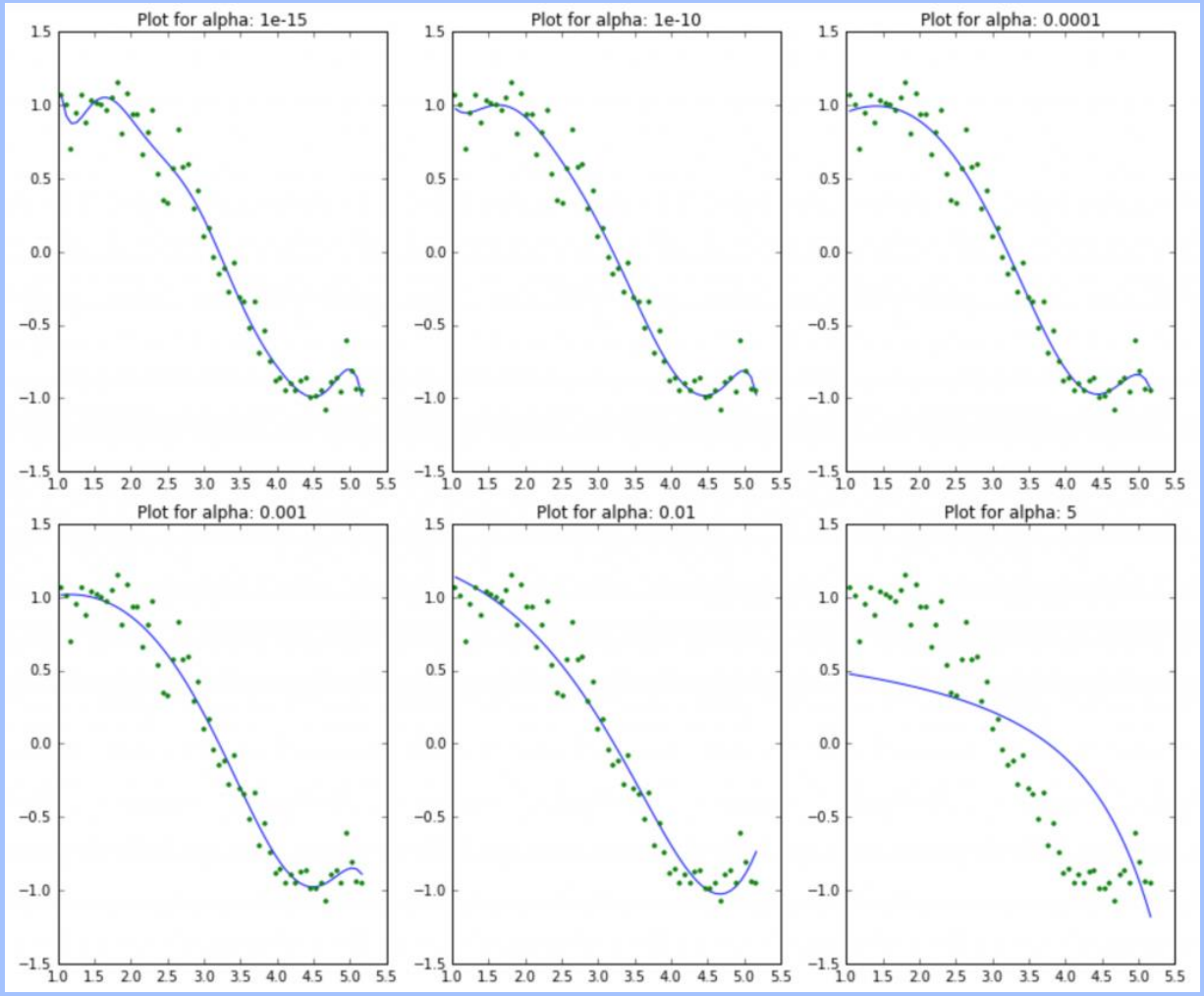
Ridge Regression vs OLS

- "Iso-RSS" lines



Ridge Regression

- Use different λ values for the sine function prediction

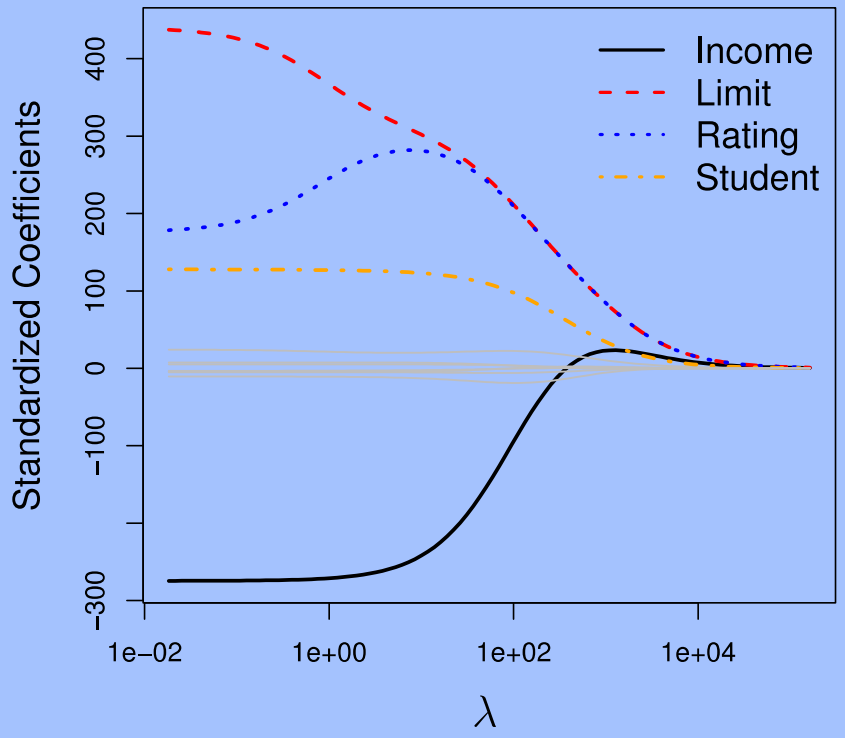


Ridge Regression

- Check the Table Ridge in Lecture5a.xlxs
- The RSS increases with increase in λ, this model complexity reduces
- An λ as small as 1e-15 gives us significant reduction in magnitude of coefficients.
- High λ values can lead to significant underfitting. Note the rapid increase in RSS for values of λ greater than 1
- Though the coefficients are **very small**, they are **NOT zero**.

Credit Data: Ridge Regression

- As λ increases, the standardized coefficients shrinks towards zero.



Choosing λ

- As λ increases, the model complexity reduces.
- Though higher values of λ reduce overfitting, significantly high values can cause underfitting as well
- Thus λ should be chosen wisely.
- Obviously want to choose λ that minimizes the mean squared error
- Standard practice is to use cross-validation
  - the value of alpha is iterated over a range of values and the one giving higher cross-validation score is chosen.

My Great Great Grand Advisor ☺

Lagrange Relaxation

Subgradient Optimization: Now, look at the following problem, for some fixed value of λ

$$\text{Max } z = 8x_1 + 9x_2 + 5x_3 + 4x_4 + \lambda(42 - 16x_1 - 20x_2 - 12x_3 - 10x_4)$$

Subject to (s.t.)

$$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4 \quad \{0,1\}$$

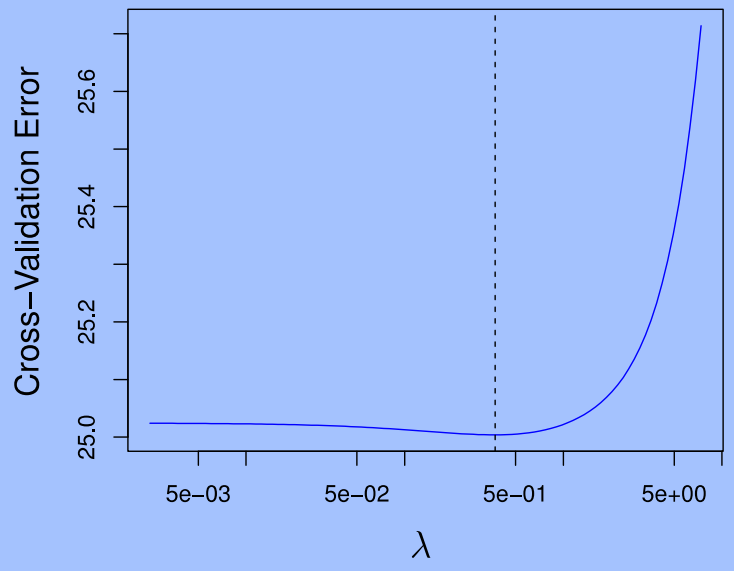
Choosing λ

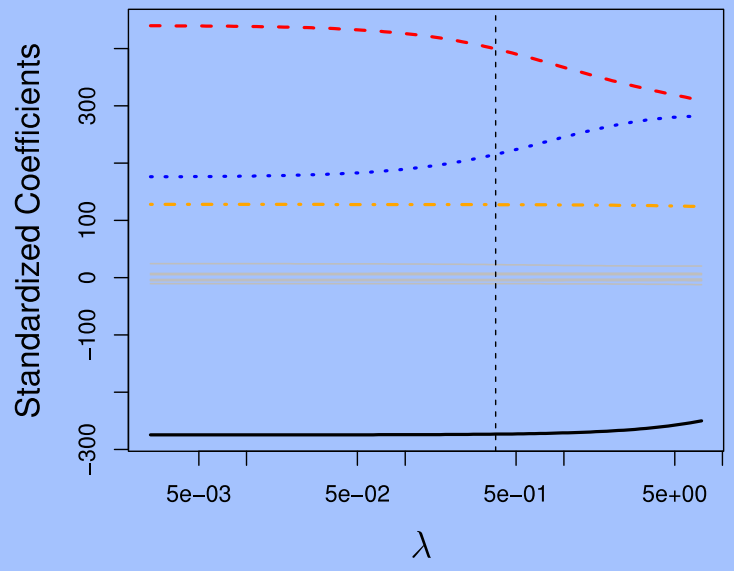
- To choose λ through cross-validation, you should choose a set of P values of λ to test, split the dataset into K folds

- for p in 1:P:
  - for k in 1:K:
    - keep fold k as hold-out data
    - use the remaining folds and $\lambda = \lambda_p$ to estimate β_{ridge}
    - predict hold-out data: $y_{test,k}=X_{test,k}\beta_{ridge}$
    - compute a sum of squared residuals: $SSR_k = ||y - y_{test,k}||^2$
  - end for k
  - average SSR over the folds: $SSR_p=(1/K)\sum SSR_k$
  - end for p
  - choose optimal value: $\lambda_{opt} = \text{argmin}_p SSR_p$

Selecting the Tuning Parameter

- We need to decide on a value for λ
- Select a grid of potential values, use cross validation to estimate the error rate on test data (for each value of λ) and select the value that gives the least error rate





Lasso Regression

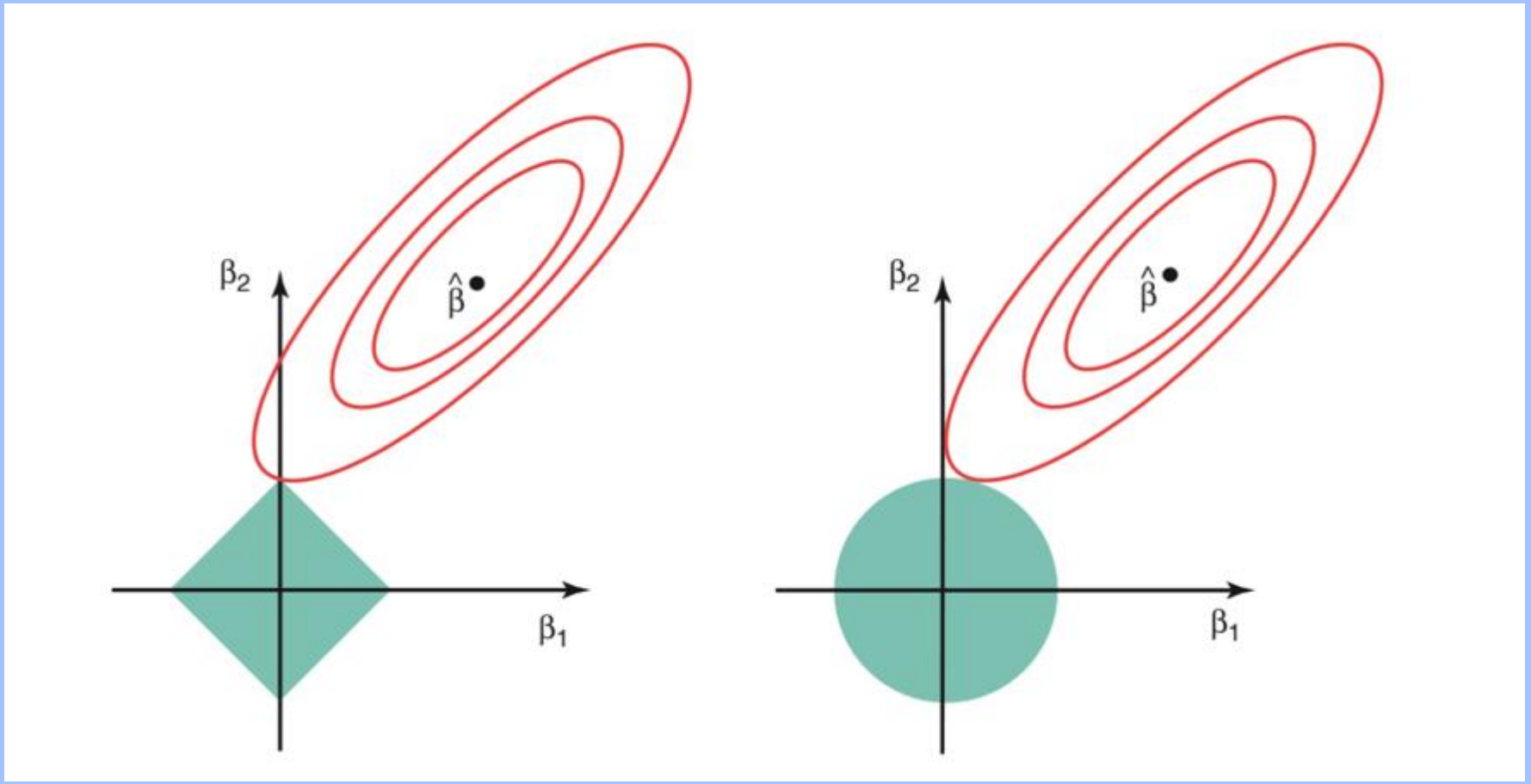
- Lasso: least absolute shrinkage and selection operator
- **Lasso Regression:**
  - Performs L1 regularization, i.e. adds penalty equivalent to **absolute value of the magnitude** of coefficients
  - Minimization objective = LS Obj + λ * (sum of absolute value of coefficients)
- In Lasso Regression, we impose the lasso constraint to the coefficients
  - minimize $\Sigma(y_i - \beta^T z_i)^2$
  - s.t. $\Sigma\ |\beta_j| \leq t$
- In function form
  - minimize $\Sigma(y_i - \beta^T z_i)^2 + \lambda\ (\Sigma\ |\beta_j| - t)$

Lasso Regression

- Even with low values of λ coefficients of some features are reduced to zero
- Lasso selects the only a subset of the entire feature space
- Hence Lasso performs an automatic feature selection whereas Ridge does not

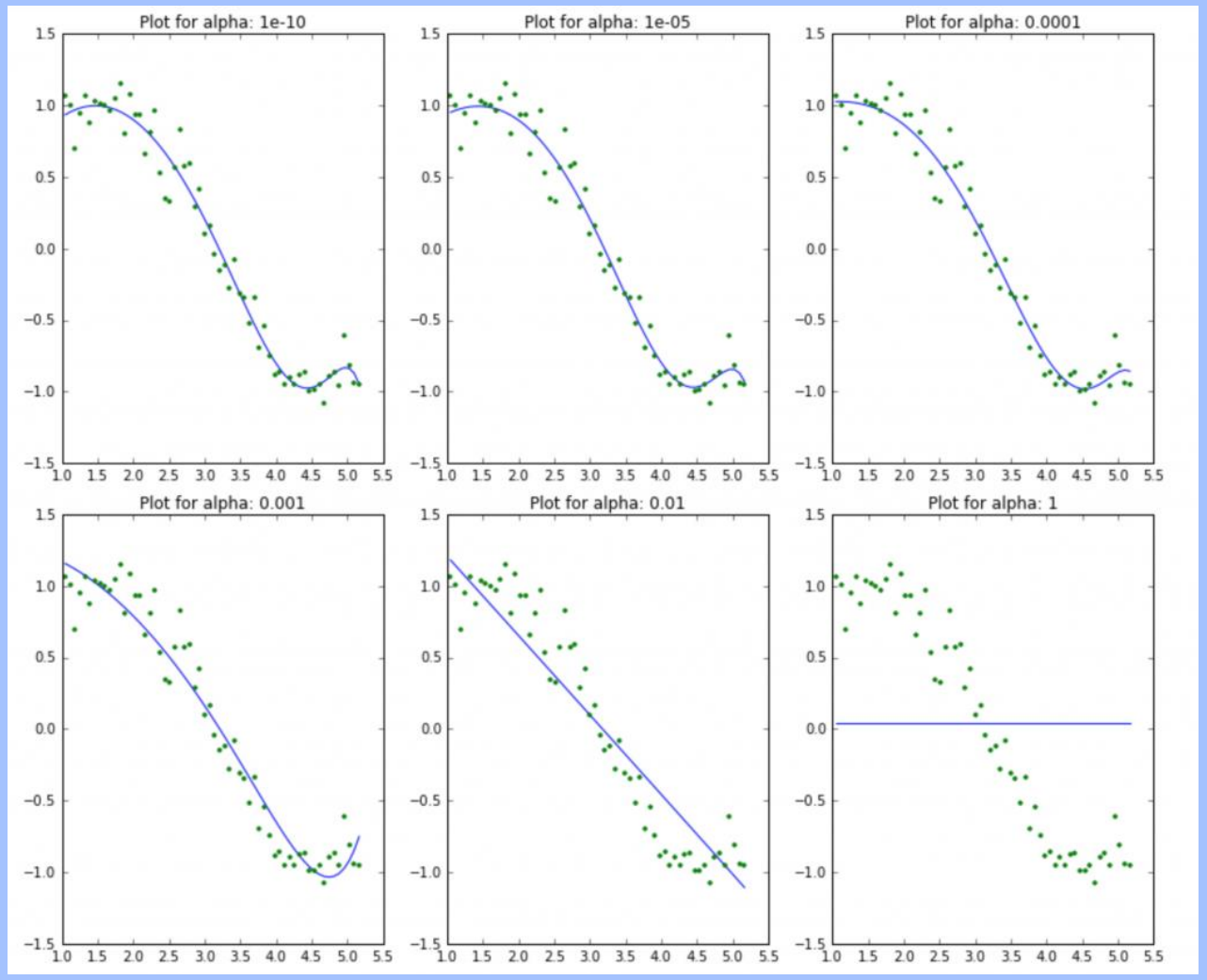
Lasso Regression vs OLS

- "Iso-RSS" lines



Lasso Regression

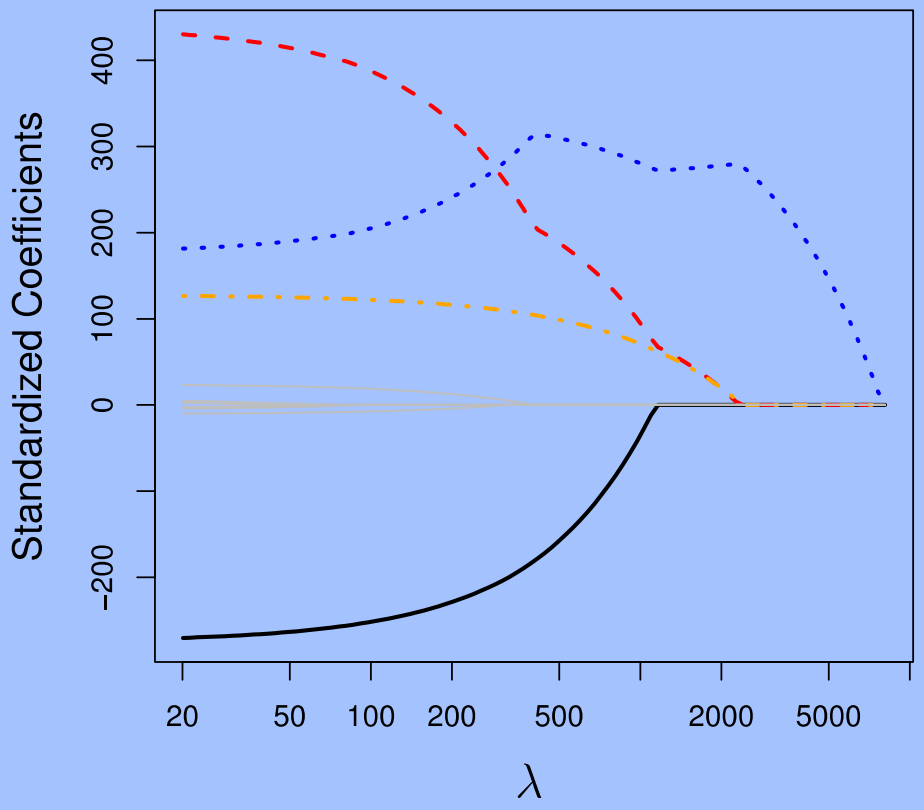
- Use different λ values for the sine function prediction



Lasso Regression

- Check the Table Lasso in Lecture5a.xlxs
- Compared to the previous table, what are the differences?

Credit Data: LASSO



Lasso Regression

- For the same values of λ, the coefficients of lasso regression are much smaller as compared to that of ridge regression
- For the same λ, lasso has higher RSS (poorer fit) as compared to ridge regression
- Many of the coefficients are zero even for very small values of λ ~ sparsity

Ridge Regression vs Lasso Regression

- **Ridge:**
  - includes all of the features in the model.
  - major advantage of ridge regression is coefficient shrinkage
  - reducing model complexity.
- **Lasso:**
  - feature selection
- Compare to standard feature selection ridge and lasso regression provide
  - **better output**,
  - can be **automated**

Ridge Regression vs Lasso Regression

- **Use Cases ~ Ridge:**
  - *Prevent overfitting*.
  - Not preferred when the number of features are really high.
  - Works well even in presence of highly correlated features as it will include all of them in the model but the coefficients will be distributed among them depending on the correlation.
- **Use Cases ~ Lasso:**
  - S*parse solutions*
  - Preferred when the number of features are really high
  - Arbitrarily selects any one feature among the highly correlated ones and reduced the coefficients of the rest to zero. Also, the chosen variable changes randomly with change in model parameters. This generally doesn't work that well as compared to ridge regression.

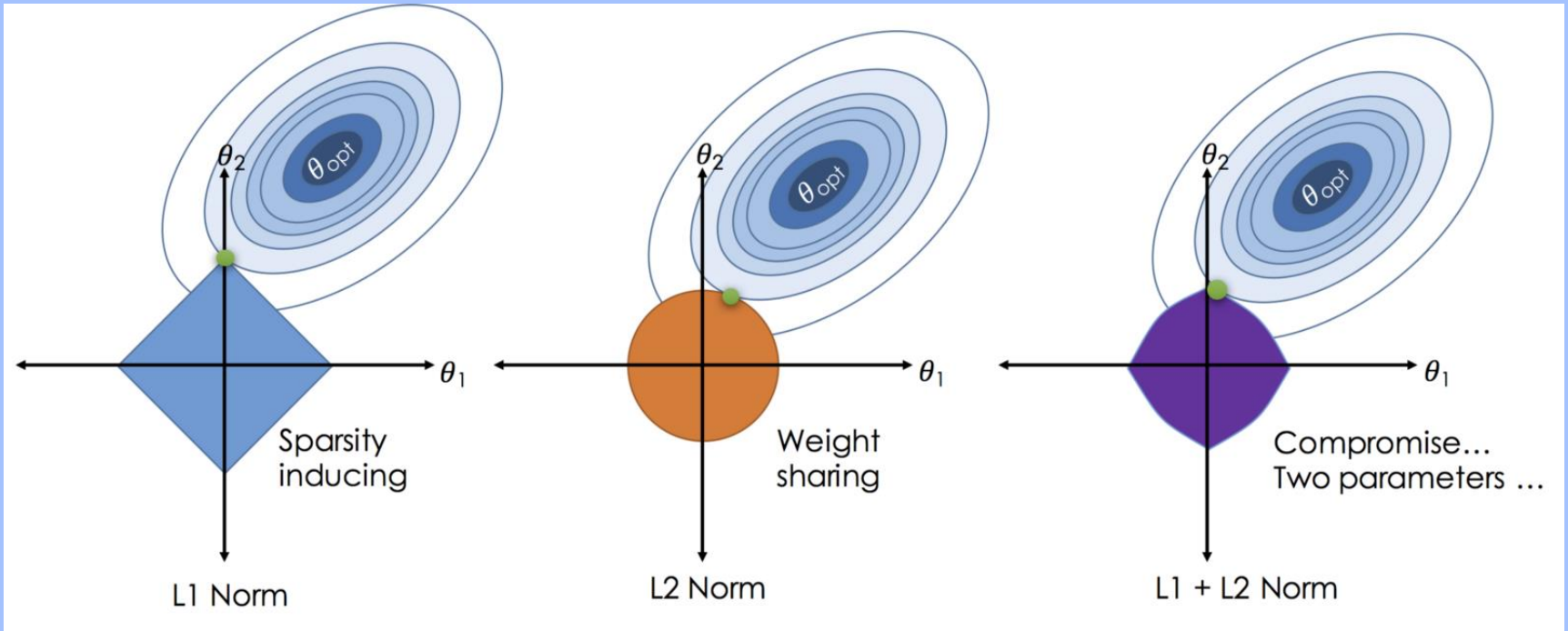
Data with High Number of Feature

- Suppose we have a data with a large number of features
- Applying Ridge regression
  - Keep all the features
  - Shrink the coefficients
  - Complexity may not be reduced with that many features
- Applying Lasso regression
  - Removing some of the features
  - Correlated variables ~ keeps only one variable
  - Loss of information resulting in lower accuracy

Elastic Net Regression

- Elastic Net Regression is basically a combination of both Ridge and Lasso regression
- Elastic Net is useful when there are multiple features which are correlated. Lasso is likely to pick one of these at random, while Elastic Net is likely to pick both.
- Two penalty terms with two parameters
  - Min $\Sigma(y_i - \beta^T z_i)^2 + \lambda_1 (\Sigma |\beta_j| - t) + \lambda_2 (\Sigma \beta_j^2 - t)$

Elastic Net Regression



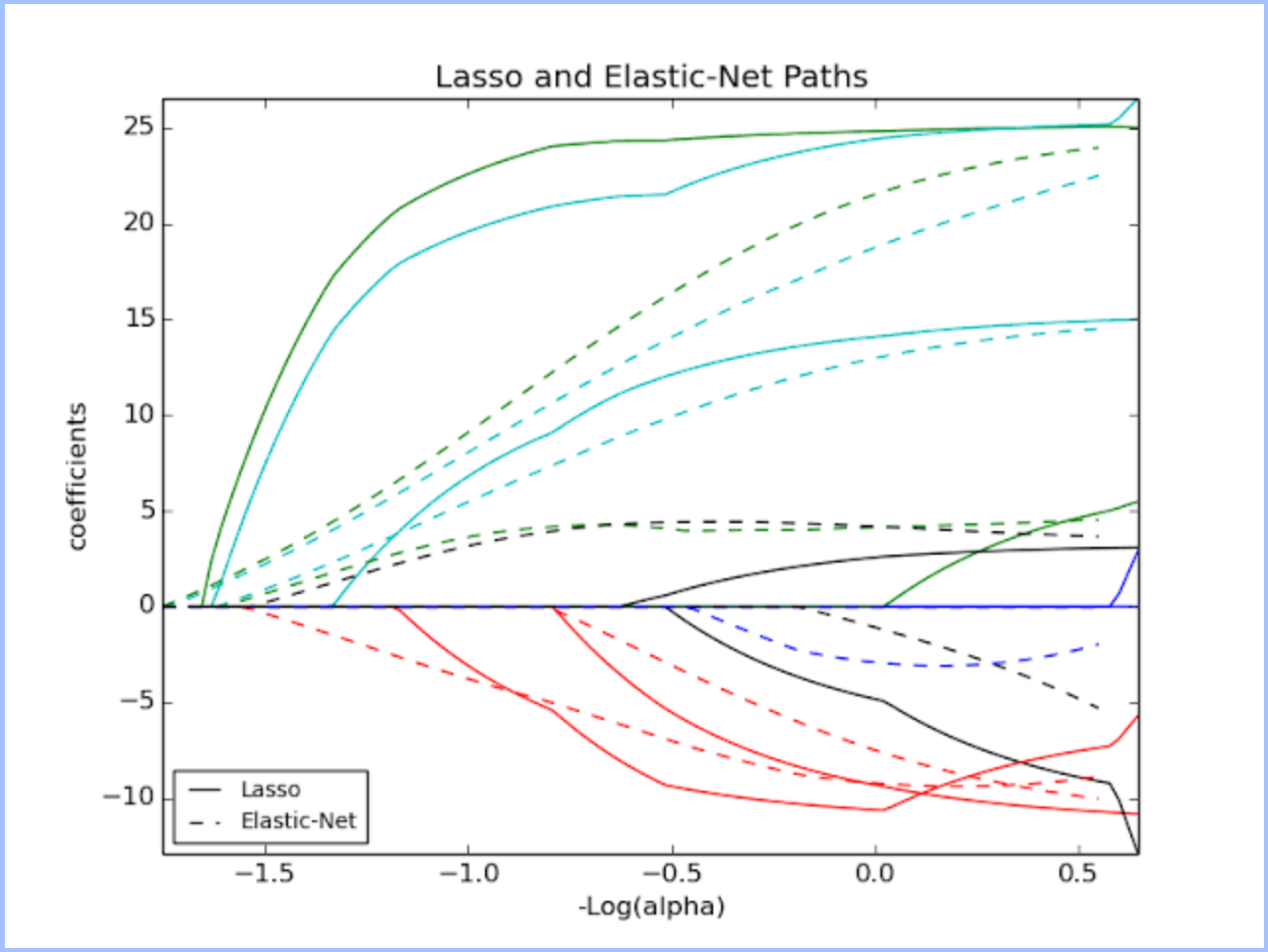
Elastic Net Regression

- Elastic Net Regression
  - It encourages group effect in case of highly correlated variables
  - There are no limitations on the number of selected variables
  - It can suffer with double shrinkage

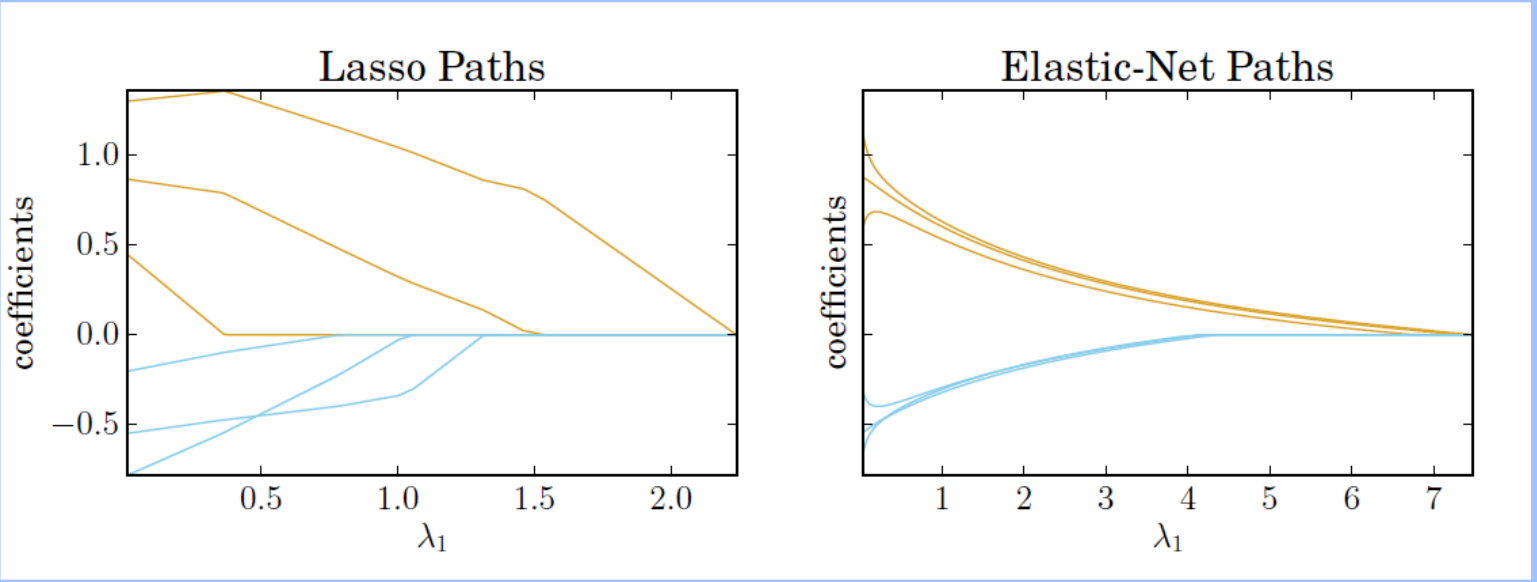
Elastic Net Regression

- When variables are highly correlated (and same scale, after normalization),
  - we want to give them roughly the same weight.
- Why?
  - Let their error cancel out

Elastic Net Regression



Elastic Net Regression



Dimension Reduction Methods

- The methods that we have discussed so far in this chapter have involved fitting linear regression models, via least squares or a shrunken approach, using the original predictors, $X_1, X_2, \ldots, X_p$.
- 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 Methods

Let $Z_1, Z_2, \ldots, Z_M$ represent $M < p$ *linear combinations* of our original p predictors. That is,

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

for some constants $\varphi_{m1}, \ldots, \varphi_{mp}$.

We can then fit the linear regression model,

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

using ordinary least squares.

Note that in model, the regression coefficients are given by $\theta_0, \theta_1, \ldots, \theta_M$. If the constants $\varphi_{m1}, \ldots, \varphi_{mp}$ are chosen wisely, then such dimension reduction approaches can often outperform OLS regression.

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

where

$$\beta_j = \sum_{m=1}^{M} \theta_m \phi_{mj}. \qquad (3)$$

Hence 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, since now they must take the form.

Can win in the bias-variance tradeoff.

Principal Components Analysis

- PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.
- Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.

Principal Components Regression

- Here we apply principal components analysis (PCA) to define the linear combinations of the predictors, for use in our regression.
- The first principal component is that (normalized) linear combination of the variables with the largest variance.
- The second principal component has largest variance, subject to being uncorrelated with the first.
- And so on.
- Hence with many correlated original variables, we replace them with a small set of principal components that capture their joint variation.

Principal Components Regression

- The *first principal component* of a set of features X_1, X_2, . .., X_p is the normalized linear combination of the features

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \ldots + \phi_{p1}X_p$$

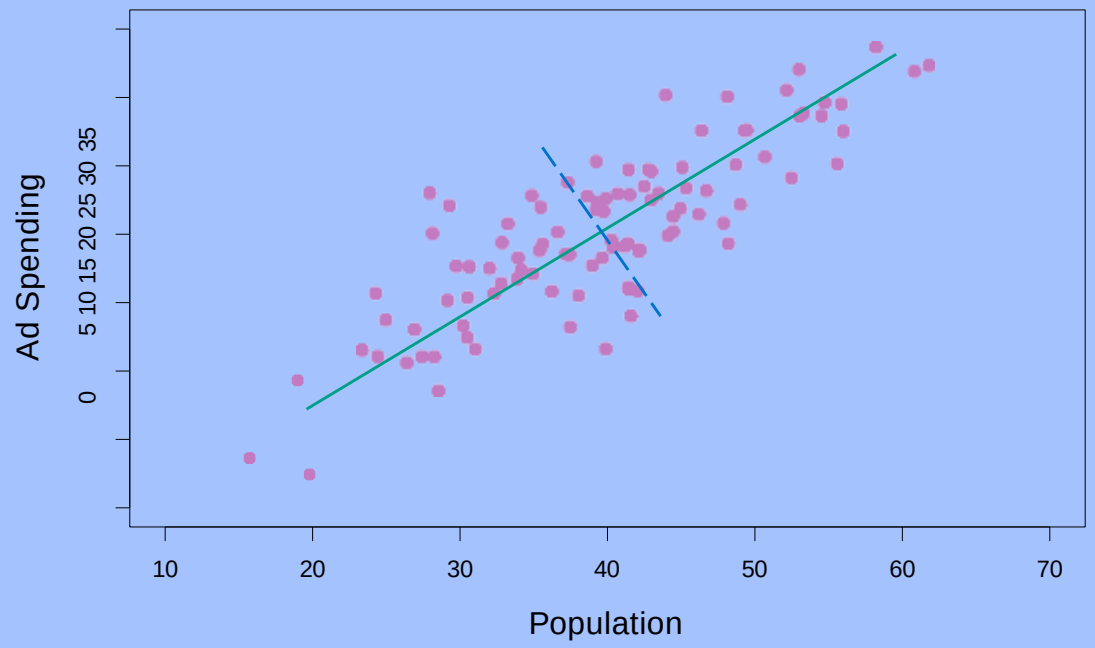
that has the largest variance. By *normalized*, we mean that

$$\sum_{j=1}^{p} \phi_{j1}^2 = 1.$$

We refer to the elements $\varphi_{11}, \ldots, \varphi_{p1}$ as the loadings of the first principal component; together, the loadings make up the principal component loading vector, $\varphi_1 = (\varphi_{11}\, \varphi_{21} \ldots \varphi_{p1})^T$.

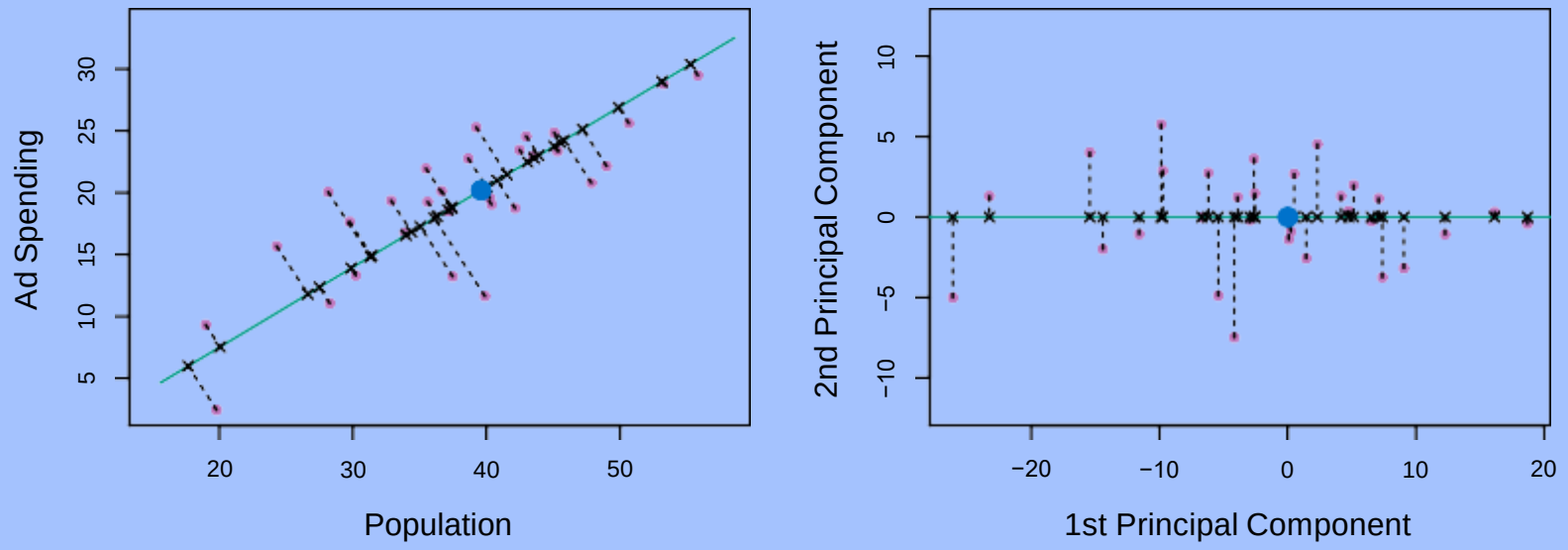
We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

Principal Components Regression



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.*

Principal Components Regression



A subset of the advertising data. Left: The first principal component, chosen to minimize the sum of the squared perpendicular distances to each point, is shown in green. These distances are represented using the black dashed line segments. Right: The left-hand panel has been rotated so that the first principal component lies on the x-axis.

Computation of Principal Components

- Suppose we have a $n \times p$ data set **X**. Since we are only interested in variance, we assume that each of the variables in **X** has been centered to have mean zero (that is, the column means of **X** are zero).
- We then look for the linear combination of the sample feature values of the form

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \ldots + \phi_{p1} x_{ip}$$

- for $i = 1, \ldots, n$ that has largest sample variance, subject to

$$\sum_{j=1}^{p} \phi_{j1}^2 = 1.$$

- Since each of the x_{ij} has mean zero, then so does z_{i1} (for any values of φ_{j1}). Hence the sample variance of the z_{i1} can be written as

$$\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2.$$

Computation of Principal Components

- Plugging in the first principal component loading vector solves the optimization problem

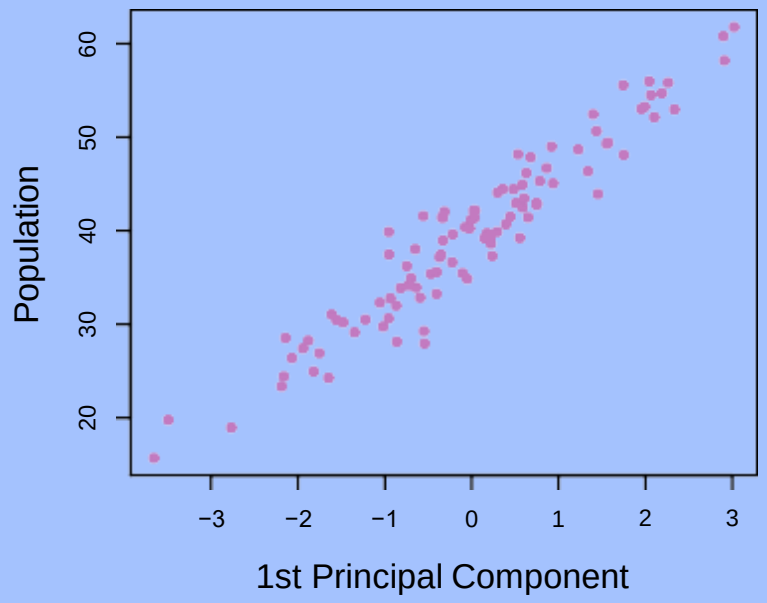
$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{j1}x_{ij}\right)^2 \text{ subject to } \sum_{j=1}^{p}\phi_{j1}^2 = 1.$$

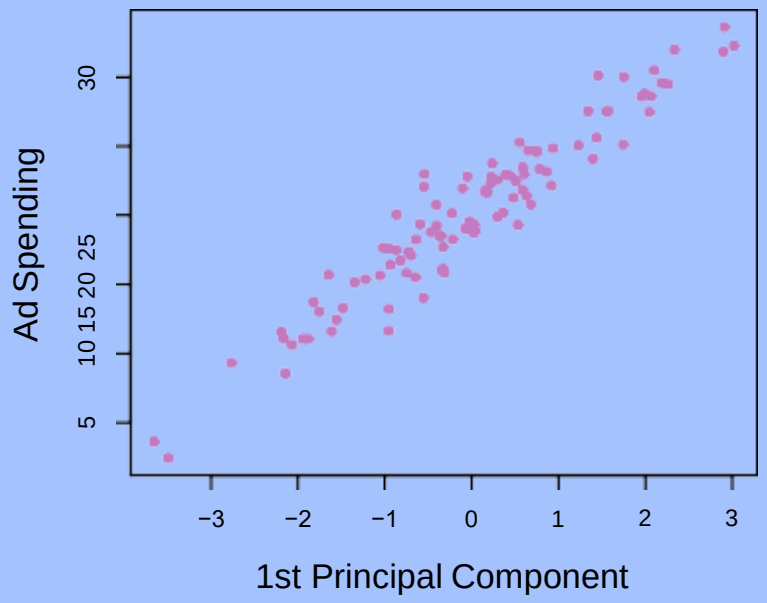
- This problem can be solved via a singular-value decomposition of the matrix **X**, a standard technique in linear algebra.
- We refer to Z_1 as the first principal component, with realized values $z_{11}, \ldots, z_{n1}$

Geometry of PCA

- The loading vector φ_1 with elements $\varphi_{11}, \varphi_{21}, \ldots, \varphi_{p1}$ defines a direction in feature space along which the data vary the most.
- If we project the n data points $x_1, \ldots, x_n$ onto this direction, the projected values are the principal component scores $z_{11}, \ldots, z_{n1}$ themselves.

Principal Components Regression





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

Computation of Principal Components

- The second principal component is the linear combination of $X_1, \ldots, X_p$ that has maximal variance among all linear combinations that are *uncorrelated* with Z_1.
- The second principal component scores $z_{12}, z_{22}, \ldots, z_{n2}$ take the form

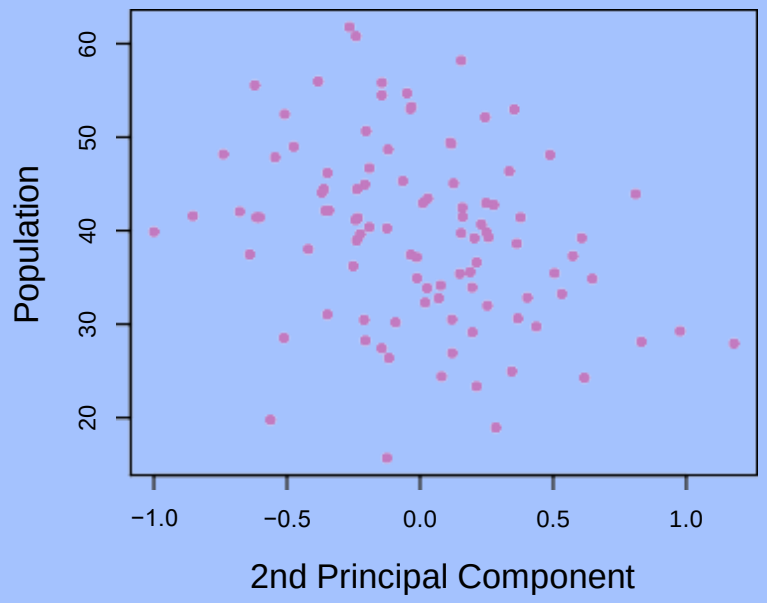
$$z_{i2} = \varphi_{12}x_{i1} + \varphi_{22}x_{i2} + \ldots + \varphi_{p2}x_{ip},$$

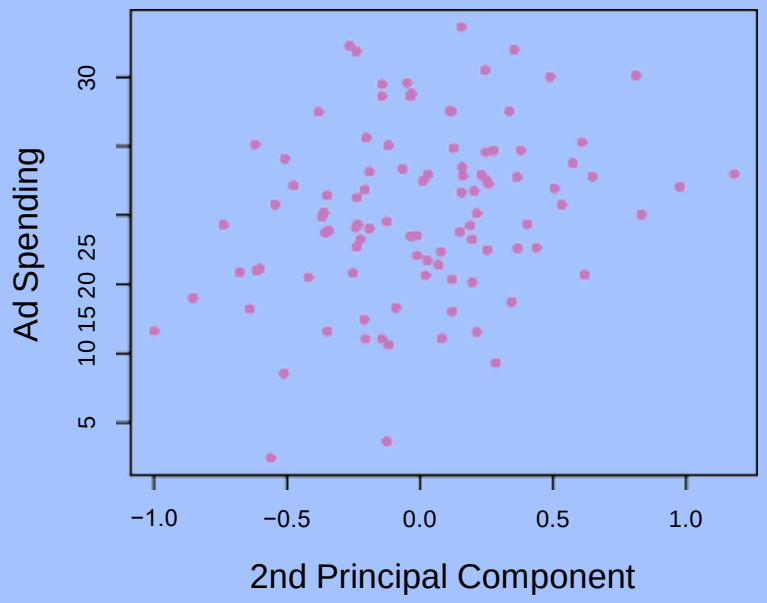
where φ_2 is the second principal component loading vector, with elements $\varphi_{12}, \varphi_{22}, \ldots, \varphi_{p2}$.

Computation of Principal Components

- It turns out that constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining the direction φ_2 to be orthogonal (perpendicular) to the direction φ_1. And so on.
- The principal component directions φ_1, φ_2, φ_3, . . . are the ordered sequence of right singular vectors of the matrix **X**, and the variances of the components are 1/n times the squares of the singular values. There are at most min(n − 1, p) principal components.

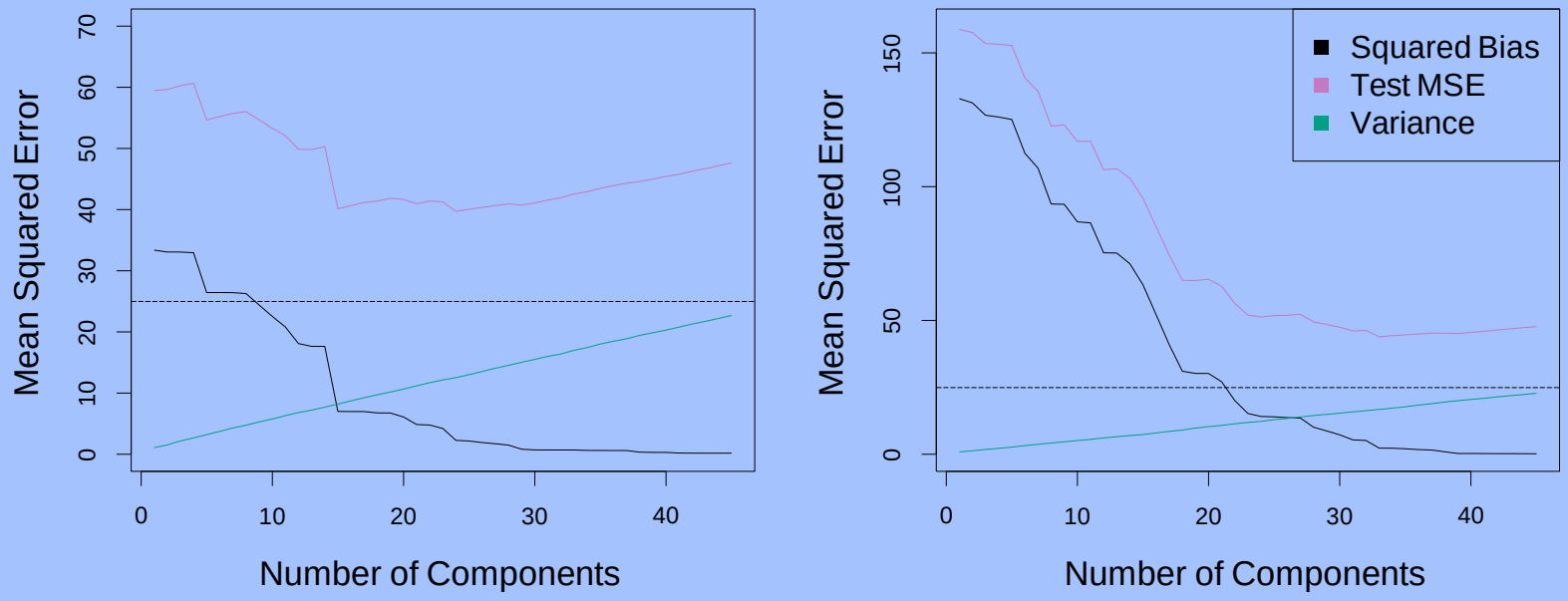
Principal Components Regression





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

Application to Principal Components Regression

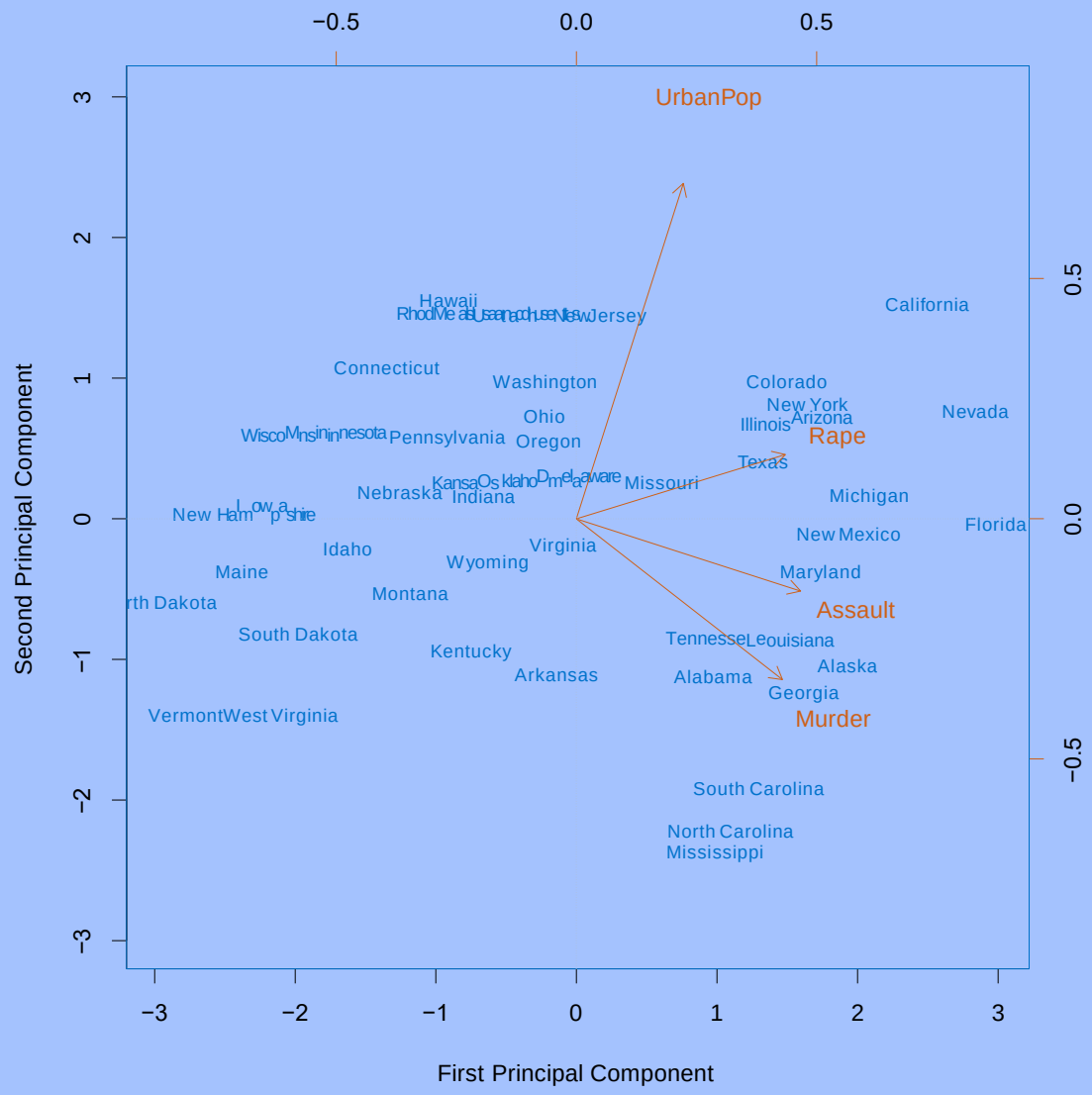


PCR was applied to two simulated data sets. The black, green, and purple lines correspond to squared bias, variance, and test mean squared error, respectively.

Example

- USAarrests data: For each of the fifty states in the United States, the data set contains the number of arrests per 100, 000 residents for each of three crimes: Assault, Murder, and Rape. We also record UrbanPop (the percent of the population in each state living in urban areas).
- The principal component score vectors have length $n = 50$, and the principal component loading vectors have length $p = 4$.
- PCA was performed after standardizing each variable to have mean zero and standard deviation one.

USAarrests data: PCA plot

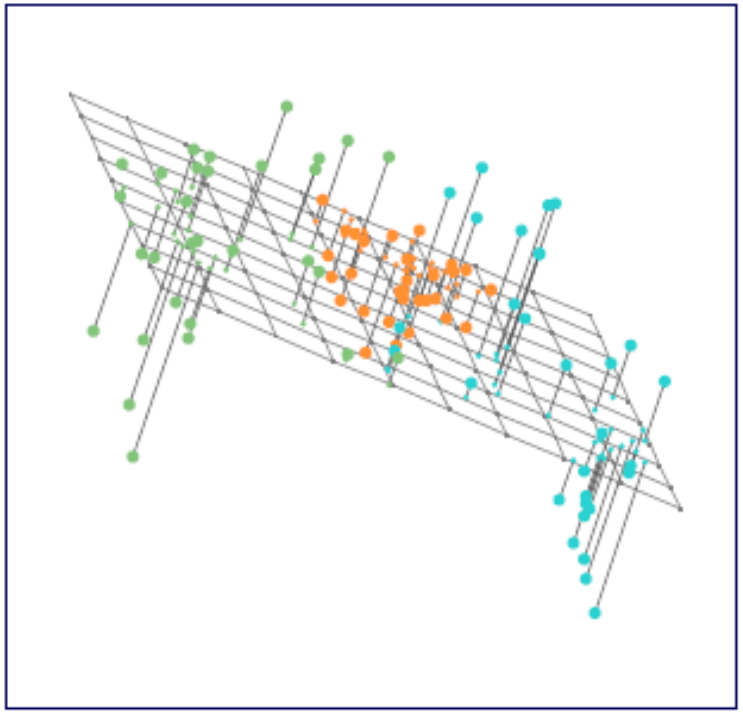


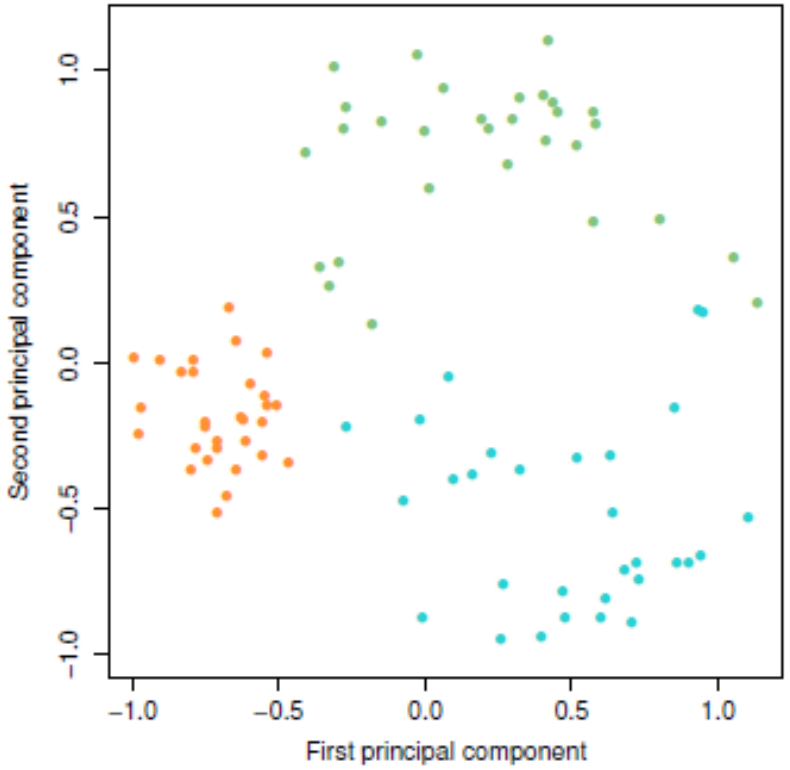
Example

- The first two principal components for the USArrests data.
- The blue state names represent the scores for the first two principal components.
- The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for Rape on the first component is 0.54, and its loading on the second principal component 0.17 [the word Rape is centered at the point (0.54, 0.17)].
- This figure is known as a *biplot*, because it displays both the principal component scores and the principal component loadings.

	PC1	PC2
Murder	0.5358995	-0.4181809
Assault	0.5831836	-0.1879856
UrbanPop	0.2781909	0.8728062
Rape	0.5434321	0.1673186

Example



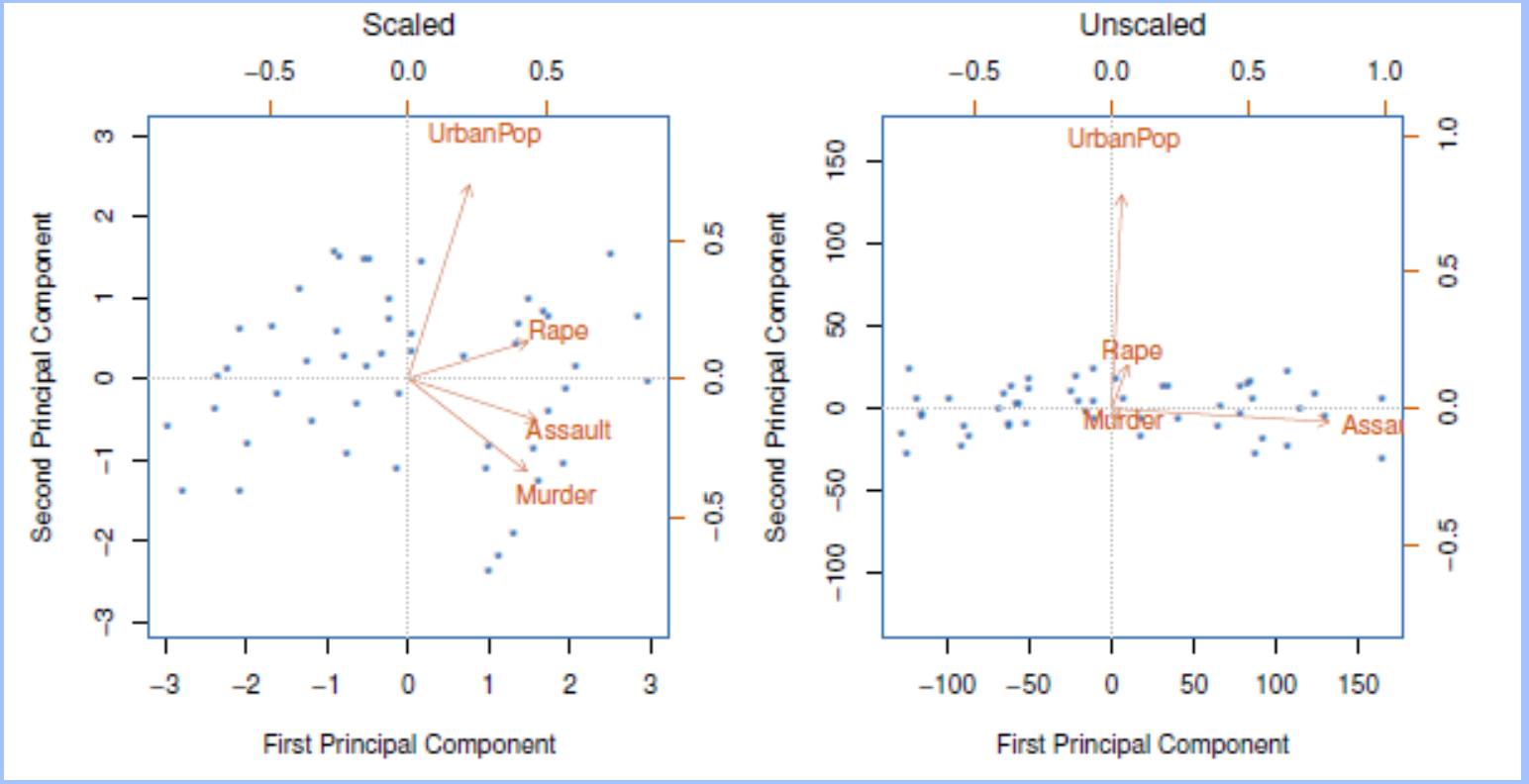

Second principal component
First principal component
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

Example

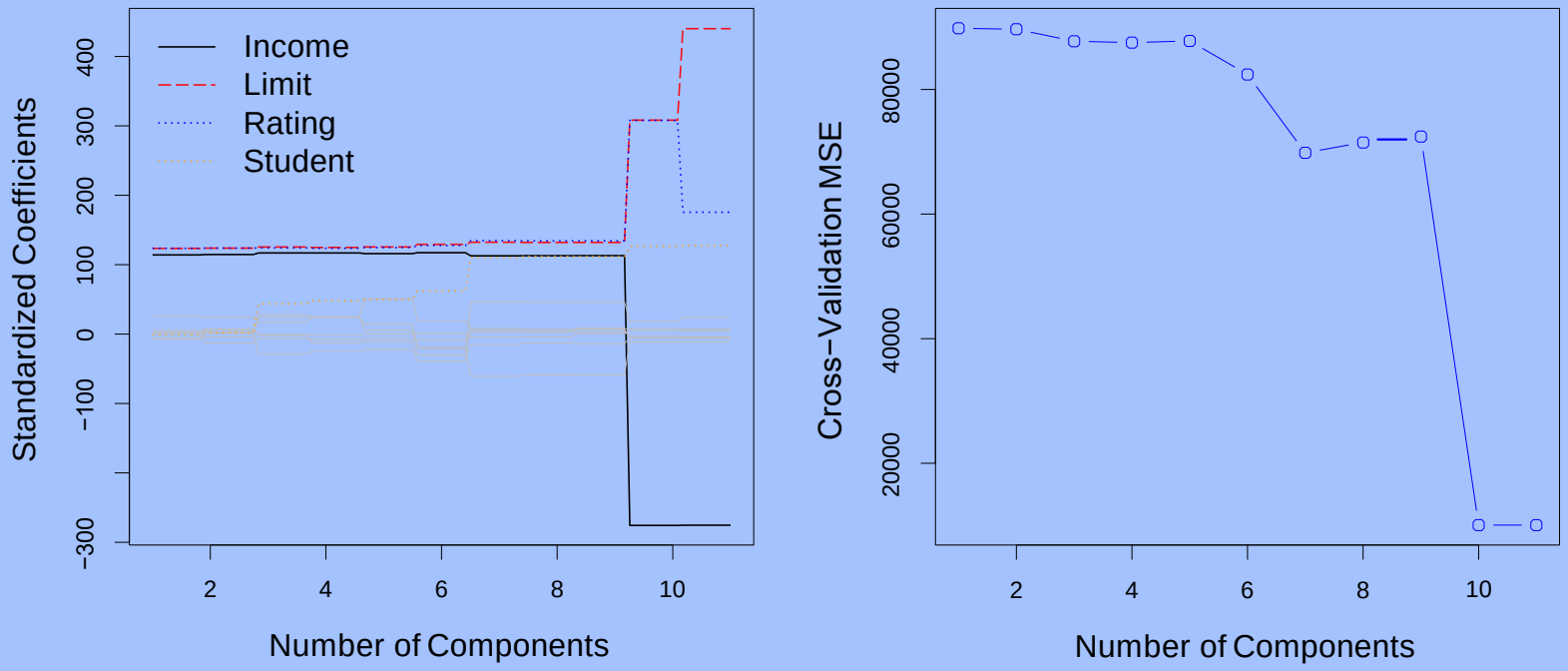
- PCA find the hyperplane closest to the observations. The first principal component loading vector has a very special property: it defines the line in p-dimensional space that is *closest* to the n observations (using average squared Euclidean distance as a measure of closeness)
- The notion of principal components as the dimensions that are closest to the n observations extends beyond just the first principal component.
- For instance, the first two principal components of a data set span the plane that is closest to the n observations, in terms of average squared Euclidean distance.

Scaling

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended.
- If they are in the same units, you might or might not scale the variables.



Choosing the number of directions *M*



Left: *PCR standardized coefficient estimates on the* Credit *data set for different values of M.* Right: *The* 10*-fold cross validation MSE obtained using PCR, as a function of M.*

Feature Selection or Hyper-parameter Tuning

- Which one comes first? Why