Lecture #6: Dimension Reduction

CS 109A, STAT 121A, AC 209A: Data Science

Pavlos Protopapas Kevin Rader Margo Levine
Rahul Dave

Lecture Outline

Regularization Review

More on Interaction Terms

High Dimensionality

Principal Components Analysis (PCA)

PCA for Regression (PCR)

PCA vs Variable Selection

Regularization Review

Overfitting and Regularization

Regularization: An Overview

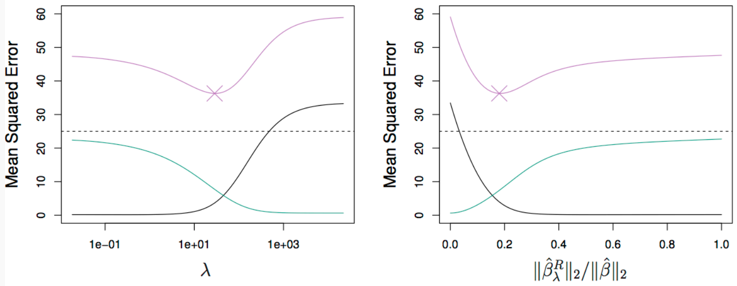
The idea of regularization revolves around modifying the loss function L; in particular, we add a ***regularization term*** that penalizes some specified properties of the model parameters

$$L_{reg}(\beta) = L(\beta) + \lambda R(\beta),$$

where λ is a scalar that gives the weight (or importance) of the regularization term.

Fitting the model using the modified loss function L_{reg} would result in model parameters with desirable properties (specified by R).

Bias-Variance Trade-off and Regularization



LASSO Regression

Since we wish to discourage extreme values in model parameter, we need to choose a regularization term that penalizes parameter magnitudes. For our loss function, we will again use MSE.

Together our regularized loss function is

$$L_{LASSO}(\beta) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i|^2 + \lambda \sum_{j=1}^{J} |\beta_j|.$$

Note that $\sum_{j=1}^{J} |\beta_j|$ is the ℓ_1 norm of the vector $\boldsymbol{\beta}$

$$\sum_{j=1}^{J} |\beta_j| = \|\boldsymbol{\beta}\|_1$$

Hence, we often say that L_{LASSO} is the loss function for ***ℓ_1 regularization***.

Finding model parameters $\boldsymbol{\beta}_{LASSO}$ that minimize the ℓ_1 regularized loss function is called ***LASSO regression***.

Ridge Regression

Alternatively, we can choose a regularization term that penalizes the squares of the parameter magnitudes.

Then, our regularized loss function is

$$L_{Ridge}(\beta) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i|^2 + \lambda \sum_{j=1}^{J} \beta_j^2.$$

Note that $\sum_{j=1}^{J} \beta_j^2$ is related to the ℓ_2 norm of $\boldsymbol{\beta}$

$$\sum_{j=1}^{J} \beta_j^2 = \|\boldsymbol{\beta}\|_2^2$$

Hence, we often say that L_{Ridge} is the loss function for ***ℓ_2 regularization***.

Finding model parameters $\boldsymbol{\beta}_{Ridge}$ that minimize the ℓ_2 regularized loss function is called ***ridge regression***.

Choosing λ

In both ridge and LASSO regression, we see that the larger our choice of the ***regularization parameter*** λ, the more heavily we penalize large values in β,

1. If λ is close to zero, we recover the MSE, i.e. ridge and LASSO regression is just ordinary regression.
2. If λ is sufficiently large, the MSE term in the regularized loss function will be insignificant and the regularization term will force β_{Ridge} and β_{LASSO} to be close to zero.

To avoid ad-hoc choices, we should select λ using cross-validation.

Ridge - Computational complexity

Solution to ridge regression:

$$\beta = (X^T X + \lambda I)^{-1} X^T Y$$

The solution of the Ridge/Lasso regression involves three steps.

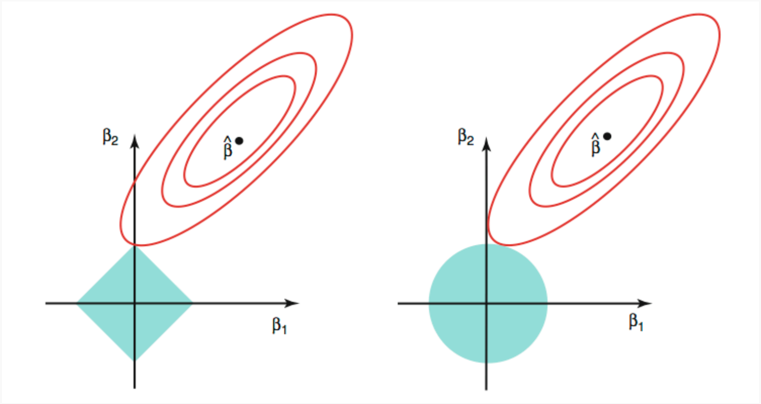
1. Select λ
2. Find the minimum of the ridge/Lasso regression loss function (using linear algebra) as with the multiple regression and record the R^2 **on the test set**.
3. Find the λ that gives the largest R^2

Variable Selection as Regularization

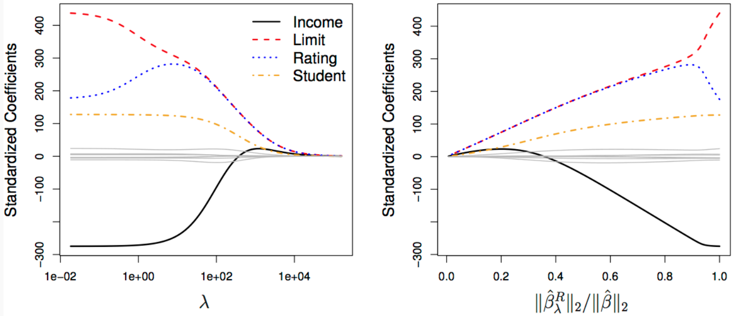
Since LASSO regression tend to produce zero estimates for a number of model parameters - we say that LASSO solutions are ***sparse*** - we consider LASSO to be a method for variable selection.

Many prefer using LASSO for variable selection (as well as for suppressing extreme parameter values) rather than stepwise selection, as LASSO avoids the statistic problems that arises in stepwise selection.

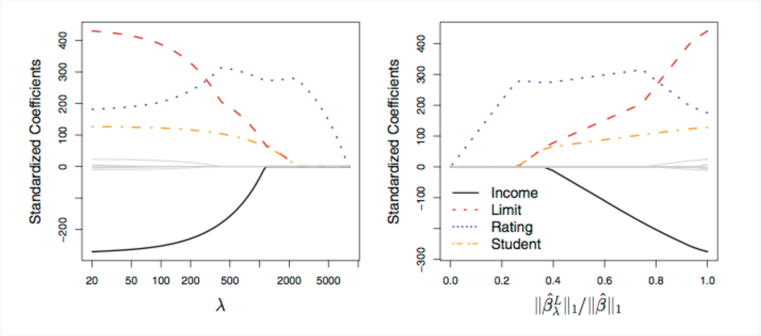
Lasso vs. Ridge



Ridge Regularization



Lasso Regularization



More on Interaction Terms

Interaction Terms: A Review

Recall that an interaction term between predictors X_1 and X_2 can be incorporated into a regression model by including the multiplicative (i.e. cross) term in the model, for example

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 (X_1 \cdot X_2) + \epsilon.$$

Example

Suppose X_1 is a binary predictor indicating whether a NYC ride pickup is a tax or an Uber, X_2 is the times of day of the pickup and Y is the length of the ride.

What is the interpretation of β_3?

Including Interaction Terms in Models

Recall that to avoid overfitting, we sometimes elect to exclude a number of terms in a linear model.

It is standard practice to always include the ***main effects*** in the model. That is, we always include the terms involving only one predictor, $\beta_1 X_1, \beta_2 X_2$ etc.

Question: Why are the ***main effects*** important?

Question: In what type of model would it make sense to include the interaction term without one of the main effects?

How Many Interaction Terms?

Example

Our NYC taxi and Uber dataset has 1.1 billion taxi trips and 19 million Uber trips. Each trip is described by $p = 23$ predictors (and 1 response variable). How many interaction terms are there?

- Two-way interactions: $\binom{p}{2} =$
- Three-way interactions: $\binom{p}{3} =$
- Etc

The total number of interaction terms (including main effects) is $\sum_{k=1}^{p} \binom{p}{k} = 2^p \approx 8.3$ million.

What is problem with building a model that includes all possible interaction terms?

Model Unidentifiability

Previously, we had been using samples of 100k observations from the dataset to build our models. If we include all possible interaction terms, our model will have 8.3 mil parameters. **We will not be able to uniquely determine 8.3 mil parameters with only 100k observations**. In this case, we call the model ***unidentifiable***.

In practice, we can:

- increase the number of observation
- consider only scientifically important interaction terms
- perform variable selection
- perform another ***dimensionality reduction*** technique like PCA

High Dimensionality

When Does High Dimensionality Occur?

The problem of high dimensionality can occur when the number of parameters exceeds or is close to the number of observations. This can occur when we consider lots of interaction terms, like in our previous example. But this can also happen when the number of main effects is high.

For example:

- When we are performing polynomial regression with a high degree and a large number of predictors.
- When the predictors are genomic markers in a computational biology problem.
- When the predictors are the counts of all English words appearing in a text.

A Framework For Dimensionality Reduction

One way to reduce the dimensions of the feature space is to create a new, smaller set of predictors by taking linear combinations of the original predictors.

We choose $Z_1, Z_2, \ldots, Z_m$, where $m < p$ and where each Z_i is a linear combination of the original p predictors

$$Z_i = \sum_{j=1}^{p} \phi_{ji} X_j$$

for fixed constants ϕ_{ji}. Then we can build a linear regression regression model using the new predictors

$$Y = \beta_0 + \beta_1 Z_1 + \ldots + \beta_m Z_m + \epsilon.$$

Notice that this model has a smaller number ($m < p$) of parameters.

A Framework For Dimensionality Reduction

A method of dimensionality reduction includes 2 steps:

1. Determine a optimal set of new predictors $Z_1, \ldots, Z_m$, for $m < p$.
2. Express each observation in the data in terms of these new predictors. The transformed data will have m columns rather than p.

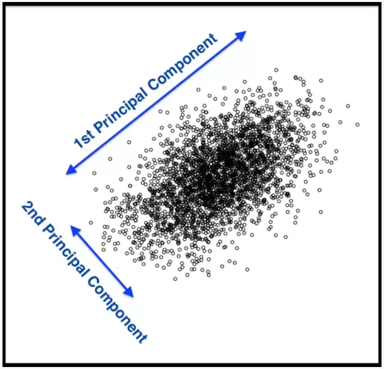
Thereafter, we can fit a model using the new predictors.

The method for determining the set of new predictors (what do we mean by an optimal predictors set) can differ according to application. We will explore a way to create new predictors that captures the variations in the observed data.

Principal Components Analysis (PCA)

Principal Components Analysis (PCA)

Principal Components Analysis (PCA) is a method to identify a new set of predictors, as linear combinations of the original ones, that captures the 'maximum amount' of variance in the observed data.



Principal Components Analysis (PCA)

Definition

Principal Components Analysis (PCA) produces a list of p ***principle components*** $(Z_1, \ldots, Z_p)$ such that

- Each Z_i is a linear combination of the original predictors, and it's vector norm is 1
- The Z_i's are pairwise orthogonal
- The Z_i's are ordered in decreasing order in the amount of captured observed variance.

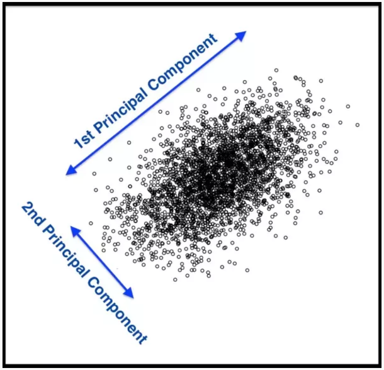
  That is, the observed data shows more variance in the direction of Z_1 than in the direction of Z_2.

To perform dimensionality reduction we select the top m principle components of PCA as our new predictors and express our observed data in terms of these predictors.

The Intuition Behind PCA

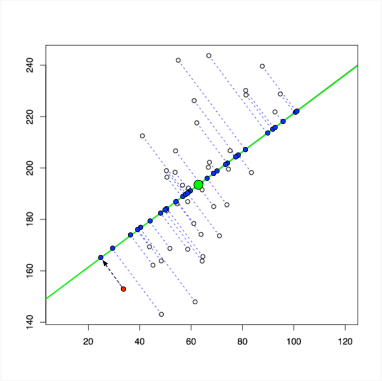
Top PCA components capture the most of amount of variation (interesting features) of the data.

Each component is a linear combination of the original predictors - we visualize them as vectors in the feature space.



The Intuition Behind PCA

Transforming our observed data means projecting our dataset onto the space defined by the top m PCA components, these components are our new predictors.



The Math behind PCA

PCA is a well-known result from linear algebra. Let **Z** be the $n \times p$ matrix consisting of columns $Z_1, ..., Z_p$ (the resulting PCA vectors), **X** be the $n \times p$ matrix of $X_1, ..., X_p$ of the original data variables (each re-centered to have mean zero, and without the intercept), and let **W** be the $p \times p$ matrix whose columns are the eigenvectors of the square matrix $\mathbf{X}^T\mathbf{X}$, then

$$\mathbf{Z}_{n\times p} = \mathbf{X}_{n\times p}\mathbf{W}_{p\times p}$$

Implementation of PCA using linear algebra

To implement PCA yourself using this linear algebra result, you can perform the following steps:

1. Subtract off the mean for each of your predictors (so they each have mean zero).
2. Calculate the eigenvectors of the $\mathbf{X}^T\mathbf{X}$ matrix and create the matrix with those columns, $\mathbf{W}$, in order from largest to smallest eigenvalue.
3. Use matrix multiplication to determine $\mathbf{Z} = \mathbf{X}\mathbf{W}$

Note: this is not efficient from a computational perspective. This can be sped up using Cholesky decomposition.
However, PCA is easy to perform in Python using the `decomposition.PCA` function in the `sklearn` package.

PCA for Regression (PCR)

Using PCA for Regression

PCA is easy to use in Python, so how do we then use it for regression modeling in a real-life problem?

If we use all p of the new Z_j, then we have not improved the dimensionality. Instead, we select the first M PCA variables, $Z_1, ..., Z_M$, to use as predictors in a regression model.

The choice of M is important and can vary from application to application. It depends on various things, like how collinear the predictors are, how truly related they are to the response, etc…

What would be the best way to check for a specified problem?

Train and Test!!!

PCA vs Variable Selection

Bibliography

1. Bolelli, L., Ertekin, S., and Giles, C. L. ***Topic and trend detection in text collections using latent dirichlet allocation***. In European Conference on Information Retrieval (2009), Springer, pp. 776-780.
2. Chen, W., Wang, Y., and Yang, S. ***Efficient influence maximization in social networks. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (2009)***, ACM, pp. 199-208.
3. Chong, W., Blei, D., and Li, F.-F. ***Simultaneous image classification and annotation. In Computer Vision and Pattern Recognition***, 2009. CVPR 2009. IEEE Conference on (2009), IEEE, pp. 1903-1910.
4. Du, L., Ren, L., Carin, L., and Dunson, D. B. ***A bayesian model for simultaneous image clustering, annotation and object segmentation***. In Advances in neural information processing systems (2009), pp. 486-494.
5. Elango, P. K., and Jayaraman, K. ***Clustering images using the latent dirichlet allocation model.***
6. Feng, Y., and Lapata, M. ***Topic models for image annotation and text illustration***. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (2010), Association for Computational Linguistics, pp. 831-839.
7. Hannah, L. A., and Wallach, H. M. ***Summarizing topics: From word lists to phrases.***
8. Lu, R., and Yang, Q. ***Trend analysis of news topics on twitter***. International Journal of Machine Learning and Computing 2, 3 (2012), 327.