Regression recap

Recall that in regression we are given training data

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$$

where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$

In linear regression we assume that we are trying to estimate a function of the form

$$h(\mathbf{x}) = \boldsymbol{\beta}^T \mathbf{x} + \beta_0$$

where $\boldsymbol{\beta} \in \mathbb{R}^d$, $\beta_0 \in \mathbb{R}$

Least squares regression: Select $\boldsymbol{\beta}, \beta_0$ to minimize

$$\widehat{R}_n(\boldsymbol{\beta}, \beta_0) := \sum_{i=1}^{n} \left(y_i - \boldsymbol{\beta}^T \mathbf{x}_i - \beta_0\right)^2$$

Least squares regression

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad \boldsymbol{X} = \begin{bmatrix} 1 & x_1(1) & \cdots & x_1(d) \\ 1 & x_2(1) & \cdots & x_2(d) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n(1) & \cdots & x_n(d) \end{bmatrix} \quad \boldsymbol{\theta} = \begin{bmatrix} \beta_0 \\ \beta(1) \\ \vdots \\ \beta(d) \end{bmatrix}$$

$$\widehat{R}_n(\boldsymbol{\theta}) = \sum_{i=1}^{n} (y_i - \boldsymbol{\beta}^T \mathbf{x}_i - \beta_0)^2 = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2$$

Minimizer given by

$$\widehat{\boldsymbol{\theta}} = \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

provided that $\boldsymbol{X}^T\boldsymbol{X}$ is ***nonsingular***

Regularization and regression

Overfitting occurs as $d \to n$

In this regime, we have *too many degrees of freedom*, and it becomes likely that will be (approximately) singular $\boldsymbol{X}^T\boldsymbol{X}$

Idea: penalize candidate solutions that are “too big”

One candidate regularizer: $r(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_2^2$$

$\lambda > 0$ is a “tuning parameter” that controls the tradeoff between fit and complexity

Do we have any other options?

What do we do if $d > n$?

Sometimes

- our data is extremely high-dimensional
- the training data is very expensive to acquire/label

In such settings, we have some additional strategies:

Dimensionality reduction

Feature selection

Dimensionality reduction

We observe data $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

The goal of ***dimensionality reduction*** is to transform these inputs to new variables

$$\mathbf{x}_i \to \mathbf{z}_i \in \mathbb{R}^k$$

where $k \ll d$ in such a way that ***preserves information***

Dimensionality reductions serves two main purposes:

- Helps (many) algorithms to be more computationally efficient
- Helps prevent overfitting (a form of regularization), especially when $n \le d$

Curse of Dimensionality

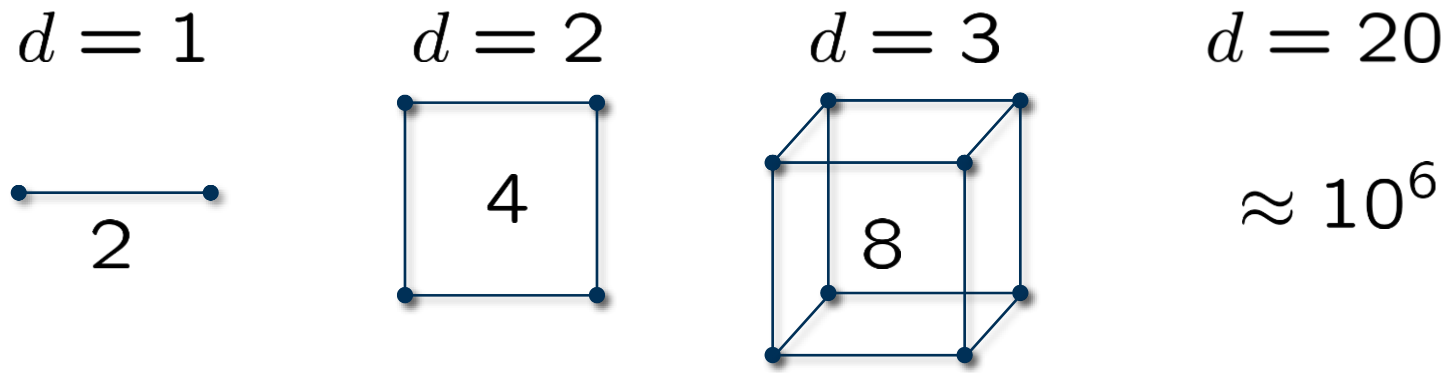
Curse of dimensionality

As the dimensionality of our feature space grows, the volume of the space increases...

A lot...

In learning, this often translates to requiring exponentially more data in order for the results to be reliable

Example: With binary features, how much data do we need to have at least one example of every possible combination of features?



Principal component analysis (PCA)

- Unsupervised
- Linear
- Loss criteria: Sum of squared errors

The idea behind PCA is to find an approximation

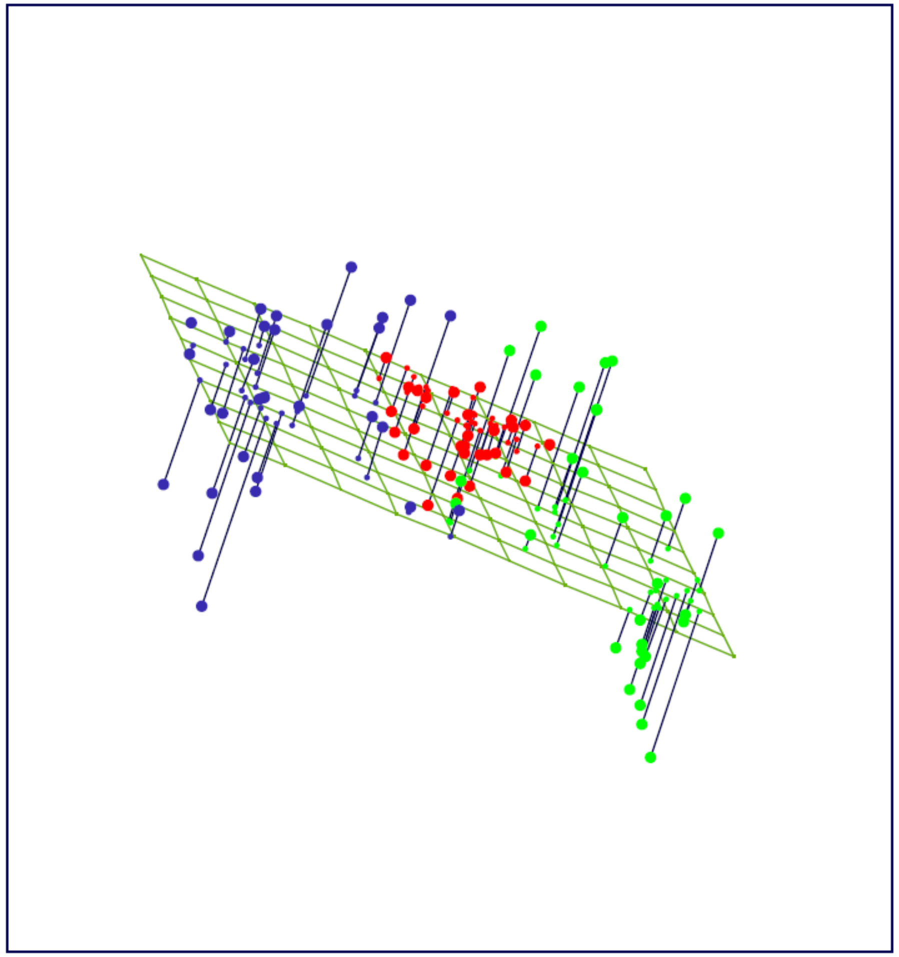
$$\mathbf{x}_i \approx \boldsymbol{\mu} + \mathbf{A}\mathbf{z}_i$$

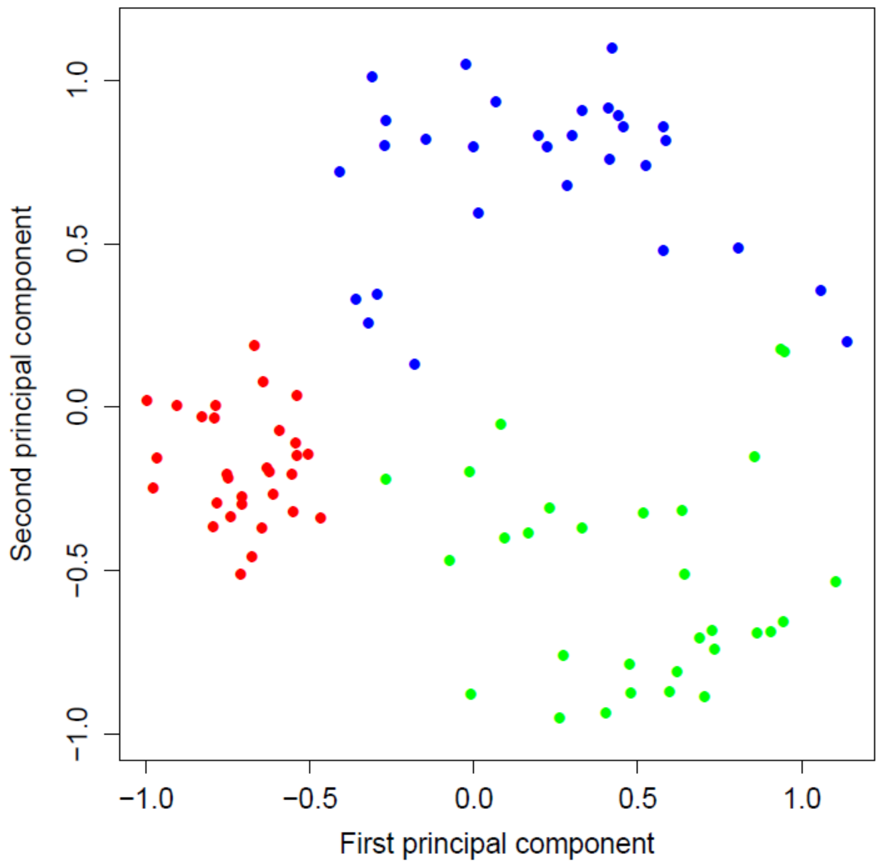
where

- $\boldsymbol{\mu} \in \mathbb{R}^d$
- $\mathbf{A} \in \mathbb{R}^{d \times k}$ with orthonormal columns
- $\mathbf{z}_i \in \mathbb{R}^k$

Example

From Chapter 14 of Hastie, Tibshirani, and Friedman





Derivation of PCA

Mathematically, we can define $\boldsymbol{\mu}$, $\mathbf{A}$ and $\mathbf{z}_1, \ldots, \mathbf{z}_n$ as the solution to

$$\min_{\boldsymbol{\mu}, \mathbf{A}, \{\mathbf{z}_i\}} \quad \sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu} - \mathbf{A}\mathbf{z}_i\|_2^2$$

The hard part of this problem is finding $\mathbf{A}$

Given $\mathbf{A}$, it is relatively easy to show that

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

$$\mathbf{z}_i = \mathbf{A}^T(\mathbf{x}_i - \boldsymbol{\mu})$$

Determining $\mathbf{z}_i$

Suppose $\boldsymbol{\mu}, \mathbf{A}$ are fixed. We wish to minimize

$$\sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu} - \mathbf{A}\mathbf{z}_i\|_2^2$$

Claim: We must have

$$\begin{aligned} \mathbf{z}_i &= (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T(\mathbf{x}_i - \boldsymbol{\mu}) \\ &= \mathbf{A}^T(\mathbf{x}_i - \boldsymbol{\mu}) \end{aligned}$$

Why?

Determining $\mathbf{z}_i$ is just standard least-squares regression

Determining $\boldsymbol{\mu}$

Setting $\mathbf{z}_i = \mathbf{A}^T(\mathbf{x}_i - \boldsymbol{\mu})$ and still supposing $\mathbf{A}$ is fixed, our problem reduces to minimizing

$$\begin{aligned}\sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu} - \mathbf{A}\mathbf{A}^T(\mathbf{x}_i - \boldsymbol{\mu})\|_2^2 \\ = \sum_{i=1}^{n} \|(\mathbf{I} - \mathbf{A}\mathbf{A}^T)(\mathbf{x}_i - \boldsymbol{\mu})\|_2^2 \\ = \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})^T \underbrace{(\mathbf{I} - \mathbf{A}\mathbf{A}^T)^T(\mathbf{I} - \mathbf{A}\mathbf{A}^T)}_{\mathbf{B}}(\mathbf{x}_i - \boldsymbol{\mu})\end{aligned}$$

Determining $\boldsymbol{\mu}$

Taking the gradient with respect to $\boldsymbol{\mu}$ and setting this equal to zero, we obtain

$$-2\sum_{i=1}^{n} \mathrm{B}(\mathbf{x}_i - \boldsymbol{\mu}) = 0$$

$$\Rightarrow \quad -2\mathrm{B}\left(\sum_{i=1}^{n} \mathbf{x}_i - n\boldsymbol{\mu}\right) = 0$$

The choice of $\boldsymbol{\mu}$ is not unique, but the easy (and standard) way to ensure this equality holds is to set

$$\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$$

Determining $\mathbf{A}$

It remains to minimize

$$\sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu} - \mathbf{A}\mathbf{A}^T(\mathbf{x}_i - \boldsymbol{\mu})\|_2^2$$

with respect to $\mathbf{A}$

For convenience, we will assume that $\boldsymbol{\mu} = \mathbf{0}$, since otherwise we could just substitute $\widetilde{\mathbf{x}}_i = \mathbf{x}_i - \boldsymbol{\mu}$

In this case the problem reduces to minimizing

$$\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{A}\mathbf{A}^T\mathbf{x}_i\|_2^2$$

Determining $\mathbf{A}$

Expanding this out, we obtain

$$\begin{aligned}\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{A}\mathbf{A}^T\mathbf{x}_i\|_2^2 &= \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{A}\mathbf{A}^T\mathbf{x}_i)^T(\mathbf{x}_i - \mathbf{A}\mathbf{A}^T\mathbf{x}_i) \\ &= \sum_{i=1}^{n} \mathbf{x}_i^T\mathbf{x}_i - 2\mathbf{x}_i^T\mathbf{A}\mathbf{A}^T\mathbf{x}_i + \mathbf{x}_i^T\mathbf{A}\underbrace{\mathbf{A}^T\mathbf{A}}_{\mathbf{A}^T\mathbf{A}=\mathbf{I}}\mathbf{A}^T\mathbf{x}_i \\ &= \sum_{i=1}^{n} \mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_i^T\mathbf{A}\mathbf{A}^T\mathbf{x}_i\end{aligned}$$

Thus, we can instead focus on maximizing

$$\sum_{i=1}^{n} \mathbf{x}_i^T\mathbf{A}\mathbf{A}^T\mathbf{x}_i$$

Determining $\mathbf{A}$

Note that for any vector $\mathbf{v}$, we have $\|\mathbf{v}\|_2^2 = \text{trace}(\mathbf{v}\mathbf{v}^T)$

Thus, we can write

$$\begin{aligned}\sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{A}\mathbf{A}^T \mathbf{x}_i &= \sum_{i=1}^{n} \|\mathbf{A}^T \mathbf{x}_i\|_2^2 \\ &= \sum_{i=1}^{n} \text{trace}(\mathbf{A}^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{A}) \\ &= \text{trace}\left(\mathbf{A}^T (\textstyle\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T) \mathbf{A}\right) \\ &= \text{trace}\left(\mathbf{A}^T \mathbf{S} \mathbf{A}\right)\end{aligned}$$

$\mathbf{S} = \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$ is a scaled version of the empirical covariance matrix, sometimes called the ***scatter*** matrix

Determining $\mathbf{A}$

The problem of determining $\mathbf{A}$ reduces to the optimization problem

$$\max_{\mathbf{A}} \ \text{trace}(\mathbf{A}^T\mathbf{S}\mathbf{A})$$
$$\text{s.t.} \ \mathbf{A}^T\mathbf{A} = \mathbf{I}$$

Analytically deriving the optimal $\mathbf{A}$ is not too hard, but is a bit more involved than you might initially expect
(especially if you already know the answer)

We will provide justification for the solution for the $k = 1$ case – the general case is proven in the supplementary notes

One-dimensional example

Consider the optimization problem

$$\max_{\mathbf{a}} \ \mathbf{a}^T \mathbf{S} \mathbf{a}$$
$$\text{s.t. } \mathbf{a}^T \mathbf{a} = 1$$

Form the Lagrangian $\mathcal{L}(\mathbf{a}) = \mathbf{a}^T \mathbf{S} \mathbf{a} + \lambda(\mathbf{a}^T \mathbf{a} - 1)$

Take the gradient and set it equal to zero

$$\mathbf{S}\mathbf{a} + \lambda \mathbf{a} = 0$$

$\Rightarrow$ $\mathbf{a}$ must be an eigenvector of $\mathbf{S}$

Take $\mathbf{a}$ to be the eigenvector of $\mathbf{S}$ corresponding to the maximal eigenvalue

The general case

For general values of k, the solution is obtained by computing the eigendecomposition of $\mathbf{S}$:

$$\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$$

where $\mathbf{U}$ is an orthonormal matrix with columns $\mathbf{u}_1, \ldots, \mathbf{u}_d$ and

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_d \end{bmatrix}$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d \geq 0$

The general case

The optimal choice of $\mathbf{A}$ in this case is given by

$$\mathbf{A} = [\mathbf{u}_1, \ldots, \mathbf{u}_k]$$

i.e., take the top k eigenvectors of $\mathbf{S}$

Terminology

- principal component transform: $\mathbf{x} \rightarrow \mathbf{z} = \mathbf{A}^T(\mathbf{x} - \boldsymbol{\mu})$
- j^{th} principal component: $z(j) = \mathbf{u}_j^T(\mathbf{x} - \boldsymbol{\mu})$
- j^{th} principal eigenvector: $\mathbf{u}_j$

Connection to SVD

Recall the singular value decomposition (SVD)

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

If $\mathbf{X}$ is a real $d \times n$ matrix

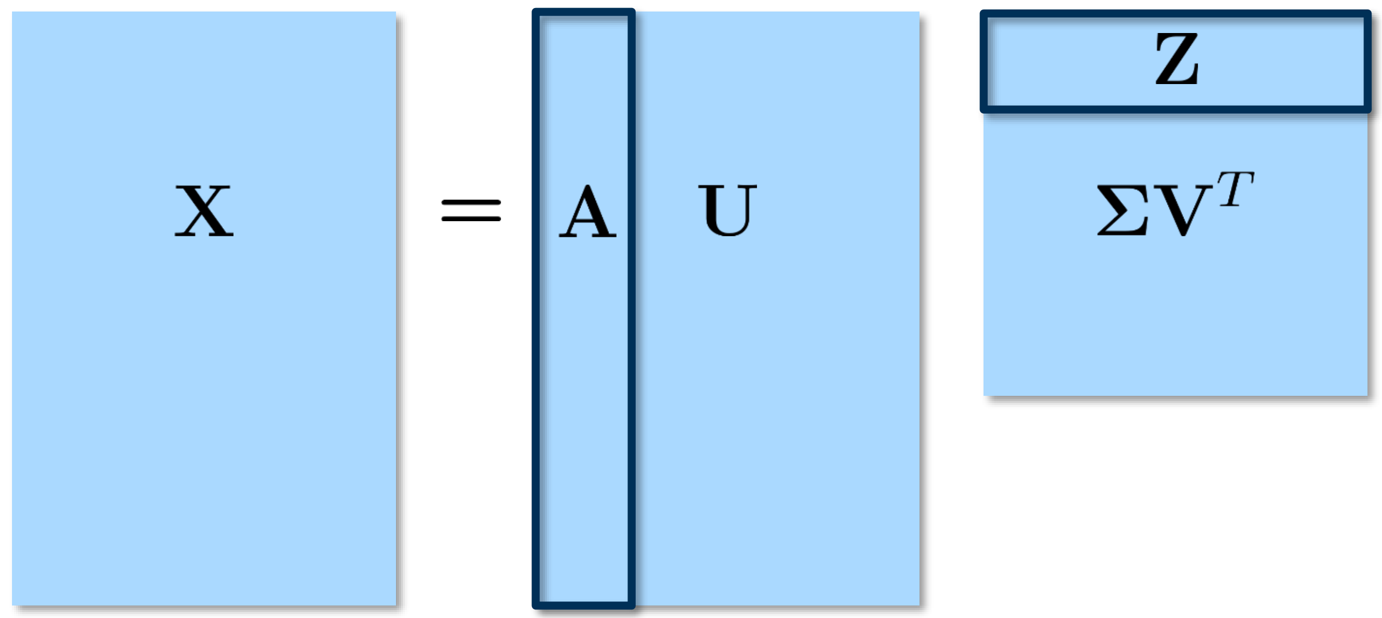
- $\mathbf{U}$ is a $d \times d$ orthonormal matrix
- $\mathbf{V}$ is an $n \times n$ orthonormal matrix
- $\mathbf{\Sigma} = \text{diag}(\sigma_1, \ldots, \sigma_r)$ is a $d \times n$ diagonal matrix where $r \leq \min(d, n)$ and

$$\begin{aligned} \sigma_i &= i^{\text{th}} \text{ singular value} \\ &= \text{square root of } i^{\text{th}} \text{ eigenvalue of } \mathbf{X}\mathbf{X}^T \end{aligned}$$

The principal eigenvectors are the first k columns of $\mathbf{U}$ when the columns of $\mathbf{X}$ are filled with $\widetilde{\mathbf{x}}_i = \mathbf{x}_i - \boldsymbol{\mu}$

Visual interpretation

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$



Practical matters

It is customary to ***center*** and ***scale*** a data set so that it has zero mean and unit variance along each feature

This puts all features on an "equal playing field"

These steps may be omitted when

- The data are known to be zero mean
- The data are known to have comparable units of measurement

To select k, we typically choose it to be large enough so that

$$\sum_{i=1}^{n} \|\mathbf{x}_i - \boldsymbol{\mu} - \mathbf{A}\mathbf{z}_i\|_2^2 = n(\lambda_{k+1} + \ldots + \lambda_d)$$

is sufficiently small

When to use PCA

- When the data form a single “point cloud” in space
- When the data are approximately Gaussian, or some other “elliptical” distribution
- When low-rank subspaces capture most of the variation

Later in the course we will learn about several alternative approaches to dimensionality reductions when these assumptions fail to hold

Feature selection

PCA and similar dimensionality reduction strategies can be very powerful but also have some significant drawbacks:

- Notice that they are completely ***unsupervised***, meaning that they do not use y to aid in constructing a good set of features
- The learned features are often extremely difficult to ***interpret***

Can we instead simply select a subset of the existing features?

The LASSO

LASSO

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{y} - \boldsymbol{X\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1$$

Can also be stated in a constrained form

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{y} - \boldsymbol{X\theta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\theta}\|_1 \leq \tau$$

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|_1 \quad \text{s.t.} \quad \|\boldsymbol{y} - \boldsymbol{X\theta}\|_2^2 \leq \sigma$$

For Tikhonov, we have a closed form solution, but LASSO ***requires*** solving an optimization problem using numerical methods

Note: Just like in ridge regression, in practice we may just want to penalize the elements of $\boldsymbol{\beta}$ (not β_0)

Sparsity and the LASSO

One can show (see supplemental notes) that if we have a data set of size n, then the solution to the LASSO $\widehat{\boldsymbol{\theta}}$ will have at most n nonzeros (for any possible dataset / $\boldsymbol{X}$)

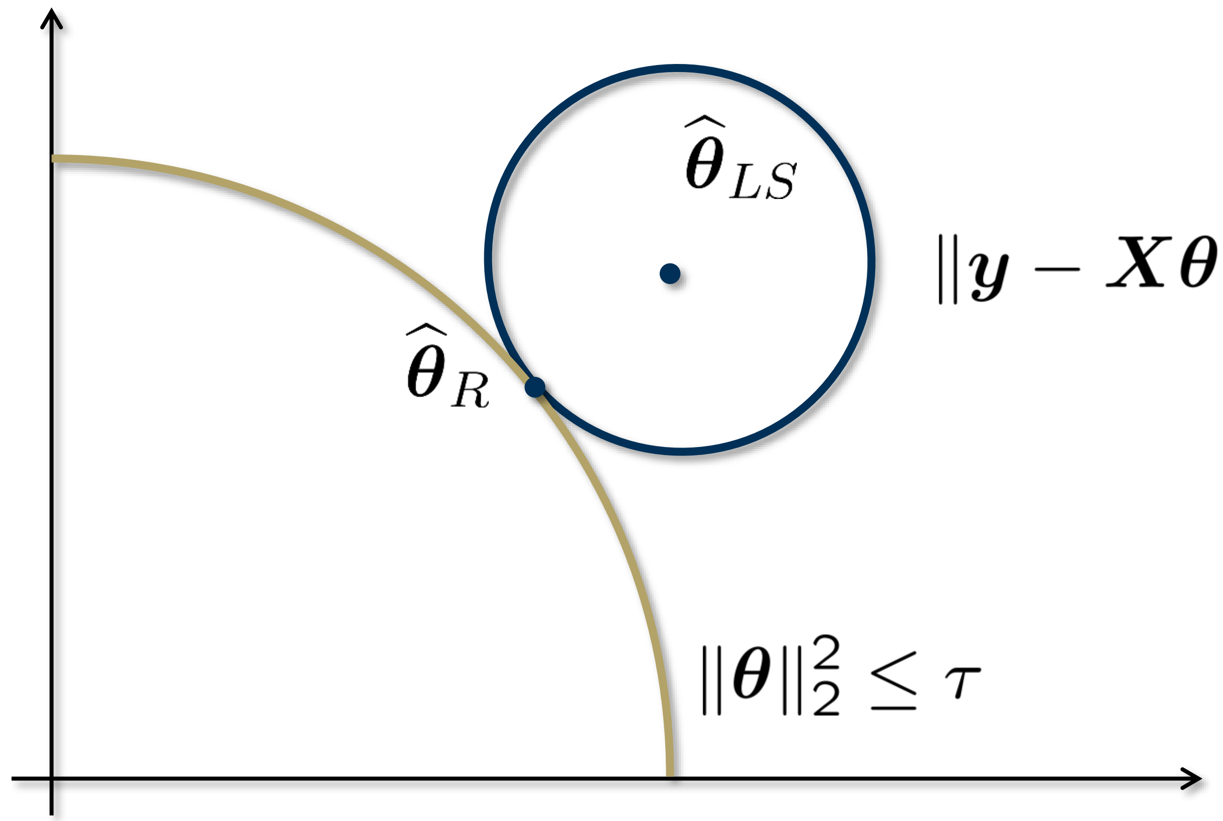
This is a nice property when $n \ll d$, since in this setting we are ***very*** susceptible to overfitting

- fewer observations than unknowns
- $\boldsymbol{X}$ has nontrivial nullspace
- we can achieve $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\theta}$, with infinitely many different choices of $\boldsymbol{\theta}$ and no obvious way to know which one is best
- limiting the number of nonzeros addresses this problem

In practice, the number of nonzeros is usually ***much smaller*** than n

Tikhonov versus least squares

Assume $\boldsymbol{\Gamma} = \boldsymbol{I}$ and that $\boldsymbol{X}$ has orthonormal columns

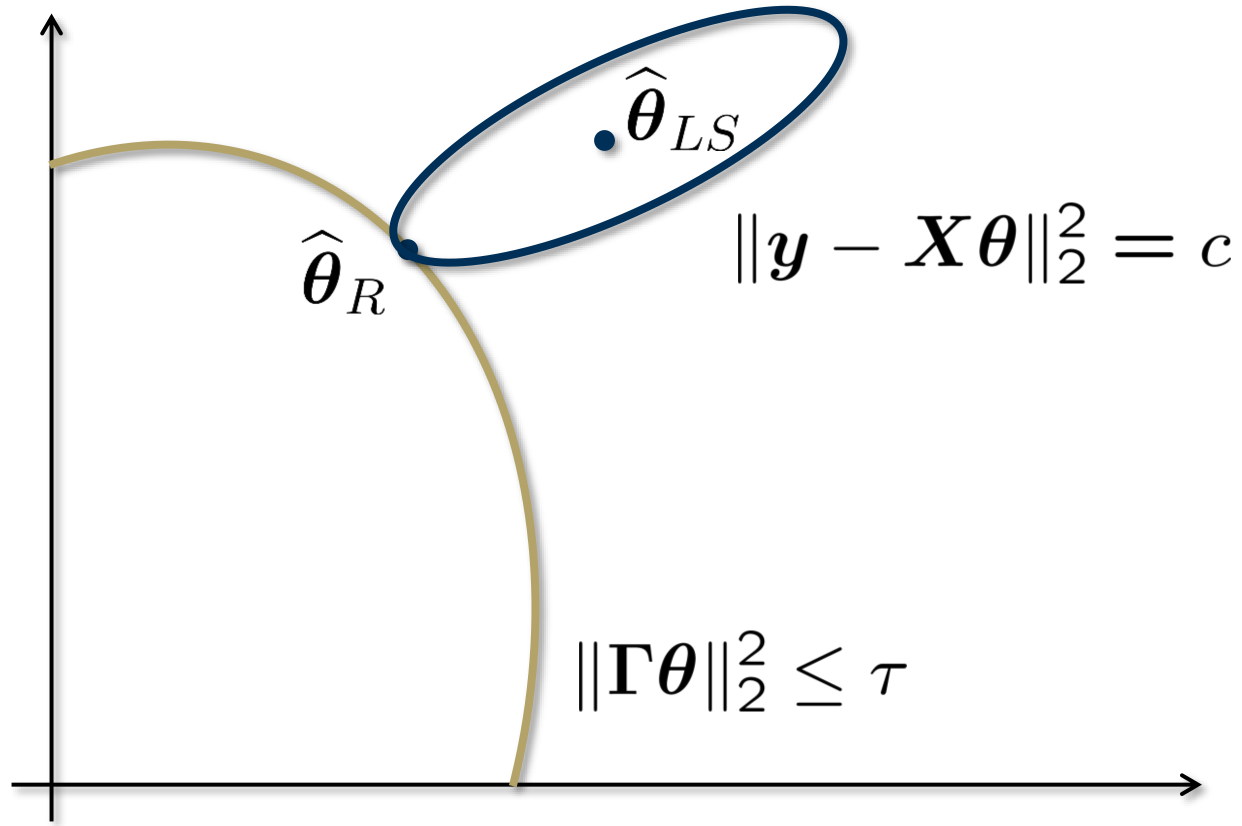


$\widehat{\boldsymbol{\theta}}_{LS}$
$\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2 = c$
$\widehat{\boldsymbol{\theta}}_{R}$
$\|\boldsymbol{\theta}\|_2^2 \leq \tau$

Tikhonov regularization is equivalent to shrinking the least squares solution towards the origin

Tikhonov versus least squares

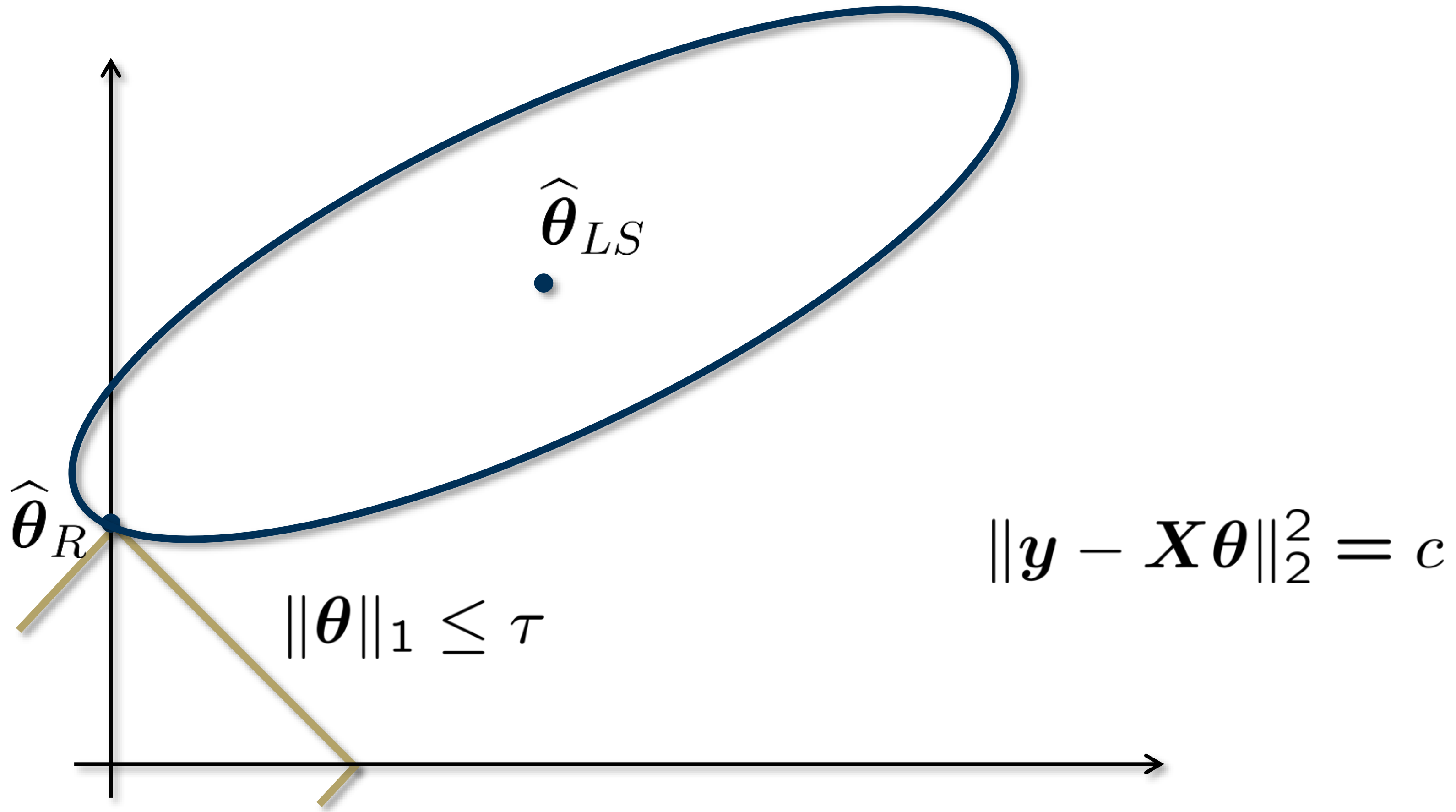
In general, we have this picture


$\widehat{\boldsymbol{\theta}}_{LS}$
$\widehat{\boldsymbol{\theta}}_{R}$
$\|\boldsymbol{y} - \boldsymbol{X\theta}\|_2^2 = c$
$\|\boldsymbol{\Gamma\theta}\|_2^2 \leq \tau$

Tikhonov regularization still shrinking the least squares solution, but weighting different dimensions more heavily

Lasso versus least squares

For the LASSO we tend to get something like this…



$\widehat{\boldsymbol{\theta}}_{LS}$
$\widehat{\boldsymbol{\theta}}_R$
$\|\boldsymbol{\theta}\|_1 \leq \tau$
$\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}\|_2^2 = c$

LASSO still shrinking the least squares solution towards the origin, but now in a way that promotes sparsity (especially in high-dimensions)

A general approach to regression

Least squares, ridge regression, and the LASSO call all be viewed as particular instances of the following general approach to regression

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \; L(\boldsymbol{\theta}) + \lambda r(\boldsymbol{\theta})$$

- $L(\boldsymbol{\theta})$, often called the ***loss function***, enforces data fidelity

$$h_{\boldsymbol{\theta}}(\mathbf{x}_i) \approx y_i$$

- $r(\boldsymbol{\theta})$ is a ***regularizer*** which serves to quantify the "complexity" of $\boldsymbol{\theta}$

We have seen some examples of regularizers, what about other loss functions?

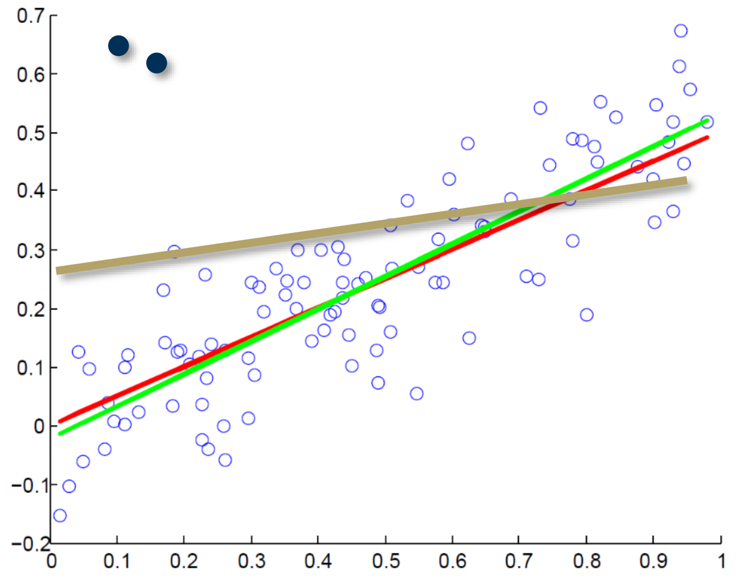
Outliers in regression

The squared error loss function is sensitive to ***outliers***

If $h(\mathbf{x}_i) - y_i$ is small, then $(h(\mathbf{x}_i) - y_i)^2$ is not too large

But if $h(\mathbf{x}_i) - y_i$ is big, then $(h(\mathbf{x}_i) - y_i)^2$ is ***really*** big

Normally this is not a bad property – we want to penalize big errors – but this can make us very sensitive to large outliers



Robust regression

What else could we do aside from least squares?

Mean absolute error

$$L_{AE}(r) = |r|$$

Huber loss

$$L_H(r) = \begin{cases} \frac{1}{2}r^2 & \text{if } |r| \leq c \\ c|r| - \frac{c^2}{2} & \text{if } |r| > c \end{cases}$$

ϵ-insensitive loss

$$L_\epsilon(r) = \begin{cases} 0 & \text{if } |r| \leq c \\ |r| - \epsilon & \text{if } |r| > \epsilon \end{cases}$$

Regularized robust regression

Suppose we combine this loss with an ℓ_2 regularizer

$$\widehat{\boldsymbol{\beta}}, \beta_0 = \underset{(\boldsymbol{\beta},\beta_0)}{\arg\min} \sum_{i=1}^{n} L_\epsilon(y_i - (\boldsymbol{\beta}^T \mathbf{x}_i + \beta_0)) + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$$

Note that the ϵ-insensitive loss has no penalty as long as your prediction is within a “margin” of ϵ

We will encounter something very similar to this in the context of classification in a few weeks…