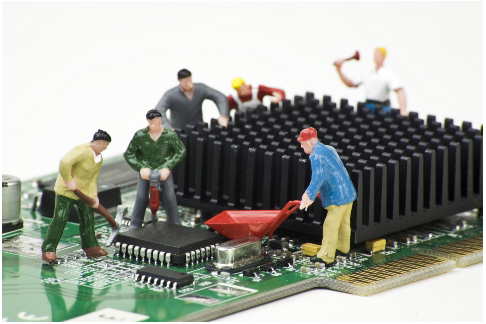
Introduction, Notation, and Overview

-Statistical Machine Learning-

Lecturer: Darren Homrighausen, PhD



Statistical learning and data mining is about..

- discovering structure in data
- making predictions about unknown quantities

Class Overview

Practically speaking, this means we seek to:

- find relationships between a group of explanatory and response variables that provides good predictive performance
- reduce the size of the group of variables for scientific, statistical, or computational purposes

and, perhaps most importantly..

Knowing the techniques, how they work, when they apply, and how to implement them

Class Outline

Over the next semester we will address:

1. High dimensional classification and regression
2. Nonparametric methods
3. Clustering
4. Graphical models

This course will emphasize methods and applications. However, theory will be presented to illustrate some important points/techniques.

References:

Main references:

- *The Elements of Statistical Learning* Hastie, Tibshirani, Friedman
- *Weak Convergence and Empirical Processes* Van der Vaart, Wellner

Secondary references:

- *Statistics for High-Dimensional Data* Bühlmann, Van de Geer
- *Generic Chaining* Talagrand
- *Introduction to Nonparametric Regression* Tsybakov
- *Convex Optimization* Boyd, Vandenberghe

High level overview

So many possibilities

There are many ways this course can go:

1. Theory: How to prove results about methods and understand statistical/quantitative papers
2. Methodology: Discuss the motivation and ‘inner workings’
3. Applications: Go over implementation, use of software, data problems
4. Computation: Go through algorithms, parallelization, data structures...

So many possibilities

There are many ways this course can go:

1. Theory: How to prove results about methods and understand statistical/quantitative papers
2. Methodology: Discuss the motivation and ‘inner workings’
3. Applications: Go over implementation, use of software, data problems
4. Computation: Go through algorithms, parallelization, data structures...

Task: Assign a percent to each category representing your interests.

Statistical learning terminology

Introduction

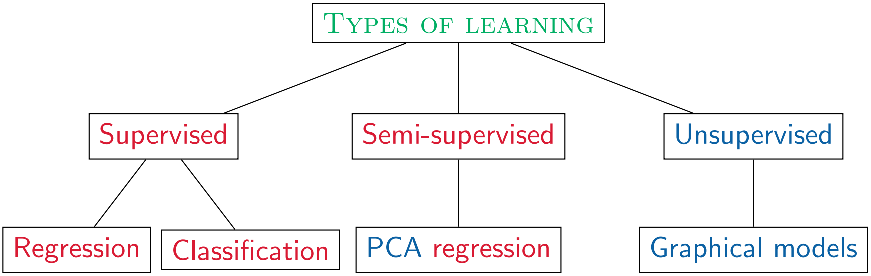
Statistical Machine Learning (SML) is statistics with a focus on prediction, scalability, and high dimensional problems

Regression: predict $Y \in \mathbb{R}$ from covariates or features X

Classification: predict $Y \in \{0, 1\}$ from covariates or features X

Finding structure:

- Finding groups or clusters in the data
- Dimension reduction
- Graphical models (conditional independence structure)



Some comments:

Comparing to the response Y gives a natural notion of prediction accuracy

Much more heuristic, unclear what a good solution would be. We'll return to this later in the semester.

Three main themes

Convexity

Convex problems can be solved efficiently. If necessary, we try to approximate nonconvex problems with convex ones

Sparsity

Many interesting problems are high dimensional

(the number of covariates, p, is large compared to the number of observations, n)

Assumptions

What assumptions do you need to make to motivate the method or guarantee some property?

Supervised Methods

(We will return to unsupervised methods later in the class)

The set-up

We observe n pairs of data $(X_1^\top, Y_1)^\top, \ldots, (X_n^\top, Y_n)^\top$

Let[1] $Z_i^\top = (X_i^\top, Y_i) \in \mathbb{R}^p \times \mathbb{R}$

We'll refer to the training data as $\mathcal{D} = \{Z_1, \ldots, Z_n\}$

Call Y_i the response, while X_i is the feature or covariate (vector)

Example: Y_i is whether a threat is detected in an image and the X_{ij} is the value at the j^{th} pixel of an image (p might be $1024^2 = 1048576$)

[1]These transposes get tiredsome. We'll get a bit sloppy and drop them selectively in what follows.

The set-up

We use the training data $\mathcal{D}$ to train an algorithm, producing a function $\hat{f} : \mathbb{R}^p \rightarrow \mathbb{R}$

Goal: Given a new $X \in \mathbb{R}^p$, we want to form predictions

$$\hat{f}(X) = \hat{Y}(X, \mathcal{D}) = \hat{Y}$$

Such that $\hat{Y}$ is a good prediction of Y, the unobserved response

Risk, Bayes, bias, variance, and approximation

Loss functions and risk

What determines good?

Define a function[2] $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ such that smaller values of ℓ indicate better performance

Two important examples:

- $\ell(\hat{f}(X), Y) = (\hat{f}(X) - Y)^2$ (regression, square-error)
- $\ell(\hat{f}(X), Y) = \mathbf{1}(\hat{f}(X) \neq Y)$ (classification, 0-1)

These expressions are both random variables. This leads us to define the (prediction or generalization) risk of a procedure $\hat{f}$ to be

$$R(\hat{f}) = \mathbb{E}\ell(\hat{f}(X), Y) = \mathbb{P}\ell(\hat{f}(X), Y) = \mathbb{P}\ell_{\hat{f}} = \int \ell(\hat{f}(X), Y) d\mathbb{P},$$

where $\mathbb{P}$ is the measure induced by $Z = (X, Y)$.

[2] This is the loss for prediction. Other tasks may have a different domain.

Risky (and lossy) business

The theoretical basis for prediction/estimation is rooted in Statistical Decision Theory.

Any distance function[3] could serve for the loss function ℓ

We can write the risk as

$$R(f) = \mathbb{E}\ell(f(X), Y) = \mathbb{E}_X \mathbb{E}_{Y|X}\ell(f(X), Y)$$

(The tower property of conditional expectation)

The measureable function f_* such that this pointwise relation holds:

$$f_*(x) = \underset{c}{\operatorname{argmin}}\, \mathbb{E}_{Y|X=x}\ell(c, Y)$$

is known as the Bayes rule with respect to the loss function ℓ.

[3] Or, for that matter, topology

An example: Squared-error loss

If the function $\ell(f(X), Y) = (f(X) - Y)^2$, then

$$f_*(x) = \mathbb{E}[Y|X = x].$$

This is known as the regression function; that is, the conditional expectation of Y given X.

(This is the Bayes rule with respect to the squared error loss function.)

This gives rise to the model

$$Y = f_*(X) + \epsilon$$

where ϵ is some mean zero fluctuation

An example: Zero-one loss

Instead, let $Y \in \mathcal{G} = \{1, \ldots, G\}$

As Y takes only a few values, zero-one prediction risk is natural

$$\ell(f(X), Y) = \mathbf{1}(Y \neq f(X)) \implies R(f) = \mathbb{E}[\ell(f(X), Y)] = ?$$

$(? = \qquad\qquad\qquad)$

Goal: Find an f such that $f(X) = Y$ as often as possible

Under this loss, we have

$$f_*(X) = \operatorname*{argmin}_{g \in \mathcal{G}} [1 - \mathbb{P}(Y = g|X)] = \arg\max_{g \in \mathcal{G}} \mathbb{P}(Y = g|X)$$

(Interpretation: The Bayes rule for classification with this loss is to pick the class that maximizes the conditional probability of Y being that class)

An example: Zero-one loss

Instead, let $Y \in \mathcal{G} = \{1, \ldots, G\}$

As Y takes only a few values, zero-one prediction risk is natural

$$\ell(f(X), Y) = \mathbf{1}(Y \neq f(X)) \implies R(f) = \mathbb{E}[\ell(f(X), Y)] = ?$$

$(? = \mathbb{P}(f(X) \neq Y))$

Goal: Find an f such that $f(X) = Y$ as often as possible

Under this loss, we have

$$f_*(X) = \operatorname*{argmin}_{g \in \mathcal{G}} [1 - \mathbb{P}(Y = g|X)] = \arg\max_{g \in \mathcal{G}} \mathbb{P}(Y = g|X)$$

(Interpretation: The Bayes rule for classification with this loss is to pick the class that maximizes the conditional probability of Y being that class)

Training error and risk estimation

Of course, we don't know $\mathbb{P}$...

We need to estimate it!

Perhaps the most intuitive estimate of the measure $\mathbb{P}$ is

$$\hat{\mathbb{P}} = \frac{1}{n}\sum_{i=1}^{n}\delta_{Z_i},$$

where δ_x is a (probability) measure that puts mass 1 at x.

This is known as the empirical measure of $\mathcal{D}$

Just like $\mathbb{P}f(X) = \int f(X)d\mathbb{P}$, we can write

$$\hat{\mathbb{P}}f(X) = \int f(X)d\hat{\mathbb{P}} = \frac{1}{n}\sum_{i=1}^{n} f(Z_i).$$

Example: Maximum likelihood estimation

Suppose that we are interested in estimating a parameter vector μ

We specify a likelihood $L_\mu(Z)$, such as by stating $Z \sim N(\mu, \Sigma) \in \mathbb{R}^p$ and writing

$$L_\mu(Z) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{-(Z-\mu)^\top \Sigma^{-1} (Z-\mu)/2}.$$

Define $\ell_\mu = \log L_\mu$. Then the maximum likelihood estimator is

$$\arg\max_\mu \hat{\mathbb{P}} \ell_\mu$$

Linear Algebra

Background

- We will write vectors as

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$$

  We write this as $z \in \mathbb{R}^n$, which is "z is a member of ar-en."

- We commonly will need to "turn" the vector, which we write as

$$z^\top = \begin{bmatrix} z_1 & z_2 & \dots & z_n \end{bmatrix}$$

Necessary Background: Notation

We will concatenate the covariates or features into the design or feature matrix $\mathbb{X}$, where

$$\mathbb{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_p \end{bmatrix} = \begin{bmatrix} X_1^\top \\ X_2^\top \\ \vdots \\ X_n^\top \end{bmatrix} \in \mathbb{R}^{n \times p}$$

In words: The covariates (columns) will be lower case letters and the observations (rows) will be upper case letters

(It appears as though the book goes back and forth between capital and lower case for various quantities)

Norms

We will need to measure the size of vectors

The most common one is the one we use every day (implictly): Euclidean distance[4]

$$||\mathbf{x}||_2 = \sqrt{\sum_{k=1}^{p} x_k^2}$$

Additionally, we will need the Manhattan distance

$$||\mathbf{x}||_1 = \sum_{k=1}^{p} |x_k|$$

In general, the ℓ^r norm is:

$$||\mathbf{x}||_r = \left(\sum_{k=1}^{p} |x_k|^r \right)^{1/r}$$

[4]Think: the Pythagorean theorem.

Singular Value Decomposition (SVD)

Singular value decomposition (SVD)

A huge amount of statistics depends on (numerical) linear algebra concepts

Many, many topics in (numerical) linear algebra are implicitly motivated by the singular value decomposition (SVD)

Some examples are:

- Multiple regression
- Penalized least squares
- Principal components analysis
- Discriminant analysis

SVD

The SVD is a generalization of the eigenvector decomposition

Instead of

$$\mathbb{X} = UDU^\top \longleftarrow \text{eigenvector decomposition}$$

we get

$$\mathbb{X} = UDV^\top \longleftarrow \text{singular value decomposition}$$

This change makes the (unique) SVD always exist

(As opposed to the eigenvector decomposition which only exists sometimes)

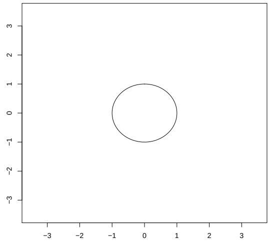
Singular value decomposition (SVD)

It turns out we can think of matrix multiplication in terms of circles and ellipsoids

Take a matrix $\mathbb{X}$ and let's look at the set of vectors

$$B = \{\beta : ||\beta||_2 \leq 1\}$$

This is a circle!

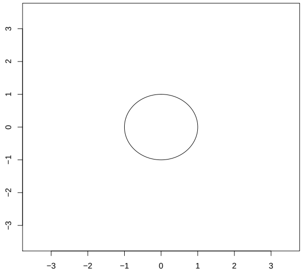


Singular value decomposition (SVD)

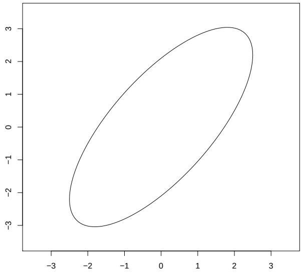
What happens when we multiply vectors in this circle by $\mathbb{X}$?

Let

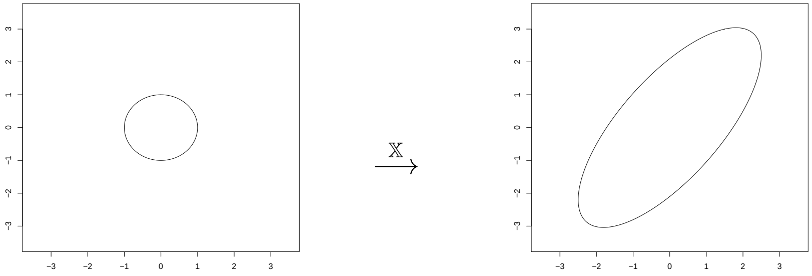
$$\mathbb{X} = \begin{bmatrix} 2.0 & 0.5 \\ 1.5 & 3.0 \end{bmatrix} \text{ and } \mathbb{X}\beta = \begin{bmatrix} 2\beta_1 + 0.5\beta_2 \\ 1.5\beta_1 + 3\beta_2 \end{bmatrix}$$



$$\xrightarrow{\mathbb{X}}$$



Singular value decomposition (SVD)

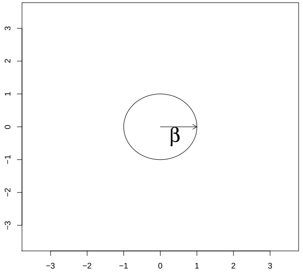


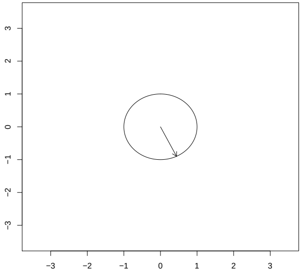
What happened?

1. The coodinate axis gets rotated
2. The new axis gets elongated (making an ellipse)
3. This ellipse gets rotated

Let's break this down into parts...

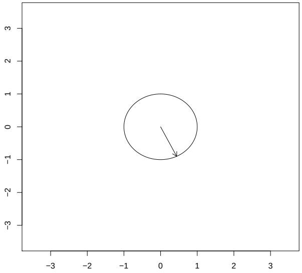
Singular value decomposition (SVD)

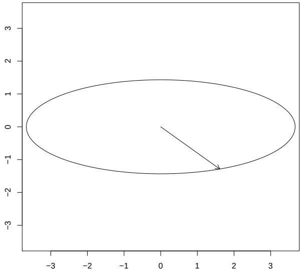




1. The coordinate axis gets rotated

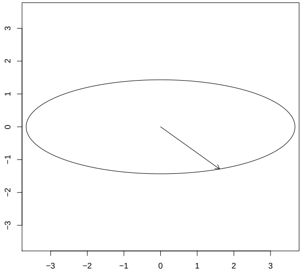
Singular value decomposition (SVD)

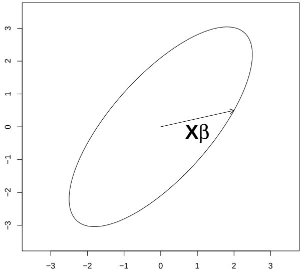




1. The coordinate axis gets rotated
2. The new axis gets elongated (making an ellipse)

Singular value decomposition (SVD)





1. The coordinate axis gets rotated
2. The new axis gets elongated (making an ellipse)
3. This ellipse gets rotated

Rotation and Elongation

Rotations: These can be thought of as just reparameterizing the coordinate axis. This means that they don't change the geometry. As the original coordinate axis was orthogonal (that is; perpendicular), the new coordinates must be as well.

Let $\mathbf{v}_1, \mathbf{v}_2$ be two normalized, orthogonal vectors. This means that:

$$\mathbf{v}_1^\top \mathbf{v}_2 = 0 \quad \text{and} \quad \mathbf{v}_1^\top \mathbf{v}_1 = \mathbf{v}_2^\top \mathbf{v}_2 = 1$$

In matrix notation, if we create V as a matrix with normalized, orthogonal vectors as columns, then:

$$V^\top V = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & & \vdots & & \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} = I$$

Here, I is the identity matrix.

Rotation and Elongation

Elongation: These can be thought of as stretching vectors along the current coordinate axis. This means that they **do** change the geometry by distorting distances. These are given by multiplication by diagonal matrices.

All diagonal matrices have the form:

$$D = \begin{bmatrix} d_1 & 0 & 0 & \dots & 0 \\ 0 & d_2 & 0 & \dots & 0 \\ & & \vdots & & \\ 0 & 0 & 0 & \dots & d_p \end{bmatrix}$$

Singular value decomposition (SVD)

Using this intuition, for any matrix $\mathbb{X}$ it is possible to write its SVD:

$$\mathbb{X} = UDV^\top$$

where

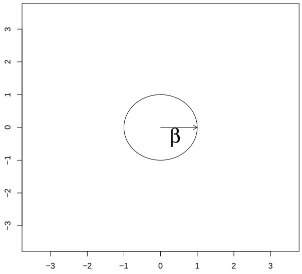
- U and V are orthogonal (think: rotations)
- D is diagonal (think: elongation)
- The diagonal elements of D are ordered as

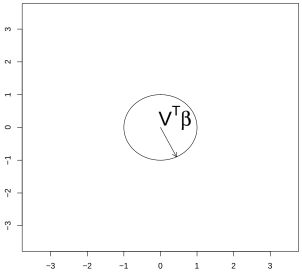
$$d_1 \geq d_2 \geq \ldots \geq d_p \geq 0$$

Many properties of matrices can be 'read off' from the SVD.

Rank: The rank of a matrix answers the question: how many dimensions does the ellipse live in? In other words, it is the number of columns of the matrix $\mathbb{X}$, not counting the columns that are 'redundant'.
It turns out the rank is exactly the quantity q such that $d_q > 0$ and $d_{q+1} = 0$.

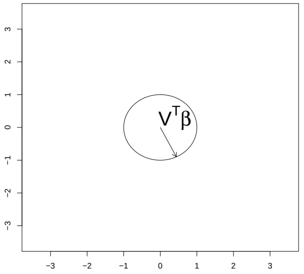
Singular value decomposition (SVD)

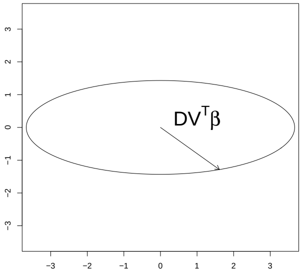




1. The coordinate axis gets rotated (Multiplication by $V^\top$)

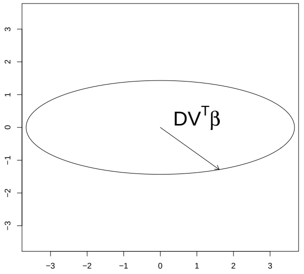
Singular value decomposition (SVD)

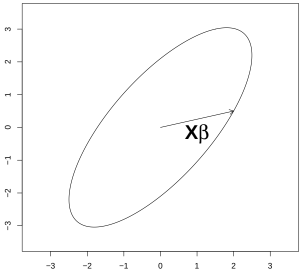




1. The coordinate axis gets rotated (Multiplication by $V^\top$)
1. The new axis gets elongated (Multiplication by D)

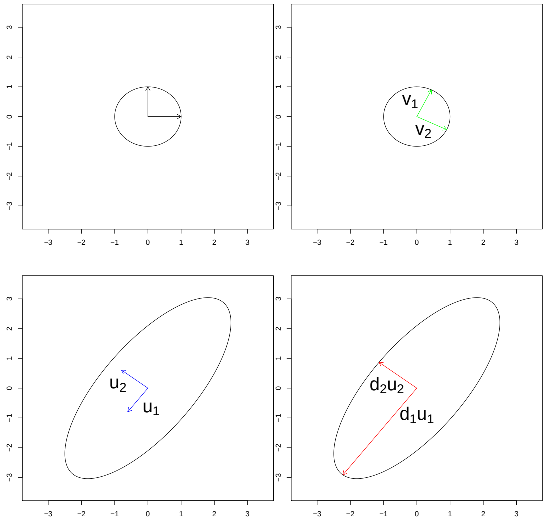
Singular value decomposition (SVD)





1. The coordinate axis gets rotated (Multiplication by $V^{\top}$)
1. The new axis gets elongated (Multiplication by D)
2. This ellipse gets rotated (Multiplication by U)

Singular value decomposition (SVD)



Summary:

Of all the possible axes of the original circle, the one given by v_1, v_2 has the unique property:

$$\mathbb{X} v_j = d_j u_j$$

for all j.

Lastly:

$$\mathbb{X} = \sum_j d_j u_j v_j^\top$$

Singular value decomposition (SVD)

Taking this last formulation:

$$\mathbb{X} = \sum_j d_j u_j v_j^\top$$

we see that matrix multiplication looks like

$$\mathbb{X}\beta = \sum_j d_j u_j v_j^\top \beta = \sum_j u_j \theta_j$$

Implication: Multiplication by $\mathbb{X}$ re-expresses β in a new coordinate system, with coordinates $\theta^\top = [\theta_1, \theta_2, \ldots]$