

Lecture 1: Introduction

CSI 5v93: Introduction to machine learning

Baylor University
Computer Science Department

Dr. Greg Hamerly
<http://cs.baylor.edu/~hamerly/>

CSI 5v93: Introduction to machine learning, Lecture 1 – p. 1/25

Syllabus and course information

CSI 5v93 section 04, CRN #14515, 2 Credits

Meeting times: 8:00 AM - 9:00 AM T/Th Rogers 312

Syllabus and schedule: in your hands and on the web

CSI 5v93: Introduction to machine learning, Lecture 1 – p. 2/25

Course structure and evaluation

We have 16 weeks (including spring break, not including finals) and 28 class meetings.

This course will be structured into three parts: presentations (25%), several projects (45%), and a final exam (30%).

For presentations you will read and synthesize a paper by a researcher in machine learning, and present that paper to the class. Each presentation will be no longer than 20 minutes.

For the projects you will implement, test, and write up results of ideas that we learn about.

The exam will focus more on theoretical aspects.

CSI 5v93: Introduction to machine learning, Lecture 1 – p. 3/25

General announcements

- First assignment: assigned today, due January 20th (see the website).
- Get the book ASAP if you don't have it.

CSI 5v93: Introduction to machine learning, Lecture 1 – p. 4/25

Questions?

Introduction to machine learning

ML is a subset of artificial intelligence.

Machine learning definition: making a program/system that can

- learn from data
- learn from experience

(these two are really the same thing)

There are many interesting parts of machine learning and reasons to study it:

- modelling how humans and animals learn
- discovering what is required to learn a concept
- if a machine can learn, then the program designer does not need to be an expert
- building programs that tune themselves

Our book and our approach

The book we use (by Hastie, Tibshirani, and Friedman) is quite good. It takes an applied approach to machine learning, and is less concerned with theoretical aspects.

Machine learning has a lot of different methods, but fewer central ideas. It is easy to get lost in the methods. I will try to draw together the concepts at a theoretical level.

We will generally follow the book this semester, so it is definitely to your advantage to own and read the book.

I am assuming no knowledge about your background in machine learning or AI, so if you have a question about something that I gloss over, please ask.

Applied ML tends to be math-heavy, especially using linear algebra, statistics, and calculus, so you should have a reference handy for those topics.

Question: how much linear algebra and statistics do you know?

Tools of this class: Matlab and L^AT_EX

For the programming assignments, you should use Matlab. Other statistical software packages are useful, but I use Matlab.

To write up your reports for the projects, you should use L^AT_EX

To help you learn both of these, the first assignment will require you to use both of them.

Chapter 1: Introduction

The general learning problem is this:

- Given some input variables X and output variables Y
- We want to *learn* the process (or function) that converts X into Y .

More abstractly, we are given some information, but not how that information was produced. We want to learn (from data) how that information was produced.

Getting started: example learning problems

- Predict whether a patient, hospitalized due to a heart attack, will have a second heart attack. The prediction is to be based on demographic, diet, and clinical measurements for that patient.
- Predict the length of stay of a patient in a hospital, based on their clinical measurements, illness, and medical history.
- Predict the price of a stock in 6 months from now, on the basis of company performance measures and economic data.
- Identify the numbers in a handwritten ZIP code, from a digitized image.
- Estimate the amount of glucose in the blood of a diabetic person, from the infrared absorption spectrum of that person's blood.
- Identify the risk factors for prostate cancer, based on clinical and demographic variables.

Central terms

- prediction (on unseen objects)
- features
- training set
- model

Supervised/unsupervised learning

There are two primary types of machine learning: supervised and unsupervised.

Supervised learning problems have a training signal – a variable which tells the learner which is the correct answer.

For example, when learning the concept of whether an email is SPAM or NOT-SPAM, the training set has example emails labeled SPAM, and example emails labeled NOT-SPAM. These labels are the training signal.

Unsupervised learning problems do *not* have a training signal. Instead, the purpose is to learn the *structure* of the training data.

For example, learning the association that people who have high incomes also drive cars that cost more than average.

There are also **semi-supervised** methods, which mix supervised and unsupervised methods, but we will not refer to them as much.

Example application: spam identification

Training data: 4601 email messages. Goal: predict whether each is SPAM or NOT-SPAM.

This is a *supervised classification* problem.

Input: an email message. Output: prediction of SPAM or NOT-SPAM.

	george	you	your	hp	free	hpl	!	our	re
SPAM	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13
NOT-SPAM	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42

Average percentage of words (or characters) in an email message equal to the indicated word (or character). The chosen words show the largest difference between SPAM and NOT-SPAM.

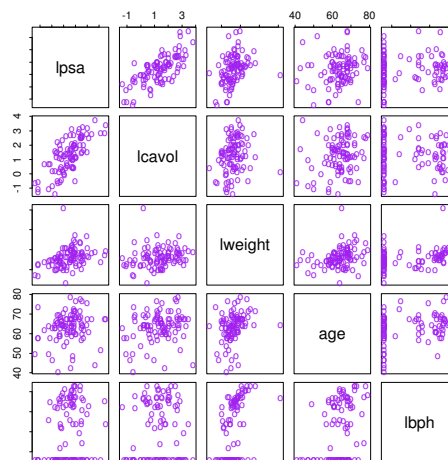
A learning method might output a rules like these:

if (%george < 0.6) and (%you > 1.5) then SPAM else NOT-SPAM

if $(0.2 \times \%you - 0.3 \times \%george) > 0$ then SPAM else NOT-SPAM

Example application: prostate cancer

Goal: predict the log of a prostate specific antigen (lpsa) based on other clinical measurements.



Inputs: clinical measurements. Output: predicted LPSA.

This is a *supervised regression* problem.

Chapter 2: Overview of supervised learning

- 2.1 – Introduction
- 2.2 – Variable types and terminology
- 2.3 – Two simple approaches to prediction: Least squares and nearest neighbors
- 2.4 – Statistical decision theory
- 2.5 – Local methods in high dimensions
- 2.6 – Statistical models, supervised learning, and function approximation
- 2.7 – Structured regression models
- 2.8 – Classes of restricted estimators
- 2.9 – Model selection and the bias-variance tradeoff

CSI 5v93: Introduction to machine learning, Lecture 1 – p. 17/25

Supervised learning and variables (2.1)

Supervised learning: learning with the correct answers known (e.g. SPAM or NOT-SPAM).

Inputs = independent variables = predictors

Outputs = dependent variables = responses

CSI 5v93: Introduction to machine learning, Lecture 1 – p. 18/25

Variable types (2.2)

Quantitative variable: involves real numbers.

- Example: 0.3, 10.65, -10002
- also called *real* variables
- distance between values has real meaning
- *regression* is prediction of quantitative values

Qualitative variable: involves classes

- Example: red, green, blue
- also called *discrete variables* or *categorical variables* or *factors*
- measurements *may* not mean anything
- *classification* is prediction of qualitative values

Variable naming (from your textbook)

We will call our input variable X . If X is a vector, then use subscripts for accessing its components: X_j .

Quantitative outputs will have the name Y , qualitative outputs will have the name G (for *group*).

Observed values are written in lowercase.

Matrix is bold, upper-case: \mathbf{X}

Learning task: given the value of an input vector X , make a good prediction of the output Y , denoted as \hat{Y} .

Training data will be numbered $1 \dots N$ (or $1 \dots n$), so:
 $(x_1, y_1), \dots, (x_i, y_i), \dots, (x_n, y_n)$.

Least squares (linear regression – 2.3.1)

A linear model is one which is linear *in the inputs*.

Examples:

- linear model: $f(x) = 3x$
- linear model: $f(x, y) = 3x - 10y$
- nonlinear model: $f(x, y) = x^2 - 4\sqrt{y} + xy$
- NOTE! linear model: $f(x, x^2, y, xy) = x^2 - xy + x - y$

Linear models are popular because:

- simple to understand
- simple to find
- stable (high bias)

Linear models make strong assumptions about the relationship between the inputs and the output.

Linear model

Given a vector of inputs $X = (X_1, X_2, \dots, X_p)$, predict the output Y with the model

$$(1) \quad \hat{Y} = \hat{\beta}_0 + \sum_{j=1}^p X_j \hat{\beta}_j$$

Using linear algebra's inner-product (aka dot product) notation we can write this as:

$$(2) \quad \hat{Y} = X^T \hat{\beta}$$

where here β is a vector of all the coefficients, and X^T is the transpose of the vector X .

Two-dimensional example (on the board).

Finding the coefficients

We defined our model: the output (Y) equals the inputs (X) times their respective coefficients (β).

Here, the learning problem is finding the appropriate coefficients β for a given training set. How should we do this?

Our first step is to define an *error function*, which is a common thing in machine learning.

The error function will tell us how well our estimate $\hat{\beta}$ is doing at describing the training data.

Least squares error function

The most common method for finding coefficients of a linear function is *least squares*.

Error function:

$$(3) \quad \text{RSS}(\beta) = \sum_{i=1}^n (y_i - x_i^T \beta)^2$$

We want to *minimize* this error function for a fixed training set.

Minimization should remind you of calculus: take the derivative with respect to β , set to zero. . .

Remember that β is a vector.

2-minute journal

Please write a response to the following on a piece of paper and hand it in immediately. Please make it anonymous (no names). Write about:

- major points you learned today
- areas not understood or requiring clarification