ECE-271B
Statistical Learning II

Nuno Vasconcelos

ECE Department, UCSD
The course

- the course is a graduate level course in statistical learning
- in SLI we covered the foundations of Bayesian or generative learning
- this term we will focus on discriminant learning
- SLI is not required and I will try to make SLII self-contained, but SLI graduates will have some advantage
- together, the two courses will give you the necessary background to understand 90% of the papers in the area
- more on generative vs discriminant later
Logistics

- **Grading:** homework (30%)
- final project (70%)

- **Homework:**
  - Three/four problem sets, roughly every two weeks
  - homework is individual
  - OK to work on problems with someone else but you have to:
    - write your own solution
    - write down the names of who you collaborated with
  - homework is due roughly two weeks after it is issued.
  - however, because this is a graduate course, there will be some flexibility
Final project

► individual, no collaboration.

► ideal would be a contribution publishable in a conference.

► this could be
  • new theory or algorithms
  • a creative application of existing techniques to, say, vision, speech, data-mining, neurosciences
  • a thorough experimental comparison of existing techniques
  • a tutorial review of literature not covered in class

► not restricted to material covered in SL II. Material covered in SLI also acceptable.

► however, SLII material is preferred
Final project

deliverables:
• 12 page project report by end of term.
• 20 minute presentations on the last two classes.

project grading:
• final paper: 70 % (50 % for content, 20 % for writing)
• presentation 30 %

other classes:
• can be a joint project with another class, but you will have to do more work
• discuss your plans with the instructor
Project progress

- make sure you **start early**: intermediate materials to keep track of your progress

**suggested schedule**

- Week 3: 2-page project proposal
- Week 5: meet with instructor (10/15 min, optional)
- Week 7: 1-page intermediate project report
- Week 9: final report
- Week 10: presentation

- note that it is your responsibility to find a suitable project topic
Resources

Course web page:
http://www.svcl.ucsd.edu/~nuno
  • all handouts, problem sets, code will be available there

TA: Hamed Masnadi-Shirazi, hmasnadi@ucsd.edu
  • Office hours: TBA

Me: Nuno Vasconcelos, nuno@ece.ucsd.edu, EBU1-5603
  • Office hours: Friday, 9:30-11 AM

My assistant:
  • Travis Spackman (tspackman@ece.ucsd.edu), EBU1, may sometimes be involved in administrative issues
Texts

► there is no required text, hand-outs as we go

► various texts cover bits and pieces

► if you want to concentrate on one
  • “Elements of Statistical Learning”, Hastie, Tibshirani, Fredman, Springer, 2001

► various other options:
  • B. Scholkpf, A. Smola, Learning with Kernels, MIT Press, 2002.
**Texts**

- **generic books on learning or other topics we will cover**
  - Duda, Hart, and Stork, “Pattern Classification”, John Willey, 2001
  - Christopher Bishop, “Neural Networks for Pattern Recognition”, Oxford University Press, 1996.

- **stuff you must know well:**
  - “Linear Algebra”, Gilbert Strang, 1988
The course

► why statistical learning?

► there are many processes in the world that are ruled by deterministic equations

  • e.g. \( f = ma \); linear systems and convolution, Fourier, etc, various chemical laws

  • there may be some “noise”, “error”, “variability”, but we can leave with those

  • we don’t need statistical learning

► learning is needed when

  • there is a need for predictions about variables in the world, \( Y \)

  • that depend on factors (other variables) \( X \)

  • in a way that is impossible or too difficult to derive an equation for.
Statistical learning

- goal: given a function

\[ x \rightarrow f(.) \rightarrow y = f(x) \]

and a collection of example data-points, learn what the function \( f(.) \) is.

- this is called training.

- two major types of learning:
  
  - unsupervised: only \( X \) is known, usually referred to as clustering;
  
  - supervised: both are known during training, only \( X \) known at test time, usually referred to as classification or regression.
Supervised learning

- X can be anything, but the type of Y dictates the type of supervised learning problem
  - Y in \{0,1\} referred to as detection
  - Y in \{0, ..., M-1\} referred to as classification
  - Y real referred to as regression

- theory is quite similar, algorithms similar most of the time

- we will emphasize classification, but will talk about regression when particularly insightful
two major types of classifiers:

- **discriminant**: directly recover the decision boundary that best separates the classes;
- **generative**: fit a probability model to each class and then derive the classification rule (implicitly determines the border).

a lot more on this later!

**focus** will be on discriminant learning.

we have covered generative in SLI.
Caution

- How do we know learning worked?
- We care about generalization, i.e., accuracy outside training set.
- Models that are too powerful can lead to over-fitting:
  - E.g. in regression I can always fit exactly $n$ pts with polynomial of order $n-1$.
  - Is this good? How likely is the error to be small outside the training set?
  - Similar problem for classification.
- Fundamental LAW: only test set results matter!!!
Generalization

- good generalization requires controlling the trade-off between training and test error
  - training error large, test error large
  - training error smaller, test error smaller
  - training error smallest, test error largest

- the first question in learning is: how do we formalize these notions?
  - what is error?
  - how do we measure it?
  - how do we minimize it?
Loss functions

- to measure performance of a learning machine we need a loss function

- Define
  - $g(x, \alpha)$ as the output of the learning machine to input $x$ when its parameters have value $\alpha$
  - $L[y, g(x, \alpha)]$ as the loss associated by the output $g(x, \alpha)$ when the true value is $y$

- Common loss functions
  - classification: “0-1” loss

$$L[y, g(x, \alpha)] = \begin{cases} 0, & \text{if } y = g(x, \alpha) \\ 1, & \text{if } y \neq g(x, \alpha) \end{cases}$$
Loss functions and Risk

- **common loss functions**
  - regression: “squared error” loss
    \[ L[y, g(x, \alpha)] = (y - g(x, \alpha))^2 \]

- **goal of the learning machine**: to find the set of parameters that minimizes the expected value of the loss

- **this is called the risk**: 
  \[
  R(\alpha) = E_{x,y} \{ L[y, g(x, \alpha)] \} \\
  = \int P_{x,y}(x, y) L[y, g(x, \alpha)] dxdy
  \]
Risk and empirical risk

► in practice it is impossible to evaluate the risk, because we do not know what $P_{X,Y}(x,y)$ is.

► if we knew this, learning would be trivial by application of the Bayes decision rule discussed in SLI (more on this later)

► all we have is a training set

$$D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$

► we estimate the risk by the empirical risk on this training set

$$R_{emp}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} L[y_i, g(x_i, \alpha)]$$
The four fundamental questions

- the question is then: does empirical risk minimization (ERM) assure the minimization of the risk?

- Vapnik and Chervonenkis studied this question extensively and identified four fundamental questions

  1. what are the necessary and sufficient conditions for consistency of ERM, i.e. convergence?
  2. how fast is the rate of convergence? If $n$ needs to be very large, ERM is useless in practice since we only have a finite training set
  3. is there a way to control the rate of convergence?
  4. how can we design algorithms to control this rate?

- the formal answer to these questions requires a mathematical sophistication beyond what we require here
The four fundamental questions

I will try to convey the main ideas as we go along.

the nutshell answers are:

1. yes, ERM is consistent
2. the convergence rate is quite slow, only asymptotic guarantees are available
3. yes, there a way to control the rate of convergence, but it requires a different principle which VC called structural risk minimization (SRM)
4. we will talk a lot about this!

It turns out that SRM is an extension of ERM.
let’s look at this in the regression context
SRM vs ERM

- ERM minimizes training loss only.
- the problem is that more complicated functions always produce smaller training loss
- if the class of functions is the set of polynomials, it is always best to use those of degree $n-1$.
- the solution has large variance
  - if you produce another set of $n$ training points the best fit or order $n-1$ will be quite different
  - points outside of the training set are not likely to be well predicted
- since more complex models always have more degrees of freedom, this is always the case
SRM vs ERM

- to guarantee good generalization we need to penalize complexity
- VC formalized this idea by showing that
  \[ R(\alpha) \leq R_{emp}(\alpha) + \Phi(n, g) \]
- \( \Phi(n, g) \) is a confidence interval that depends on
  - number of training points \( n \)
  - VC dimension of the family of functions \( g(x, \alpha) \)
- VC dimension:
  - a measure of complexity, usually a function of the number of parameters
  - we will talk a lot about this
SRM vs ERM

- note that minimizing the bound provides guarantees on the risk even when the training set is finite!

- significance:
  - this is much more relevant in practice than the classical results which only give asymptotic guarantees
  - the bound inspires a practical way to control the generalization ability

- controlling generalization:
  \[ R(\alpha) \leq R_{emp}(\alpha) + \Phi(n, g) \]
  - given the function family, the first term only depends on parameters
  - the second term depends on the family of functions
SRM vs ERM

- this suggests the SRM principle:
  - start from a nested collection of families of functions
    \[ S_1 \subset \cdots \subset S_k \]
    where \( S_i = \{g_i(x, \alpha), \text{ for all } \alpha\} \)
  - for each \( S_i \), find the function (set of parameters) that minimizes the empirical risk
    \[ R_{\text{emp}}^i = \min_{\alpha} \frac{1}{n} \sum_{k=1}^{n} L[y_k, g_i(x_k, \alpha)] \]
  - select the function class that achieves the lowest bound on the actual risk
    \[ R^* = \min_i \{ R_{\text{emp}}^i + \Phi(n, g_i) \} \]
- note that SRM is an outer-loop around ERM
Example (polynomials)

- SRM for polynomial regression:
  - $S_i$ is the set of polynomials of degree $i$
    
    \[ S_i = \left\{ \sum_{k=0}^{i} \alpha_k x^k, \forall (\alpha_0, \ldots, \alpha_i) \right\} \]
    
  - for each $i$, find the parameters that minimizes the empirical risk
    
    \[ R_{\text{emp}}^i = \min_{\alpha} \frac{1}{n} \sum_{l=1}^{n} \left( y_l - \sum_{k=0}^{i} \alpha_k x_l^k \right)^2 \]
    
    this is a simple least squares problem

  - select the order $i$ that achieves the lowest bound
    
    \[ R^* = \min_{i} \left\{ R_{\text{emp}}^i + \Phi(n, g_i) \right\} \]
The confidence interval

- the problem is that $\Phi(n,g)$ is usually not easy to compute
- for classification, VC showed that there is a case which is simple
  - when the decision boundary is a hyperplane
  - and the classes are separable
  - minimizing $\Phi(n,g)$ is equivalent to maximizing the margin
Support vector machines

since, for separable classes, the empirical risk can be made zero by classifying each point correctly

the SRM principle can be implemented by solving the optimization problem

\[ w^* = \arg \max_{w} \text{margin}(w) \]

\[ s.t \ x_i \text{ correctly classified} \]

this is an optimization problem with \( n \) constraints, not trivial but solvable

the solution is the support-vector machine (points on the margin are “support vectors”)

we will talk a lot about these
Support vector machines

we will use them as an excuse to study core ideas in optimization theory:

• unconstrained optimization
• constrained optimization and Lagrangian methods
• duality theory

these are likely to be useful even if you do not care about statistical learning

SVM extensions:

• what if the classes are not separable?
  • straightforward extension
• what if hyperplane is not good enough?
  • this is more interesting
Kernels

- the trick is to map the problem to a higher dimensional space:
  - non-linear boundary in original space
  - becomes hyperplane in transformed space
- this can be done efficiently by the introduction of a kernel function
- classification problem is mapped into a reproducing kernel Hilbert space
- kernels are
  - at the core of the success of SVM classification
Kernels

- Kernels are interesting in many ways.
- We will use them as an excuse to study:
  - Basic aspects of functional analysis (Hilbert spaces, reproducing Hilbert spaces, etc.)
  - Basics of regularization theory
  - The Mercer conditions and positive definite functions
- Once again, of interest even if you do not care about statistical learning.
- Also a very timely topic of research.
- Many of the classical techniques (e.g., PCA, LDA, ICA, etc.) can be kernelized with significant improvement.
Other large-margin classifiers

- there are other ways of maximizing the margin other than SVMs
- one solution which is becoming popular is **boosting**
  - gradient descent on the margin function
  - trivial implementation
  - generalizes the earliest learning machine (perceptron)
  - widely popular in vision and speech
- we will cover all of these issues
- if there is time I will also talk about **Gaussian processes**, and some very recent large-margin ideas
Reasons to take the course

 ► statistical learning
  • tremendous amount of theory
  • but things invariably go wrong
  • too little data, noise, too many dimensions, training sets that do not reflect all possible variability, etc.

 ► good learning solutions require:
  • knowledge of the domain (e.g. “these are the features to use”)
  • knowledge of the available techniques, their limitations, etc. (e.g. here a linear SVM is enough, but there I need a kernel)

 ► in the absence of either of these you will fail!

 ► the course covers very recent topics in learning and can give you ideas for research

 ► will cover various topics of general interest
Any questions?