

Applied inductive learning - Lecture 6

Louis Wehenkel

Department of Electrical Engineering and Computer Science
University of Liège

Montefiore - Liège - October 24, 2005

Find slides: <http://montefiore.ulg.ac.be/~lwh/AIA/>

Probabilistic framework to study batch-mode supervised learning

- Data generation model

- Bayes model

- Learning from a sample

Hints about automatic learning theories

- Geometrical description of learning algorithms

- Statistical learning theory

- Bias/variance tradeoff

Guidelines to design automatic learning algorithms

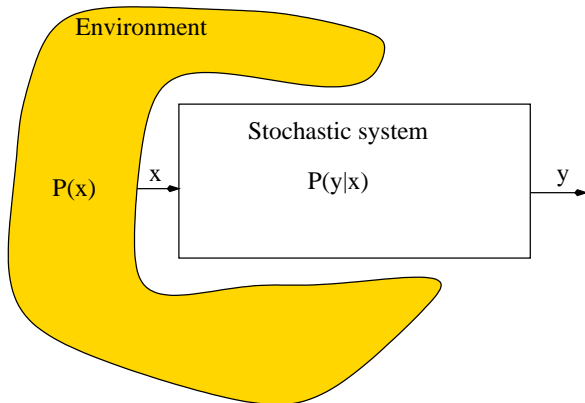


Figure: Intuitive system model: pictorial view

The practical batch-mode supervised learning problem

(Reminder)

- ▶ Objects (or observations): $LS = \{o_1, \dots, o_N\}$
- ▶ Attribute vector: $(a_1(o_i), \dots, a_n(o_i))^T$, $\forall i = 1, \dots, N.$
- ▶ Outputs: $y^i = y(o_i)$ or $c^i = c(o_i)$, $\forall i = 1, \dots, N.$
- ▶ LS Table

o	$a_1(o)$	$a_2(o)$	\dots	$a_n(o)$	$y(o)$
1	a_1^1	a_2^1	\dots	a_n^1	y^1
2	a_1^2	a_2^2	\dots	a_n^2	y^2
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
N	a_1^N	a_2^N	\dots	a_n^N	y^N

We will use the notation $x(o)$ to denote all the **inputs** and the notation $y(o)$ to denote the **output**, irrespectively of their type (numerical or categorical).

Data generation model for batch-mode supervised learning

- ▶ Objects o are drawn **independently** from some **finite** universe U according to some **fixed** probability distribution P_U
- ▶ **Inputs** and **outputs** are random variables defined on the universe U , i.e. functions mapping objects to values in the sets X and Y
- ▶ These functions induce probability distributions
 - ▶ P_X over the set $X = \{x_1, x_2, \dots\}$
 - ▶ P_Y over the set $Y = \{y_1, y_2, \dots\}$
 - ▶ $P_{X,Y}$ over the set $X \times Y = \{(x_1, y_1), (x_2, y_2), \dots\}$
 - ▶ and for every $x \in X$ the conditional probability distribution

$$P_{Y|x}(y) = \frac{P_{X,Y}(x, y)}{P_X(x)}$$

- ▶ NB. Since U is supposed finite, X , Y and $X \times Y$ are finite.

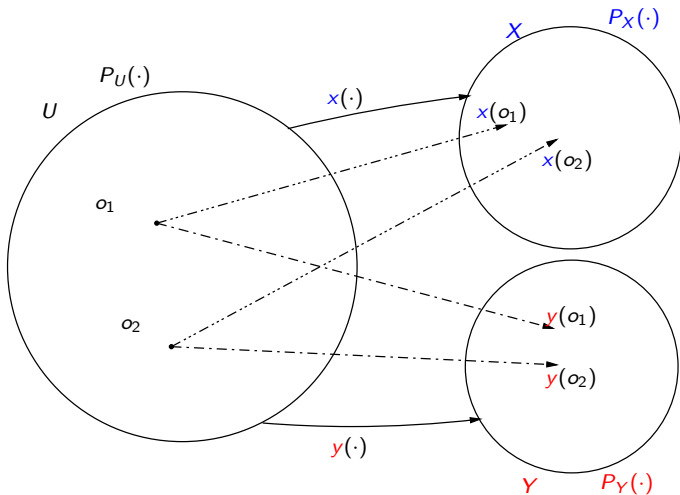


Figure: Data generation model: pictorial view

Bayes model: definition

- ▶ To simplify let's assume that inputs and outputs are binary random variables. Hence $X = \{x_1, x_2\}$ and $Y = \{y_1, y_2\}$.
- ▶ Assume that, $\forall o \in U$, we can compute the values of $P_U(o)$, $x(o)$ and $y(o)$.
- ▶ Then we can provide an algorithm to compute the most probable value of $y(o)$ given $x(o)$:

- ▶ Compute:

$$\begin{aligned} P_{X,Y}(x_i, y_j) &= \sum_{o \in U} \delta_{x(o), x_i} \delta_{y(o), y_j} P_U(o), & \forall i, j = 1, 2 \\ P_X(x_i) &= \sum_{o \in U} \delta_{x(o), x_i} P_U(o), & \forall i = 1, 2 \end{aligned}$$

- ▶ Derive for $i, j = 1, 2$:

$$P_{Y|x_i}(y_j) = \frac{P_{X,Y}(x_i, y_j)}{P_X(x_i)}$$

- ▶ Construct decision rule:

$$B(x) = \arg \max_y P_{Y|x}(y).$$

Bayes model: properties

- ▶ Let $f(x) : X \rightarrow Y$ be a decision rule. Its probability of error is

$$P_e(f) = 1 - \sum_{o \in U} \delta_{f(x(o)), y(o)} P_U(o) = 1 - \sum_x P_X(x) P_{Y|x}(f(x))$$

- ▶ Since $B(x) = \arg \max_y P_{Y|x}(y)$ we have, $\forall f \in Y^X, \forall x \in X$:

$$P_{Y|x}(f(x)) \leq P_{Y|x}(B(x)).$$

- ▶ Hence, $P_e(B) \leq P_e(f)$: **the Bayes model is optimal in Y^X .**
- ▶ Notice that $B(\cdot)$ depends only on $P_{Y|x}$, not on P_X .
- ▶ But $P_e(B)$ (**residual error**) depends on P_X .
- ▶ If $\exists f \in Y^X$ such that $\forall o \in U : y(o) = f(x(o))$, then $P_e(f = B) = 0, \forall P_X$. **The problem is said to be deterministic.**

Bayes model: extension to mean square error criterion

- ▶ Suppose $y(o) \in \mathbb{R}$ and let $f(x) : X \rightarrow Y$ be a regression model. Its expected square error is

$$SE(f) = \sum_o P_U(o) (f(x(o)) - y(o))^2 = \sum_x P_X(x) \sum_y P_{Y|x}(y) (f(x) - y)^2.$$

- ▶ Define $B_{SE}(x) = \arg \min_y \sum_{y'} P_{Y|x}(y') (y - y')^2$. We have:

$$\arg \min_y \sum_{y'} P_{Y|x}(y') (y - y')^2 = \sum_y P_{Y|x}(y) y = E_{Y|x}\{y\}.$$

- ▶ Hence $\forall f \in \mathbb{R}^X : SE(f) \geq SE(B_{SE})$.
- ▶ Notice that, again, $B_{SE}(\cdot)$ depends only on $P_{Y|x}$, not on P_X .
- ▶ In general, if we define a loss function $\ell(y, y') : Y \times Y \rightarrow \mathbb{R}^+$, we can define a corresponding Bayes model by

$$B_\ell(x) = \arg \min_y \sum_{y'} P_{Y|x}(y') \ell(y, y').$$

Learning from a finite sample: trivial algorithm

- ▶ Assume that sample ls is composed of x and y values of N objects drawn independently from U according to P_U , i.e.
 $ls = ((x^k, y^k))_{k=1}^N \in (X \times Y)^N \sim P_{X,Y}^N$.

- ▶ Compute:

$$\begin{aligned}\hat{P}_{X,Y}(x_i, y_j) &= N^{-1} \sum_{k=1}^N \delta_{x^k, x_i} \delta_{y^k, y_j} & \forall i, j = 1, 2 \\ \hat{P}_X(x_i) &= N^{-1} \sum_{k=1}^N \delta_{x^k, x_i} & \forall i = 1, 2\end{aligned}$$

- ▶ Derive for $i, j = 1, 2$:

$$\hat{P}_{Y|x_i}(y_j) = \frac{\hat{P}_{X,Y}(x_i, y_j)}{\hat{P}_X(x_i)}$$

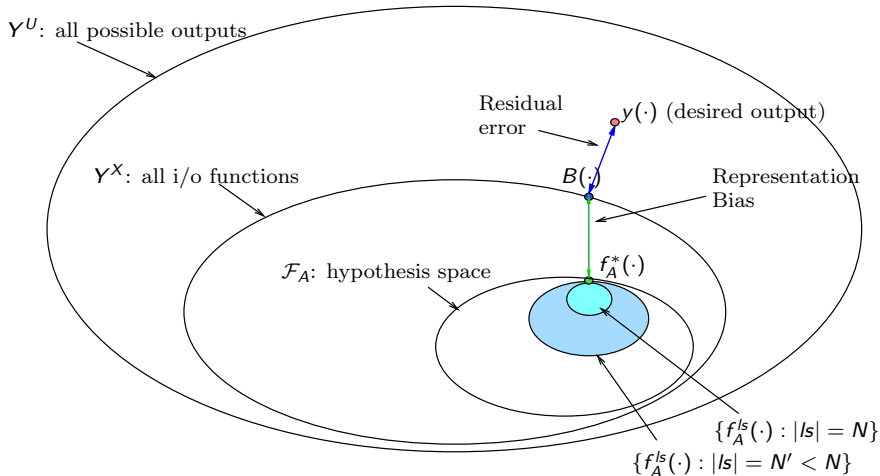
- ▶ Construct decision rule:

$$\hat{B}(x) = \arg \max_y \hat{P}_{Y|x}(y) \quad [\text{or, } \hat{B}_\ell(x) = \arg \min_y \sum_{y'} \hat{P}_{Y|x}(y') \ell(y, y')]$$

Motivation for nontrivial automatic learning algorithms

- ▶ Consider a high-dimensional input space, e.g. $x = (a_1, \dots, a_n)$
 - ▶ $a_i, y \in \{0, 1\}$, and $n = 500$: 2^{501} numbers to determine, in order to choose correct function among 2^{500} .
 - ▶ $a_i, y \in \{1, \dots, 10\}$, and $n = 100$: 10^{101} numbers to determine, in order to choose correct function among 10^{100} .
 - ▶ Trivial approach fails (for ever..., cf curse of dimensionality)
- ▶ (Two) other approaches to handle practical learning problems:
 1. A priori define set \mathcal{F} of functions $f(\cdot)$ and criterion $C(f, ls)$
 - ▶ Try to build algorithm: $A(ls) = \arg \min_{f \in \mathcal{F}} C(f, ls)$
 2. Invent algorithm A to pick a function $f_A^{ls} \in \mathcal{F}_A$.
 - ▶ See whether $\exists C : A(ls) = \arg \min_{f \in \mathcal{F}_A} C(f, ls)$.
- ▶ Questions:
 - ▶ Define “good” sets \mathcal{F} , “good” algorithms A , “good” criteria C
 - ▶ Empirical and theoretical behavior of $A, \arg \min_{f \in \mathcal{F}} C$

Geometrical viewpoint:

(fixed input space X , output space Y , algorithm A)

NB: distance between r.vars. defined by: $d(y(\cdot), y'(\cdot)) = \sum_{o \in U} P_U(o) \ell(y(o), y'(o))$

Learning theory: topics of concern

- ▶ For a given algorithm A , how do these sets and distances change
 - ▶ when the sample size $|S|$ increases
 - ▶ when the hypothesis space becomes larger
 - ▶ when the “complexity” of the target function $y(o)$ increases
- ▶ How should we change an algorithm A , or a learning problem, so as to increase accuracy
 - ▶ to reduce representation bias
 - ▶ to reduce size of $\{f_A^S\}$
 - ▶ to reduce residual error
- ▶ Three main theories
 - ▶ Statistical learning theory (Vapnik et al)
 - ▶ Bias/variance tradeoff (classical statistics)
 - ▶ Bayesian framework applied to learning (modern statistics)

Statistical learning theory: framework

For a fixed loss function ℓ , y , and hypothesis space \mathcal{F} let

- ▶ the theoretical risk of $f \in \mathcal{F}$

$$R(f) = \sum_{o \in U} \ell(f(x(o)) - y(o)) P_U(o),$$

- ▶ and the optimal function

$$f^* = \arg \min_{f \in \mathcal{F}} R(f).$$

- ▶ For any learning sample Is , let the empirical risk of $f \in \mathcal{F}$ be

$$R_e(f, Is) = \sum_{o \in Is} \ell(f(x(o)) - y(o)) |Is|^{-1},$$

- ▶ and define the empirical risk minimizing function by

$$f_*^{Is} = \arg \min_{f \in \mathcal{F}} R_e(f, Is).$$

Statistical learning theory: main qualitative results

- ▶ To characterize the behaviour of f_*^{ls} vs f^* , we need to define a measure of size of the hypothesis space. This measure is called the **VC-dimension** (see course notes).
- ▶ The main condition that has to be satisfied by \mathcal{F} is to have a **finite** VC-dimension. Under this condition, one can prove in general (for almost any P_u, x, y, ℓ) that
- ▶ Consistency holds true (convergence in a probabilistic sense)

$$\lim_{|ls| \rightarrow \infty} R_e(f_*^{ls}, ls) = \lim_{|ls| \rightarrow \infty} R(f_*^{ls}) = R(f^*)$$

- ▶ Bounds on supoptimality can be derived for finite learning samples. These allow to measure speed of convergence, and derive upper bounds on the sample size needed to reach some target precision with respect to f^* .

Statistical learning theory: intuitive counter example

- ▶ Consider a binary classification problem ($y \in \{0, 1\}$), with one real valued continuous input variable $x \in \mathbb{R}$, and suppose that

$$P(Y = 1|x) = \frac{1}{1 + \exp(-x)}$$

- ▶ The Bayes model is $B(x) = 1(x)$ (a single threshold at $x = 0$).
- ▶ Its probability of error can be computed by

$$P_e(B) = \int_{-\infty}^{+\infty} |1(x) - P(Y = 1|x)| p(x) dx > 0.$$

- ▶ Suppose that \mathcal{F} is the set \mathcal{T}^* of all decision trees of arbitrary size. Then, we have for (almost) every learning sample

$$\exists t \in \mathcal{T}^* : R_e(t, l_s) = 0. \text{ Hence } \lim_{|l_s| \rightarrow \infty} R_e(t_*^{l_s}, l_s) = 0 \neq P_e(B).$$

Statistical learning theory: intuitive discussion

- ▶ The set \mathcal{T}^* is too large: it must have an **infinite** VC-dimension
- ▶ However, for all k , the set \mathcal{T}^k of decision trees with **at most** k terminal nodes, has finite VC-dimension (see course notes).
- ▶ For such a hypothesis space the theory guarantees consistency and provides bounds, for every (finite) k .
- ▶ Because, \mathcal{T}^* can classify perfectly any learning sample of any size, it is unable to learn hypotheses which do not classify the learning sample perfectly, as was required in our **non-deterministic** example.
- ▶ Actually, **finite** VC-dimension of a hypothesis space \mathcal{F} means that for sufficiently large $|I_S|$, there exist samples which can't be classified in every possible way by choosing a function in \mathcal{F} .

Statistical learning theory: main conclusion

- ▶ Similar results hold for MPLs with an arbitrary number of hidden neurons and other flexible learning architectures.
- ▶ The **universal approximation property** of a hypothesis space, has as consequence **non learnability**.

Bias/variance decomposition: for regression problems

Accuracy of models produced by an algorithm in a given context

- ▶ Assume problem (inputs X , outputs Y , relation $P(X, Y)$) and sampling scheme (e.g. fixed size $LS \sim P^N(X, Y)$).
- ▶ Take model error function (e.g. $Err_{f,Y} \equiv E_{X,Y}\{(f(X) - Y)^2\}$) and evaluate **expected** error of algo A (i.e. $\overline{Err}_{A,Y} \equiv E_{LS}\{Err_{f_A^{LS},Y}\}$)

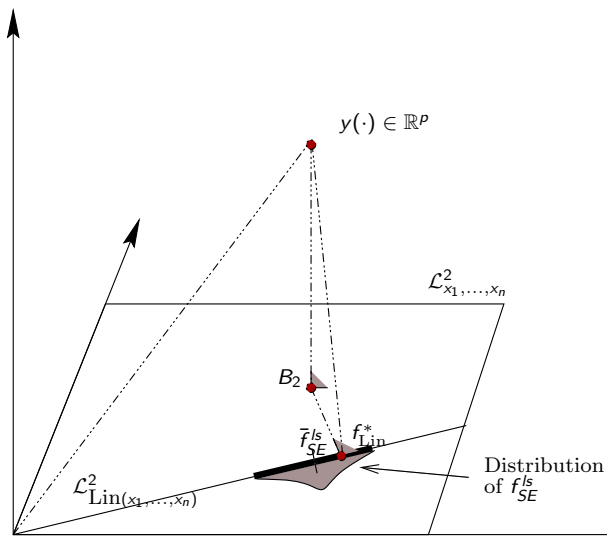
$$\text{We have } \overline{Err}_{A,Y} - Err_{B,Y} = \text{Bias}_A^2 + \text{Var}_A$$

where

- ▶ B is the best possible model (here, $B(x) \equiv E_{Y|x}\{Y\}$)
- ▶ $\text{Bias}_A^2 = Err_{\bar{f}_A,B}$ ($\bar{f}_A(x) \equiv E_{LS}\{f_A^{LS}(x)\}$)
- ▶ $\text{Var}_A = \overline{Err}_{A,\bar{f}_A}$ (dependence of model on sample)

Refined geometrical viewpoint

(suppose $U = \{1, \dots, p\}$; $Y = \mathbb{R}$)



Bias/variance tradeoff: qualitative discussion

- ▶ Increasing the hypothesis space, $\mathcal{F}' \supset \mathcal{F}$ (fixed $|S|$):
 - ▶ Decreases Bias
 - ▶ Increases Variance
 - ▶ Does not affect residual error
- ▶ Increasing the sample size, $|S'| > |S|$ (fixed \mathcal{F}):
 - ▶ Leaves Bias (typically) unchanged
 - ▶ Decreases Variance
 - ▶ Does not affect residual error
- ▶ Increasing the number of inputs a_i (fixed $|S|$, \mathcal{F} adapted):
 - ▶ Can not increase residual error, and often decreases it
 - ▶ Often reduces bias, but can also increase bias in some cases
 - ▶ Often increases variance, but can also decrease it in some cases
- ▶ **But there are other ways to influence the bias/variance tradeoff** (see next lecture).

Design of learning algorithms

Approach 1: define $C(f, I_S)$ and optimize C over \mathcal{F}

- ▶ Most studied criteria C
 - ▶ Empirical risk (loss) minimization (ERM)
 - ▶ Maximum likelihood (ML) principle
 - ▶ Maximum a posteriori (MAP) choice
- ▶ Rather powerful theories exist
- ▶ However, the most powerful learning algorithms existing today have not been obtained in this way

Approach 2: define algorithm (using intuition) and study behavior

- ▶ Bias/variance tradeoff
- ▶ Bayesian intuitions
- ▶ Vapnik intuitions

Homework

- ▶ Choose one of the simple examples discussed in this lecture
- ▶ Put numbers in the problem (defining the probability distributions $P(x)$, $P(y|x)$...)
- ▶ Choose loss function (P_e , or $|\cdot|^2$).
- ▶ Compute Bayes model
- ▶ Choose learning algorithm (e.g. DT, MLP, LR, kNN)
- ▶ Try to set up a computer experiment to estimate bias, variance, residual error etc.