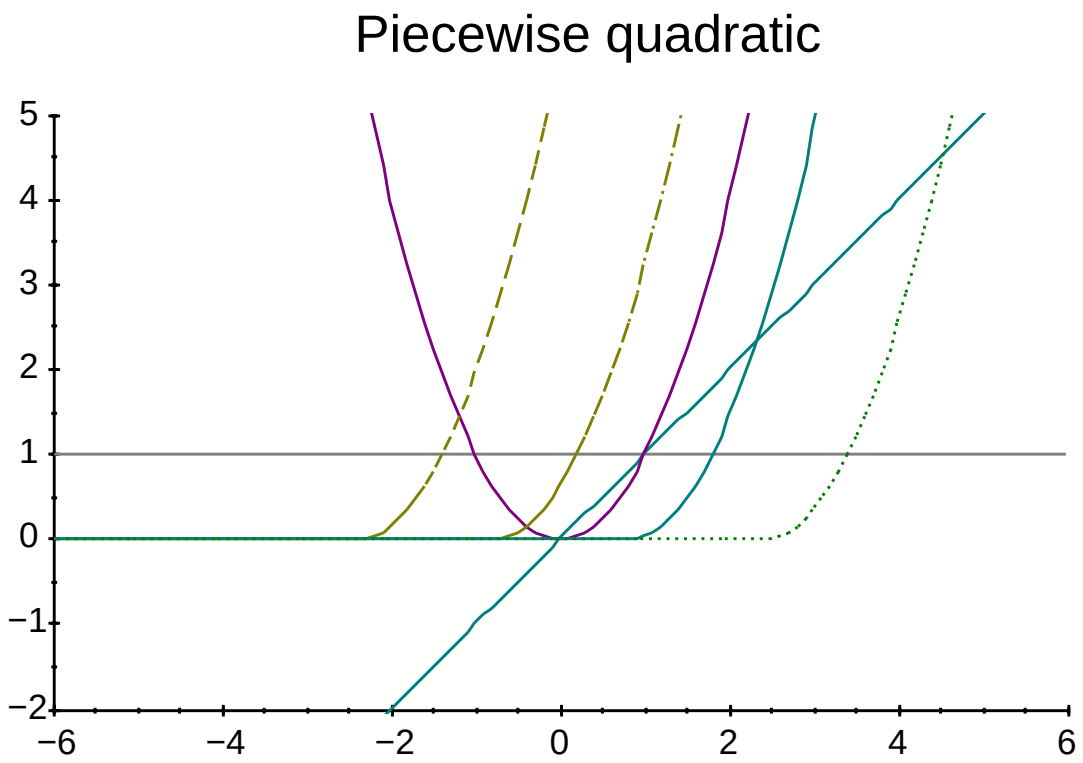
Connecting the dots with common sense and linear models

Léon Bottou

NEC Labs America

COS 424 – 2/4/2010

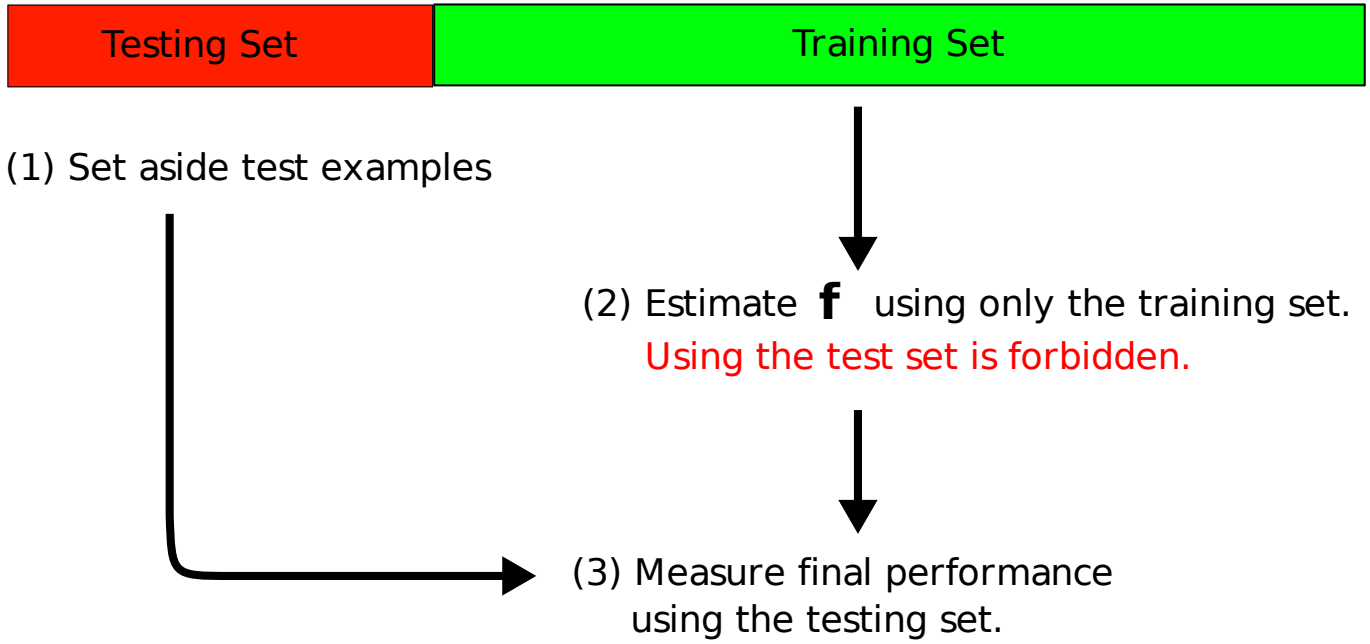
Piecewise Polynomial (Splines)



– Quadratic splines : $\Phi(x) = 1,\ x,\ x^2,\ \ldots\ \max(0, x - r_k)^2\ \ldots$

– Cubic splines : $\Phi(x) = 1,\ x,\ x^2,\ x^3,\ \ldots\ \max(0, x - r_k)^3\ \ldots$

The "training set/testing set" paradigm



– One should only use the testing set once! Of course. . .
– The more we look at the testing set, the less convincing we are.
– Public benchmarks and their problems.

Review: Probabilities

DISCRETE PROBABILITIES

Intro We have all been exposed to "informal probabilities". However it is instructive to be more precise. The goal today is to refresh our ability to reason with probabilities in simple cases. Also to explain why and when things can get complicated, and where to find the solutions in case of need.

Discrete probability space Assume we deal with a phenomenon that involves randomness, for instance it depends on k coin tosses. Let Ω be the set of all the possible sequence of coin tosses. Each $\omega \in \Omega$ is then a particular sequence of k heads or tails.

For now, we assume that Ω *contains a finite number of elements.* We make a *discrete probability space* by defining for each $\omega \in \Omega$ a *measure* $m(\omega) \in \mathbb{R}_+$. For instance this can be a count of occurences.

Since Ω is finite we can define the *elementary probabilities* $p(\omega) \triangleq \dfrac{m(\omega)}{\sum_{\omega' \in \Omega} m(\omega')}$.

Events A subset $A \subset \Omega$ is called an "event" and we define $\mathbb{P}(A) \triangleq \sum_{\omega \in A} p(\omega)$

Event language	Set language
The event A occurs	$\omega \in A$
The event A does not occur	$\omega \notin A$; $\omega \in A^c$
Both A and B occur	$\omega \in A \cap B$
A or B occur	$\omega \in A \cup B$
Either A or B occur	$\omega \in A \cup B$ and $A \cap B = \emptyset$

Essential properties $\mathbb{P}(\Omega) = 1$; $A \cap B = \emptyset \implies \mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B)$.

Derived properties $\mathbb{P}(A^c) = 1 - \mathbb{P}(A)$; $\mathbb{P}(\emptyset) = 0$; $\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cap B)$.

Random variables A *random variable* X is a function of $\omega \in \Omega$ taking values from some other set $\mathcal{X}$.

That makes $\mathcal{X}$ a probability space as well: for $B \subset \mathcal{X}$, $\mathbb{P}_{\mathcal{X}}(B) = \mathbb{P}\{X \in B\} = \mathbb{P}\{\omega \,:\, X(\omega) \in B\}$.

We write X when we should write $X(\omega)$.
We write $\mathbb{P}\{X < x\}$ when we should write $\mathbb{P}\{\omega \,:\, X(\omega) < x\}$.
Same for more complicated predicates involving one or more variables.

We write $\mathbb{P}(A, B)$ instead of $\mathbb{P}(A \cap B)$, as in $\mathbb{P}\{X < x, Y = y\} = P(\{\omega : X(\omega) < x\} \cap \{\omega \,:\, Y(\omega) = y\})$.

We sometimes write $\mathbb{P}(X)$ to represent the function $x \mapsto \mathbb{P}(X = x)$.

Conditional probabilities Suppose we know event A occurs.

We can make A a probability space with the same measure: for each $\omega \in A$, $p_A(\omega) \triangleq \dfrac{m(\omega)}{\sum_{\omega' \in A} m(\omega')}$.

Then, for each $B \subset \Omega$, we define $\mathbb{P}(B|A) \triangleq \displaystyle\sum_{\omega \in B \cap A} p_A(\omega) = \frac{\sum_{\omega \in B \cap A} m(\omega)}{\sum_{\omega \in A} m(\omega)} = \frac{\mathbb{P}(B \cap A)}{\mathbb{P}(A)}$.

Notation: $\mathbb{P}(X|Y)$ is a function $x, y \mapsto \mathbb{P}(X = x | Y = y)$.

Classification and Pattern Recognition

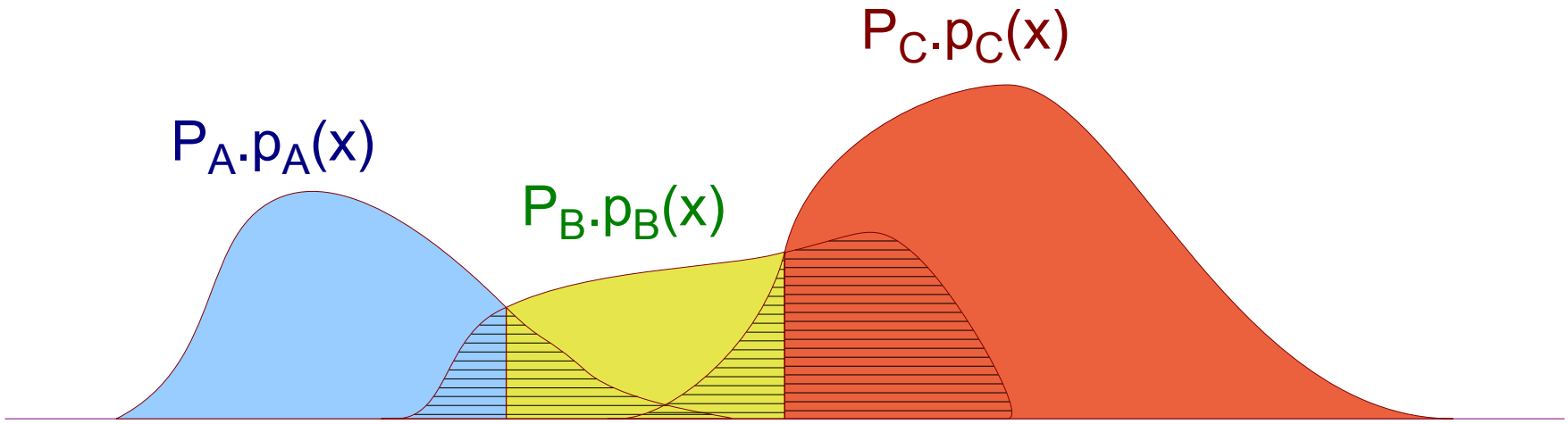
Léon Bottou

NEC Labs America

COS 424 – 2/23/2010

Bayes optimal decision rule

Comparing class densities $p_y(x)$ scaled by the class priors $P_y = \mathbb{P}\{Y = y\}$:



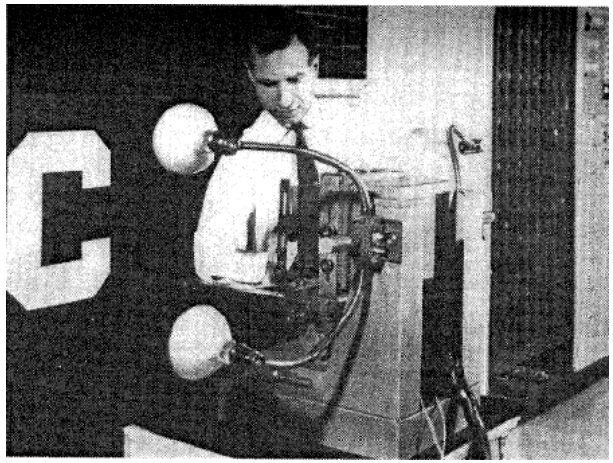
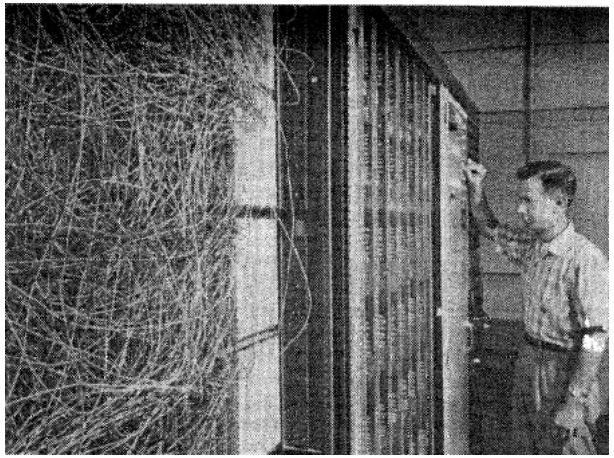
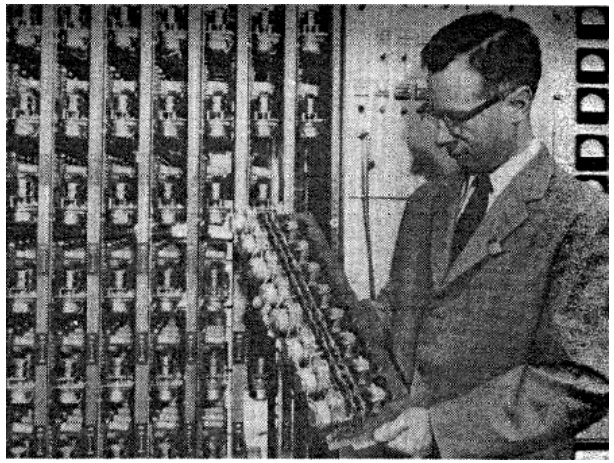
Hatched area represents the Bayes optimal error rate.

How to build a classifier from data

Given a finite set of training examples $\{(x_1, y_1), \ldots, (x_n, y_m)\}$?

- **Estimating probabilities**:
  - Find a plausible probability distribution (next lecture).
  - Compute or approximate the optimal Bayes classifier.
- **Minimize empirical error**:
  - Choose a parametrized family of classification functions a priori.
  - Pick one that minimize the observed error rate.
- **Nearest neighbours**:
  - Determine class of x on the basis of the closest example(s).

The Perceptron Mark 1 (1957)

The Perceptron is not an algorithm.

The Perceptron is a machine!

Surrogate loss functions

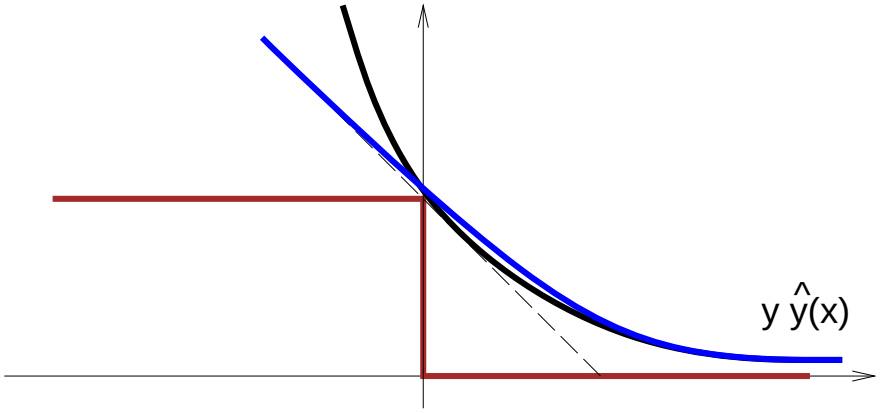
Exp loss and Log loss

Exp loss:

$$\ell(z) = \exp(-z)$$

Log loss:

$$\ell(z) = \log(1 + exp(-z))$$



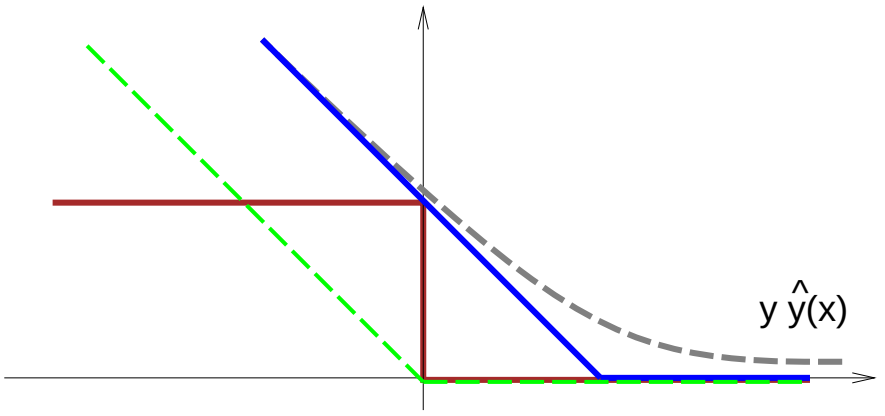
Hinges

Perceptron loss:

$$\ell(z) = \max(0, -z)$$

Hinge loss:

$$\ell(z) = \max(0, 1 - z)$$



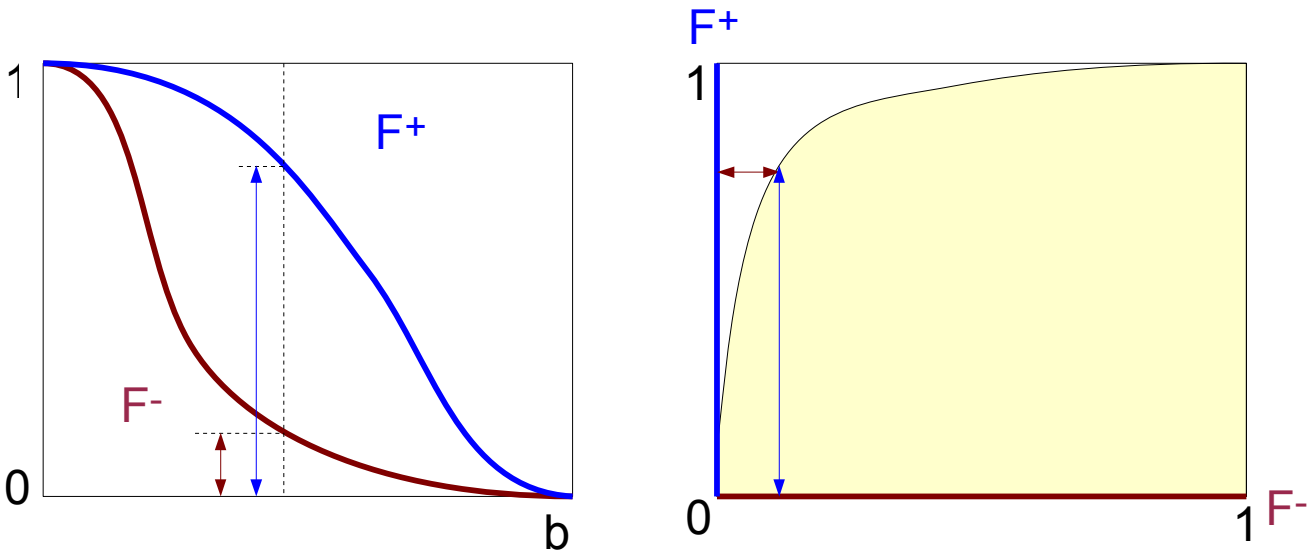
Receiver Operating Curve (ROC)

Changing the threshold

– Assigned class is $\text{sign}(f(x) - b)$.

– True positives: $F_+(b) = \mathbb{P}\{f(x) - b > 0 | Y = +1\}$

– False positives: $F_-(b) = \mathbb{P}\{f(x) - b > 0 | Y = -1\}$



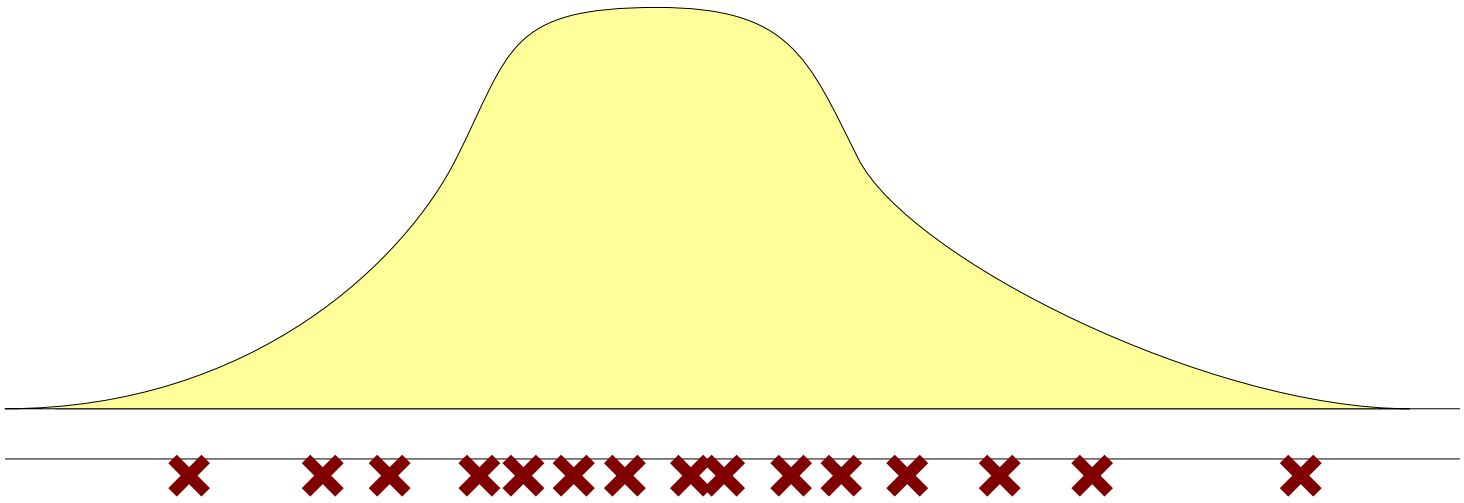
Estimating Probabilities

Léon Bottou

NEC Labs America

COS 424 – 2/25/2010

Estimating a density



Notes:

– The density is the derivative of the cumulative.

A convenient shortcut

Assume we know the distribution up to a few parameters θ.

	Discrete	Continuous
Parametric form	$\mathbb{P}\{X = x\} = f_{\theta^*}(x)$	$p(x) = f_{\theta^*}(x)$
Normalization	$\sum_x f_\theta(x) = 1$	$\int p(x)\,dx = 1$

Likelihood

– $L(\theta; x_1 \dots x_n) \triangleq \prod_{i=1}^{n} f_\theta(x_i)$

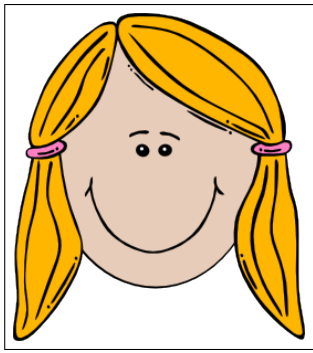
i.e. the probability of $x_1 \dots x_n$ if f_θ was the real distribution.

Maximum Likelihood Estimator (MLE)

– $\hat{\theta} \triangleq \arg\max_\theta L(\theta; x_1 \dots x_n) = \arg\max_\theta \sum_{i=1}^{n} \log f_\theta(x_i)$

Comparing estimators

Estimate $\mathbb{E}[X]$ given a sample $x_1, \ldots, x_n$.



$$\mathbb{E}[X] \approx \frac{1}{n} \sum_{i=1}^{n} x_i.$$

Jane believes in hard labor.

$$\mathbb{E}[X] \approx 3 \ !$$

Joe does not.

Is Jane's answer always better than Joe's ?

Bayesian approach: no unknown probability.

Probabilities in classical statistics

– Probabilities $\mathbb{P}\{\dots\}$ represent the unknown.

"Unknown probability distribution $\mathbb{P}\{X\}$"

"Discover something about $\mathbb{P}\{X\}$ using a sample"

"Regardless of the actual distribution..."

– Likelihoods $p_\theta(x)$ behave like probabilities but represent models.

Probabilities in Bayesian statistics

– Probabilities $\mathbb{P}\{\dots\}$ represent our beliefs.

– There are no unknown probabilities: we know what our beliefs are!

– The classical likelihood $p_\theta(x)$ is similar to the Bayesian $\mathbb{P}\{X \mid \theta\}$.

– We can have beliefs $\mathbb{P}\{\theta\}$ about θ.

Both are unfortunately represented with the same letter $\mathbb{P}$.

Putting things together

Lets use different letters:
– $\mathbb{Q}$ is the classical (unknown) probability,
– $\mathbb{P}$ is the Bayesian probability (or the classical likelihood.)

The MLE question: $\mathbb{P}\{X \mid \theta = \arg\max \mathbb{P}\{\theta \mid D\}\} \rightarrow \mathbb{Q}\{X\}$?

i.e. Is MLE consistent?

– With discrete probabilities: **yes**.
– With continuous probabilities: **often**.

The Bayesian question: $\mathbb{P}\{X \mid D\} \rightarrow \mathbb{Q}\{X\}$?

i.e. Do the priors vanish when n increases?

– With discrete probabilities: **yes**.
– With continuous probabilities: **more often than MLE**.

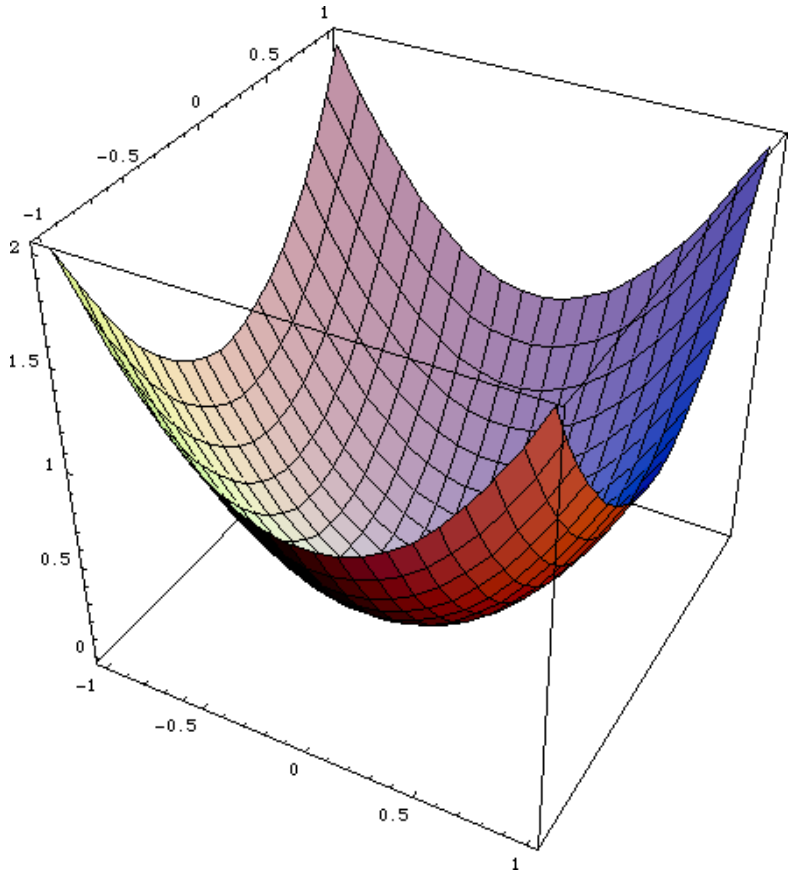
Numerical Optimization Techniques

Léon Bottou

NEC Labs America

COS 424 – 3/2/2010

Convex



Definition

$\forall\, x, y,\ \forall\, 0 \leq \lambda \leq 1,$

$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$

Property

Any local minimum is a global minimum.

Conclusion

Optimization algorithms are easy to use.

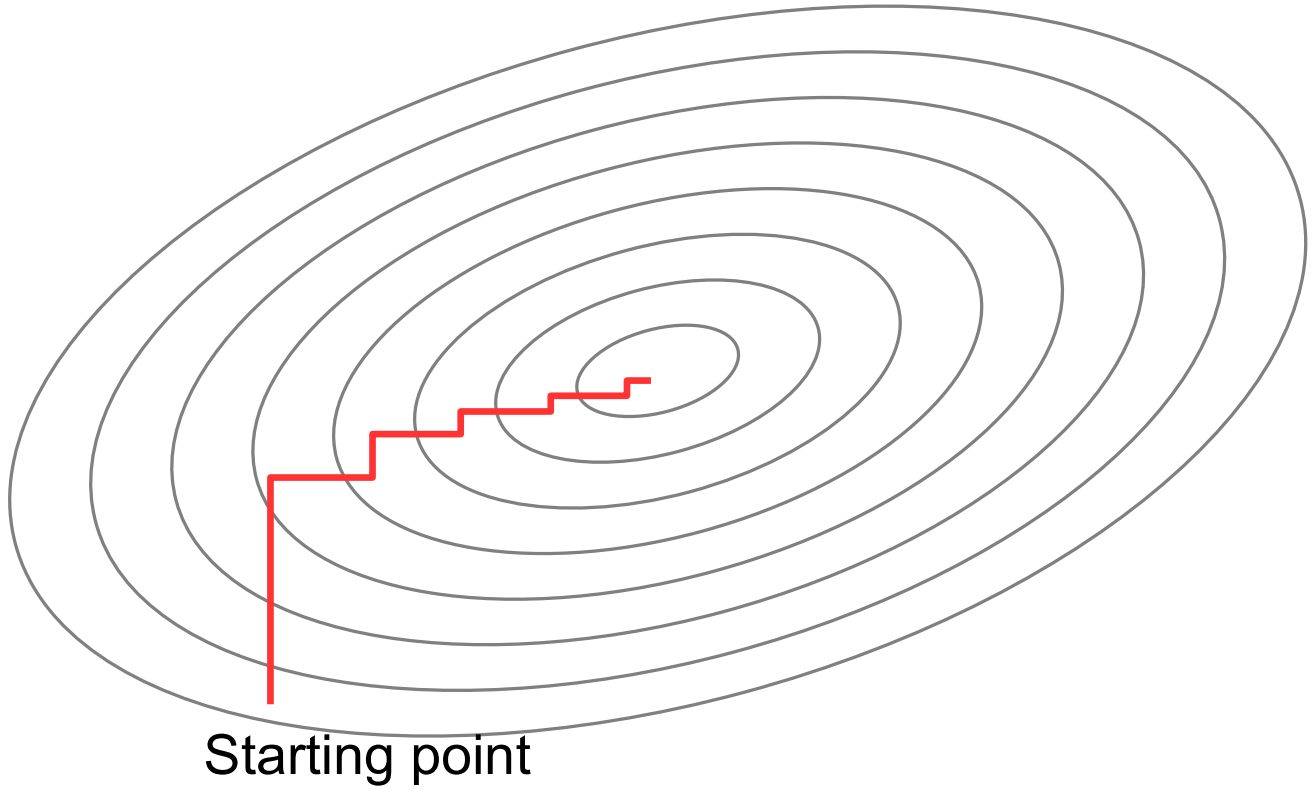
They always return the same solution.

Example: Linear model with convex loss function.

– Curve fitting with mean squared error.

– Linear classification with log-loss or hinge loss.

Coordinate Descent

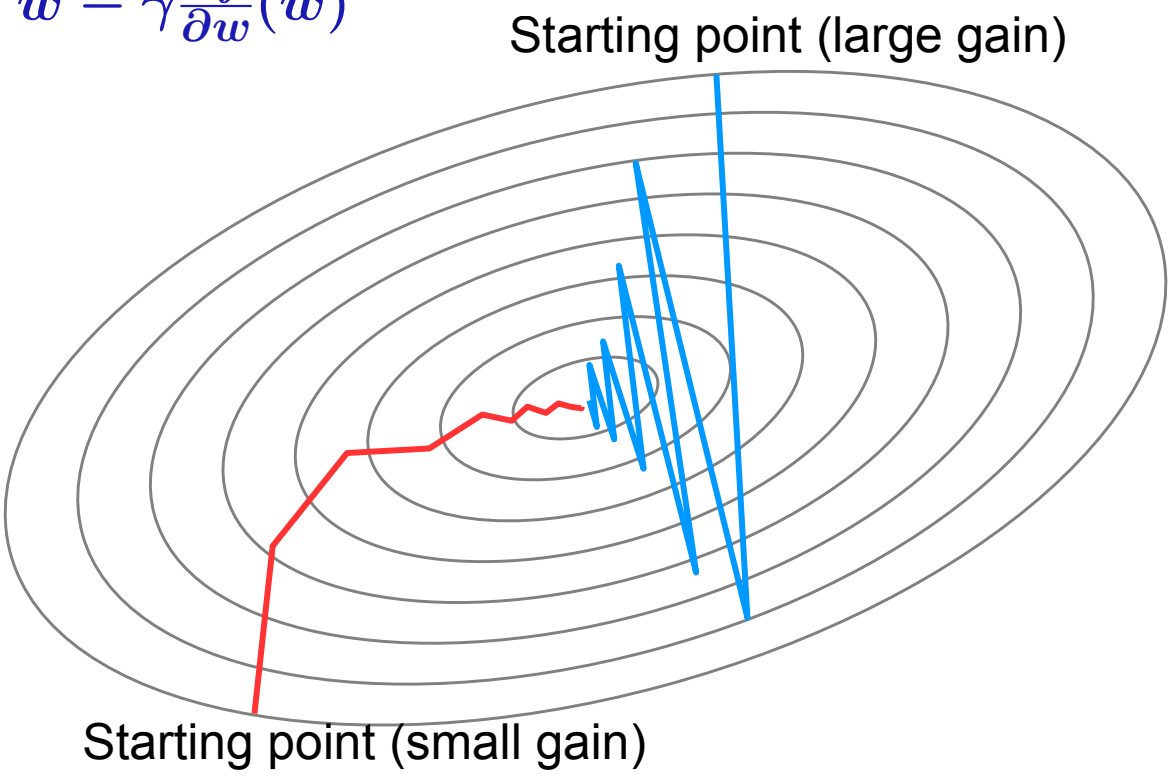


Perform successive line searches along the axes.

– Tends to zig-zag.

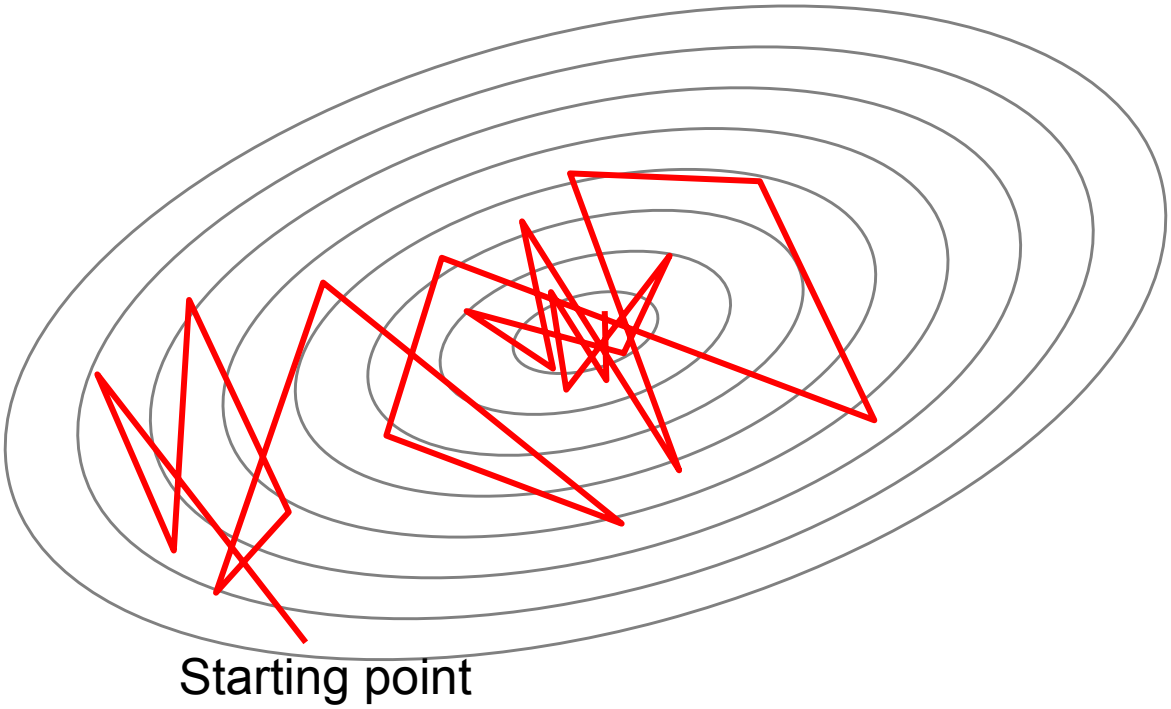
Gradient Descent

Repeat $w \leftarrow w - \gamma \frac{\partial f}{\partial w}(w)$



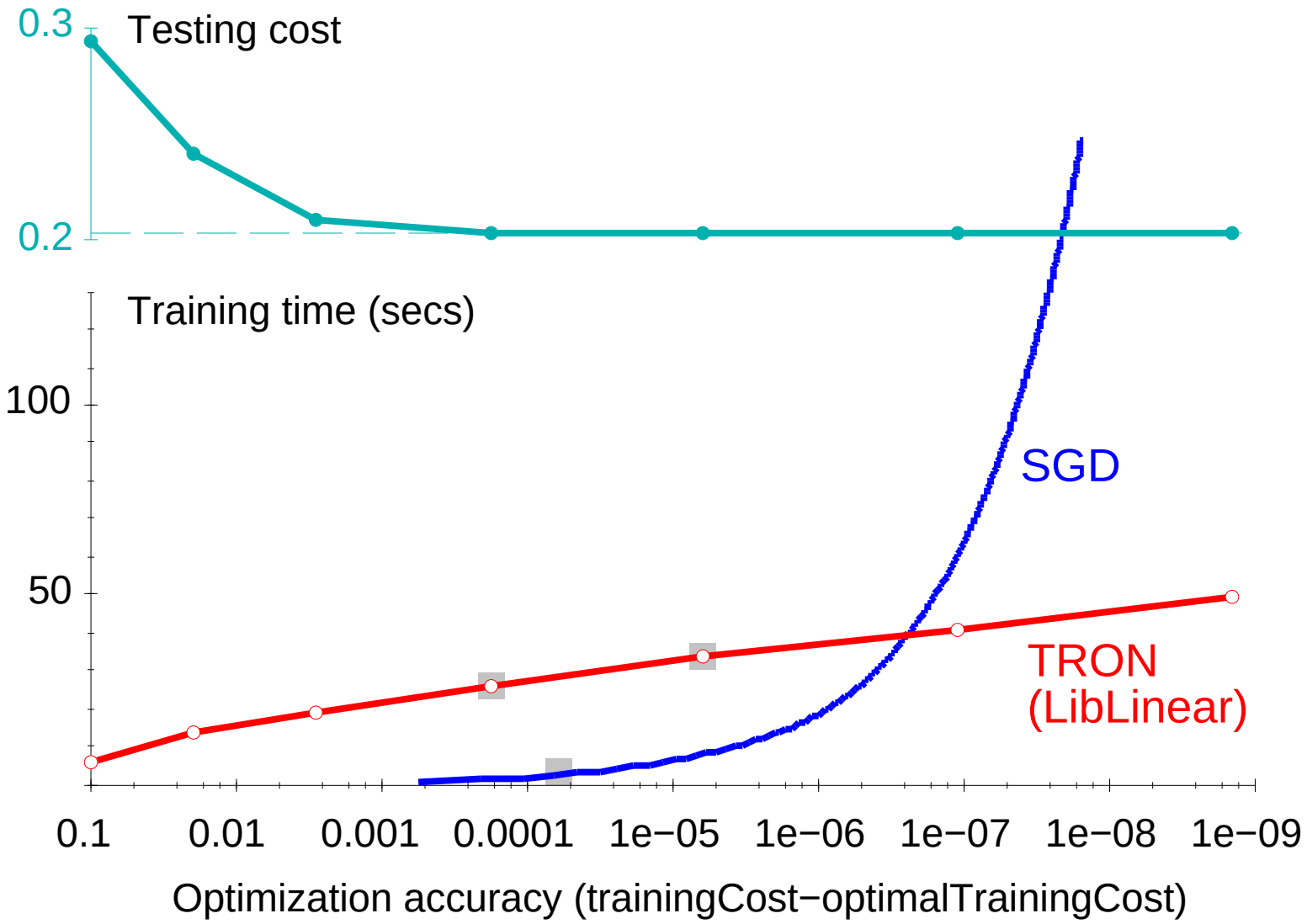
– Merge gradient and line search.
– Large gain increase zig-zag tendencies, possibly divergent.
– High curvature direction limits gain size.
– Low curvature direction limits speed of approach.

Stochastic Gradient Descent



– Very noisy estimates of the gradient.
– Gain γ_t controls the size of the cloud.
– Decreasing gains $\gamma_t = \gamma_0(1 + \lambda\gamma_0 t)^{-1}$.
– Why is it attractive?

The wall


0.3
Testing cost
0.2
Training time (secs)
100
50
SGD
TRON
(LibLinear)
0.1
0.01
0.001
0.0001
1e−05
1e−06
1e−07
1e−08
1e−09
Optimization accuracy (trainingCost−optimalTrainingCost)

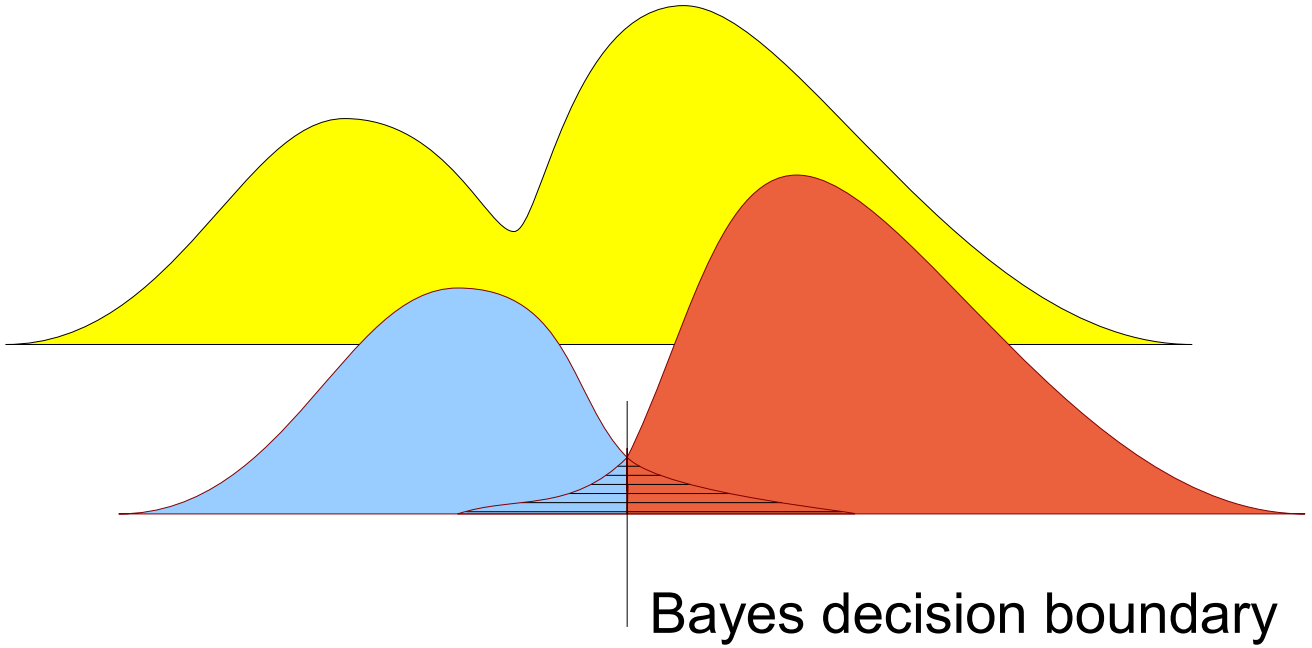
Clustering

Léon Bottou

NEC Labs America

COS 424 – 3/4/2010

What is a cluster?



Two neatly separated classes leave a trace in $\mathbb{P}\{X\}$.

Online K-Means

MacQueen's algorithm

initialize centroids w_k and $n_k = 0$.

repeat

- pick an observation x_t and determine cluster $s_t = \arg\min_k \|x_t - w_k\|^2$.
- update centroid s_t: $n_{s_t} \leftarrow n_{s_t} + 1$. $\quad w_{s_t} \leftarrow w_{s_t} + \frac{1}{n_{s_t}}(x_t - w_{s_t})$.

until satisfaction.

Comments

– MacQueen's algorithm finds decent clusters much faster.

– Final convergence could be slow. Do we really care?

– Just perform one or two passes over the randomly shuffled observations.

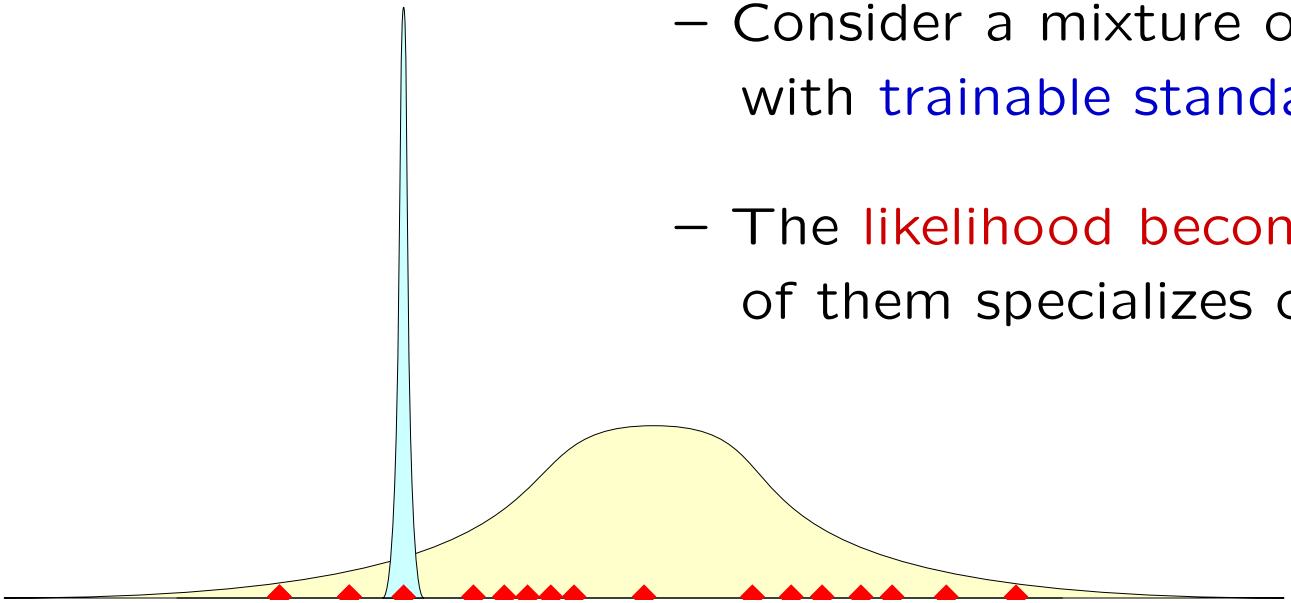
Expectation-Maximization

Léon Bottou

NEC Labs America

COS 424 – 3/9/2010

When Maximum Likelihood fails

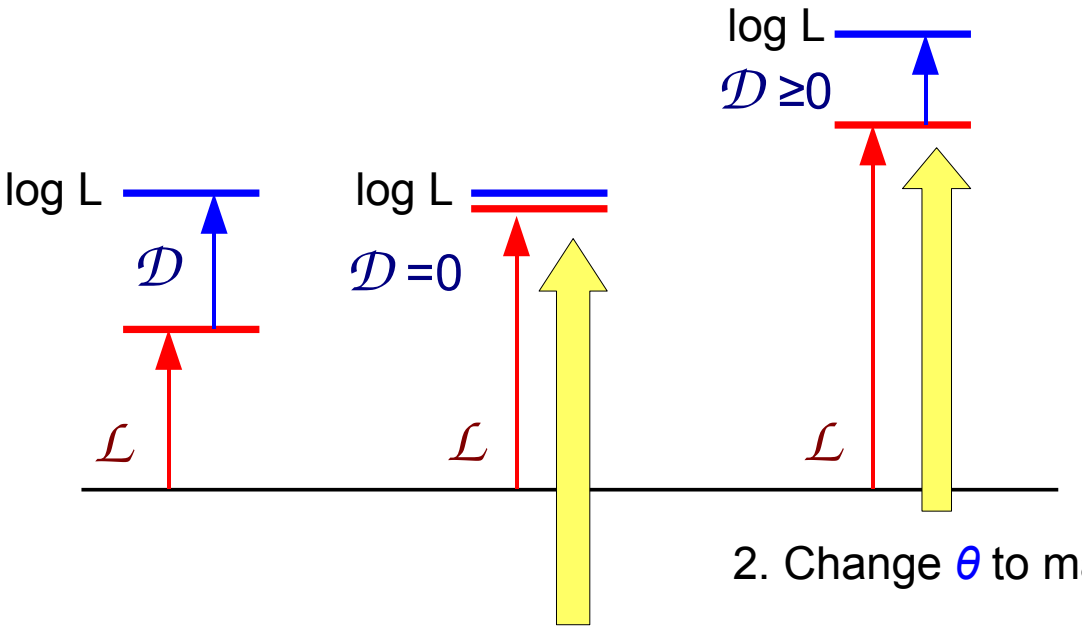


– Consider a mixture of two Gaussians with trainable standard deviations.

– The likelihood becomes infinite when one of them specializes on a single observation.

– MLE works for all discrete probabilistic models and for some continuous probabilistic models.
– This simple Gaussian mixture model is not one of them.
– People just ignore the problem and get away with it.

Expectation-Maximization



2. Change θ to maximize $\mathcal{L}$. Meanwhile $\mathcal{D}$ can only increase.

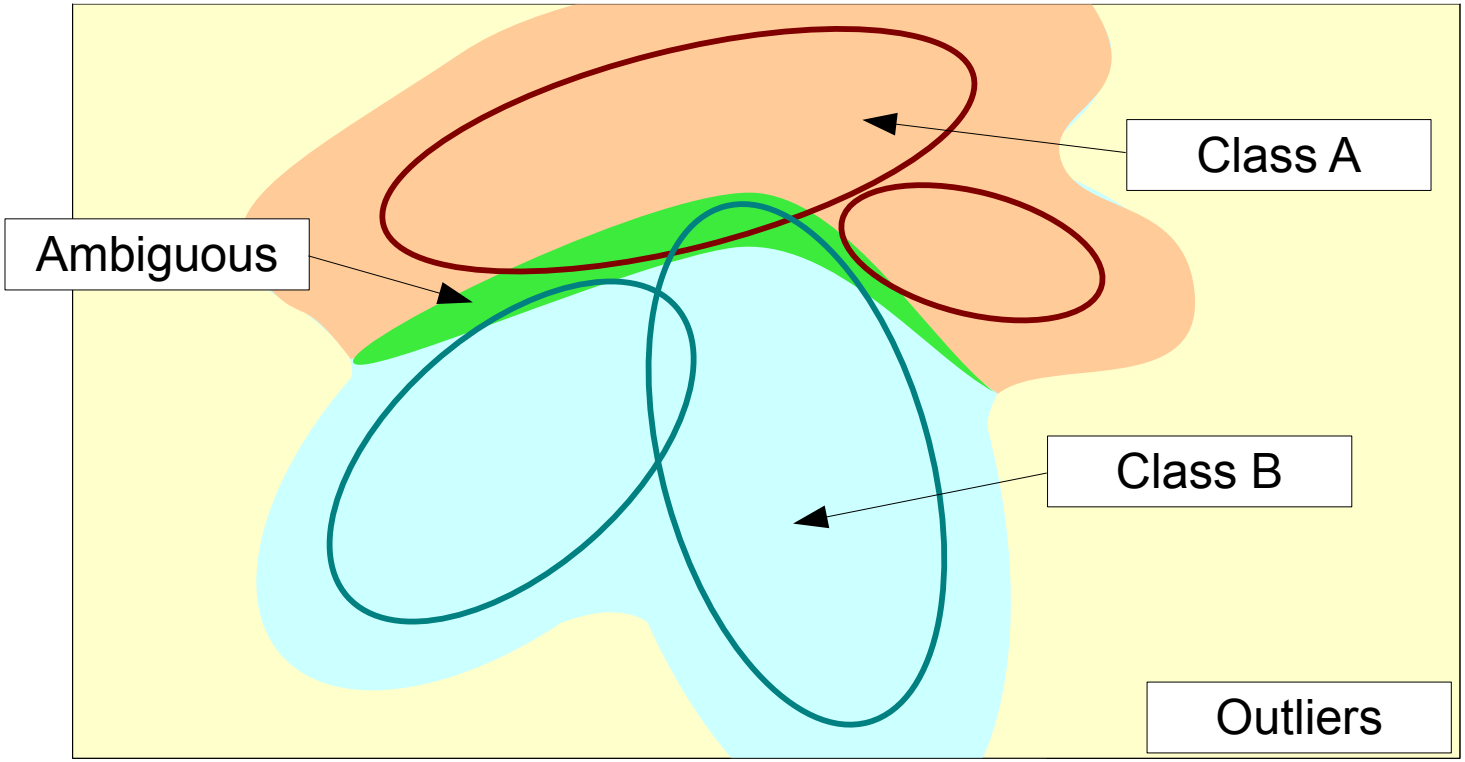
1. Change Q to minimize $\mathcal{D}$ leaving log L unchanged.

E-Step: $$q_{ik} \leftarrow \frac{\lambda_k}{\sqrt{|\Sigma_k|}} e^{-\frac{1}{2}(x_i-\mu_k)^\top \Sigma_k^{-1}(x_i-\mu_k)}$$ *remark: normalization!.*

M-Step: $$\mu_k \leftarrow \frac{\sum_i q_{ik}\, x_i}{\sum_i q_{ik}} \quad \Sigma_k \leftarrow \frac{\sum_i q_{ik}(x_i-\mu_k)(x_i-\mu_k)^\top}{\sum_i q_{ik}} \quad \lambda_k \leftarrow \frac{\sum_i q_{ik}}{\sum_{iy} q_{iy}}$$

GMM for classification

1. Model $\mathbb{P}\{X \mid Y = y\}$ for every class with a GMM.
2. Calulate Bayes optimal decision boundary.
3. Possibility to detect outliers and ambiguous patterns.



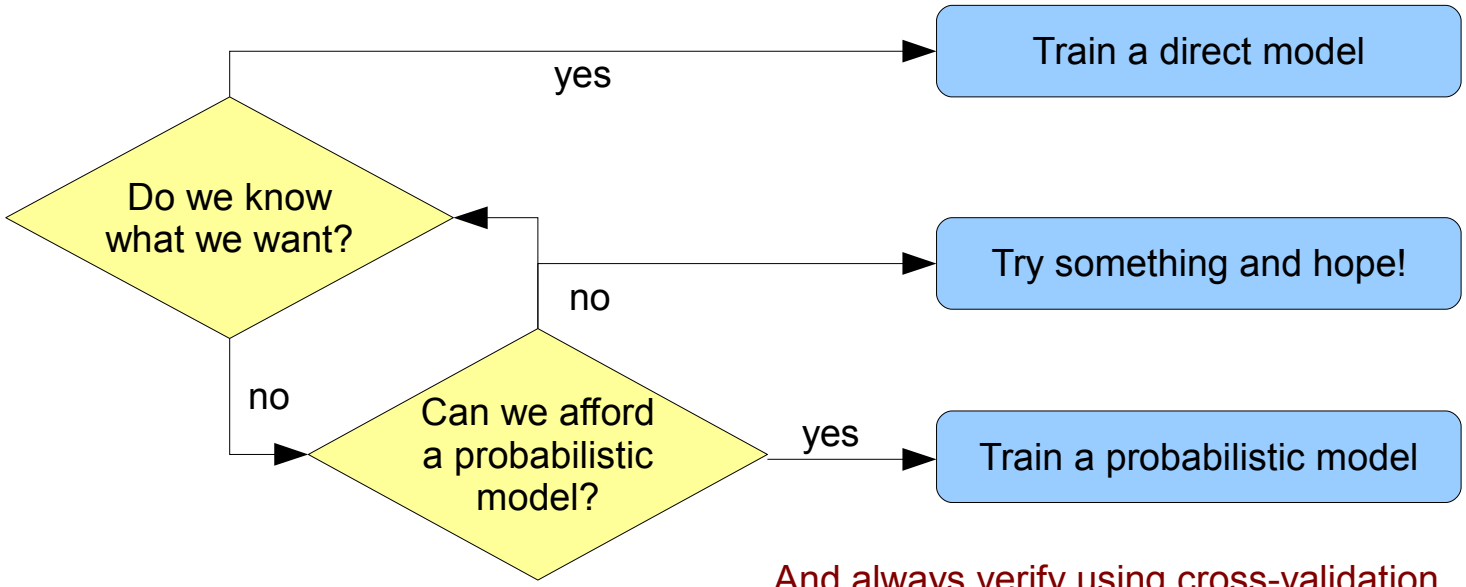
Conclusion

Expectation Maximization

– EM is a very useful algorithm for probabilistic models.
– EM is an alternative to sophisticated optimization
– EM is simpler to implement.

Probabilistic Models

– More versatile than direct approaches.
– More demanding than direct approaches (assumptions, data, etc.)



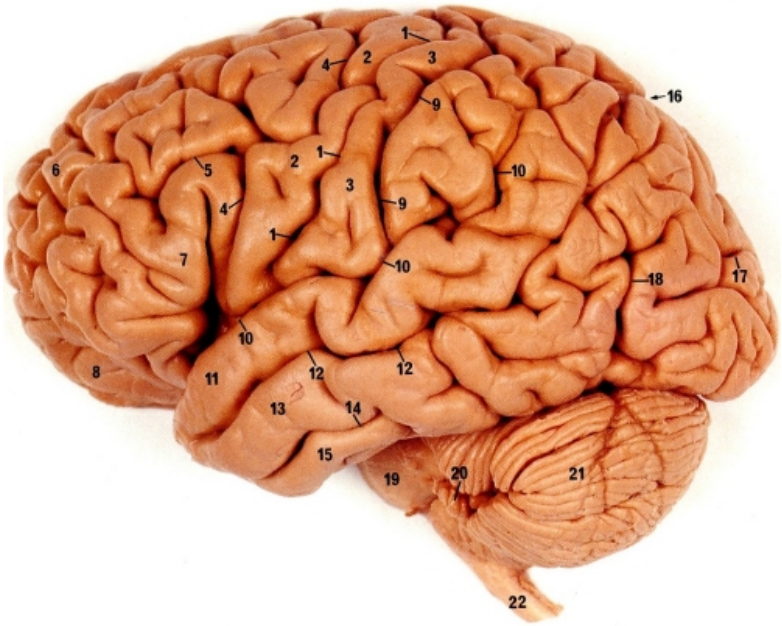
And always verify using cross-validation.

Multilayer Networks

Léon Bottou

COS 424 – 3/11/2010

An engineering perspective on the brain



The brain as a computer

– Compact
– Energy efficient (20 Watts)
– Amazingly good for perception and informal reasoning.

Bill of materials

≈ 90%: support, energy, cooling.
≈ 10%: signalling wires.

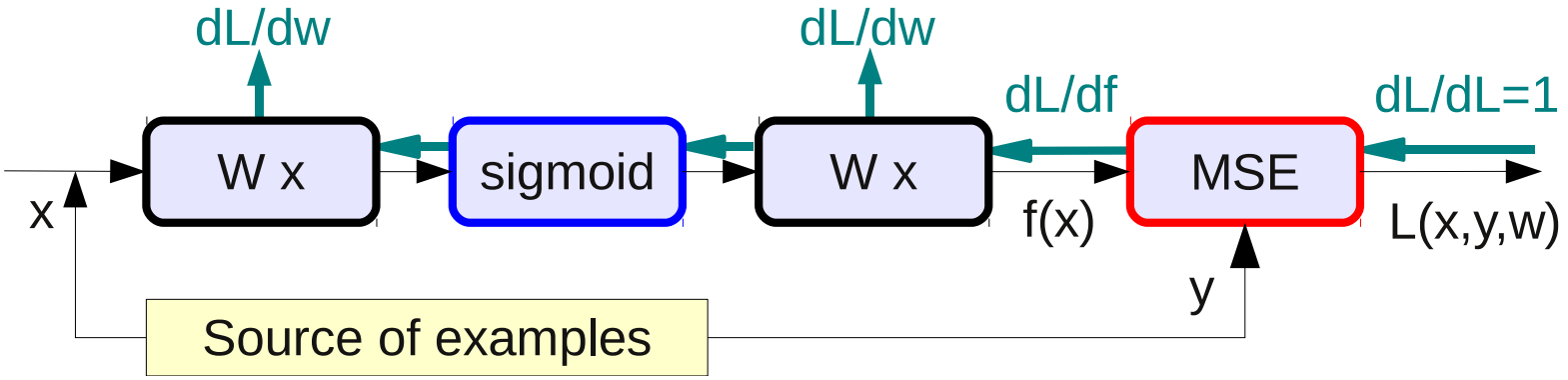
A lot of wires in a small box

– Severe wiring constraints force a very specific architecture.
– Local connections (98%) vs. long distance connections (2%).
– Layered structure (at least in the visual system.)
– This is not a universal machine!
– But this machine defines what we belive is interesting!

Back-propagation algorithm

Backward pass in the two layer network

– Set $dL/dL = 1$, compute gradients dL/dy and dL/dw for all boxes.



Update weights

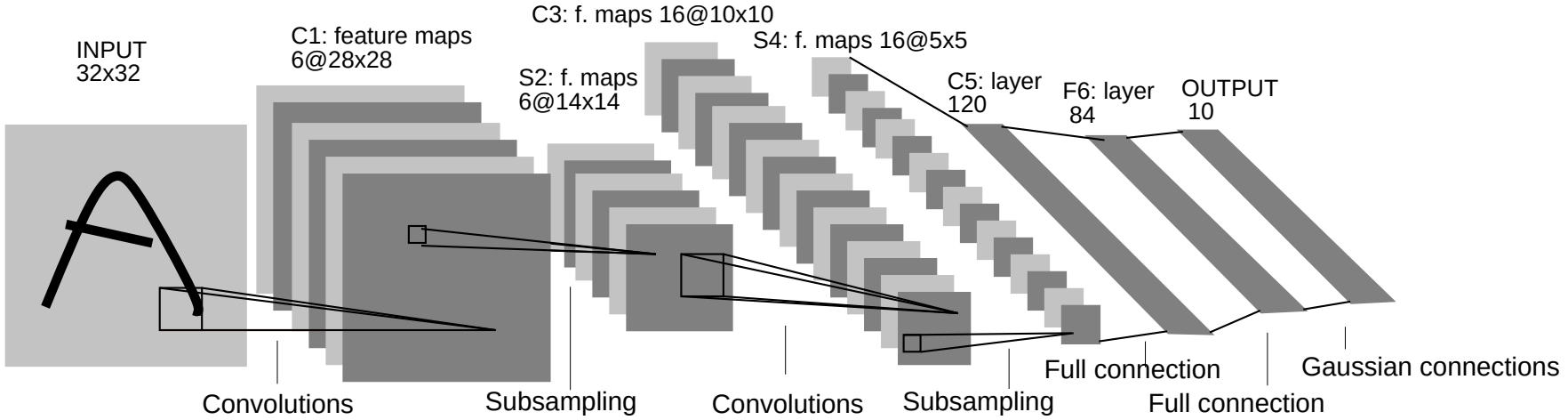
– For instance with a stochastic gradient update.

$$w \leftarrow w - \gamma_t \frac{\partial L}{\partial w}(x, y, w) \, .$$

CNNs for image analysis

2D Convolutional Neural Networks

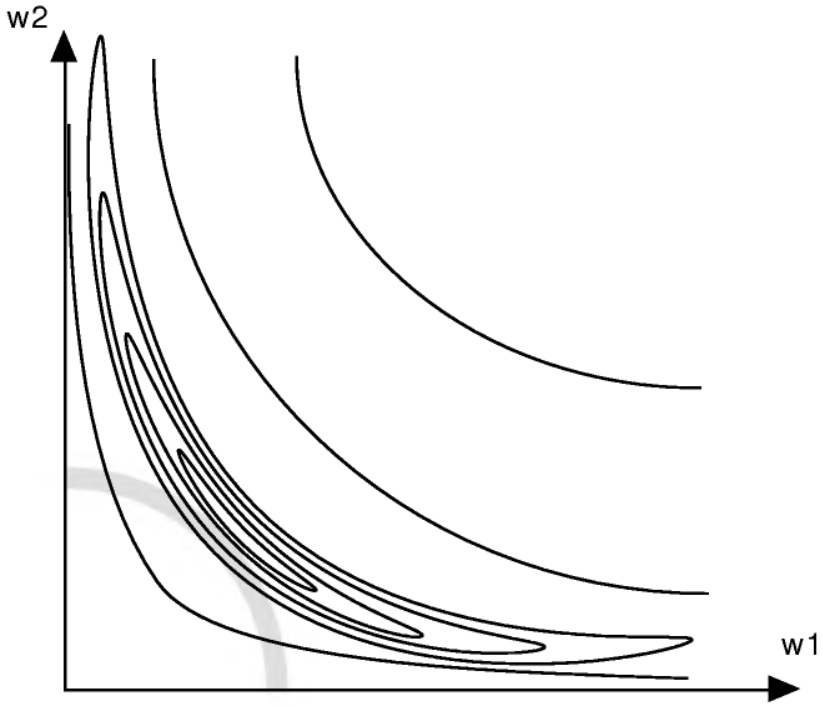
– 1989: isolated handwritten digit recognition
– 1991: face recognition, sonar image analysis
– 1993: vehicle recognition
– 1994: zip code recognition
– 1996: check reading



Optimisation for multilayer network

Landscape

– Ravine along $w_1\, w_2 = 1$.
– Massive saddle point near the origin.
– Mountains in the quadrants $w_1\, w_2 < 0$.
– Plateaux in the distance.



Tricks of the trade

– How to initialize the weights?
– How to avoid the great saddle point?
– etc.

Descriptive and Exploratory Methods

Léon Bottou

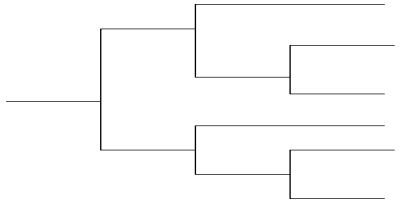
largely copied from Mireille Summa-Gettler lectures (in french)

COS 424 – 3/23/2010

A catalog of descriptive methods

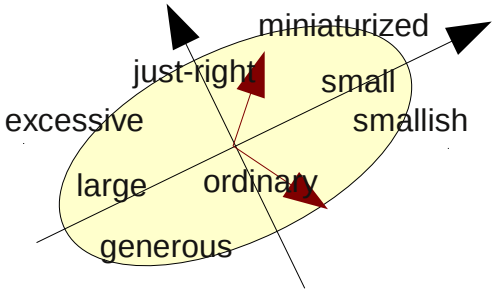
Clustering methods

– K-means, K-medioids, Gaussian mixtures. . .
– Hierarchical clustering. . .



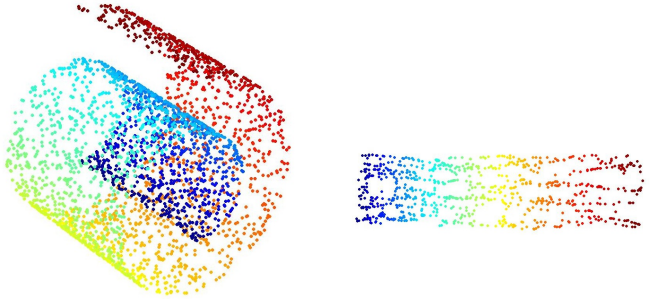
Projection methods

– Principal component analysis (PCA) [Hotelling, 30s]
– Correspondence analysis (CA) [Benzecri, 60s]
– Multiple correspondence analysis (MCA)
– Canonical correlation analysis (CCA), . . .

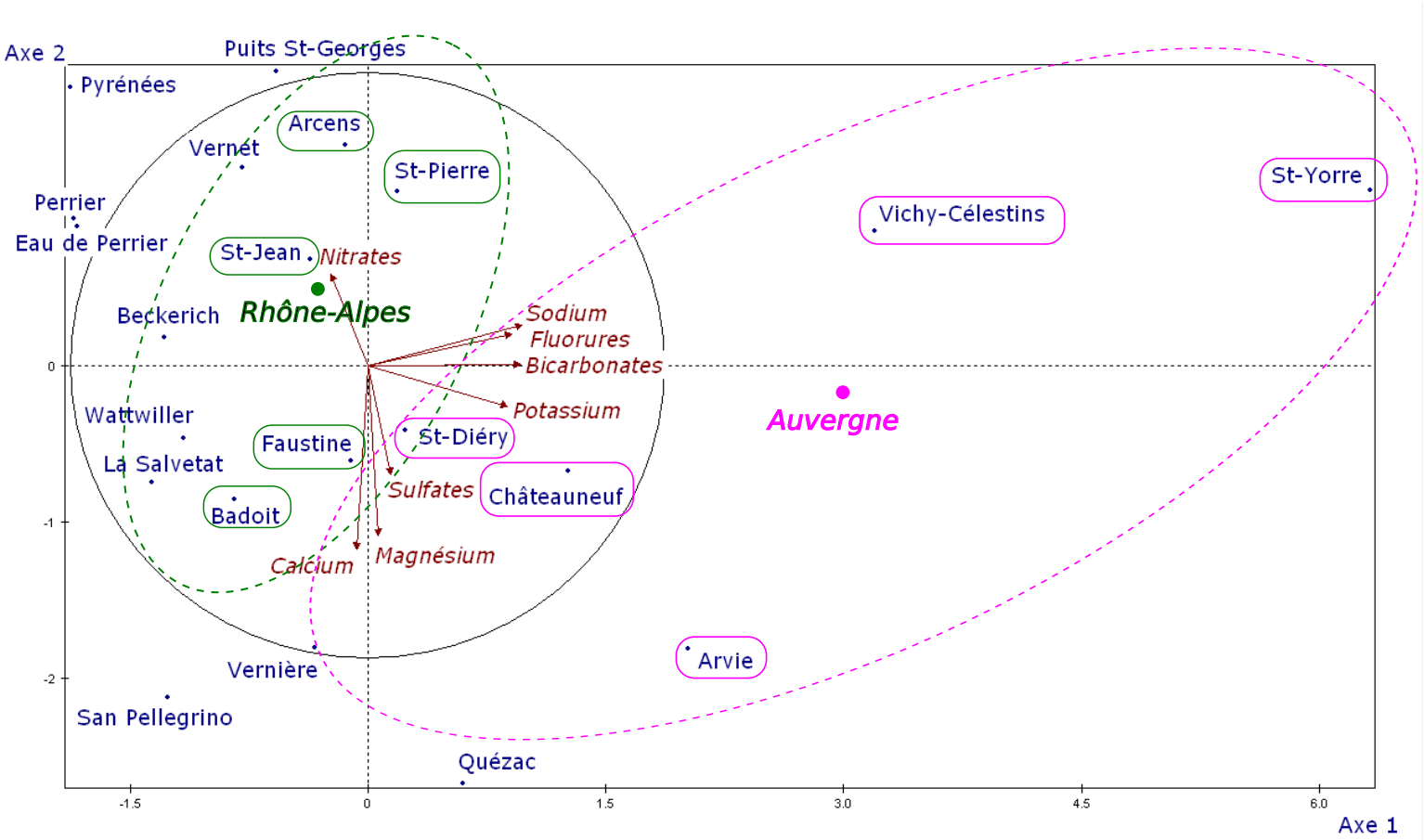


Embedding methods

– Kernel PCA
– Locally linear embbedding (LLE)
– ISOMAP



Groups are often more interesting



– The barycenter of the six "Rhônes-Alpes" springs is close to the origin.

– The barycenter of the five "Auvergne" springs is high on the first axis.

Hair and eye colors

We know for 592 english women

– the color of their eyes

– the color of their hair.

n

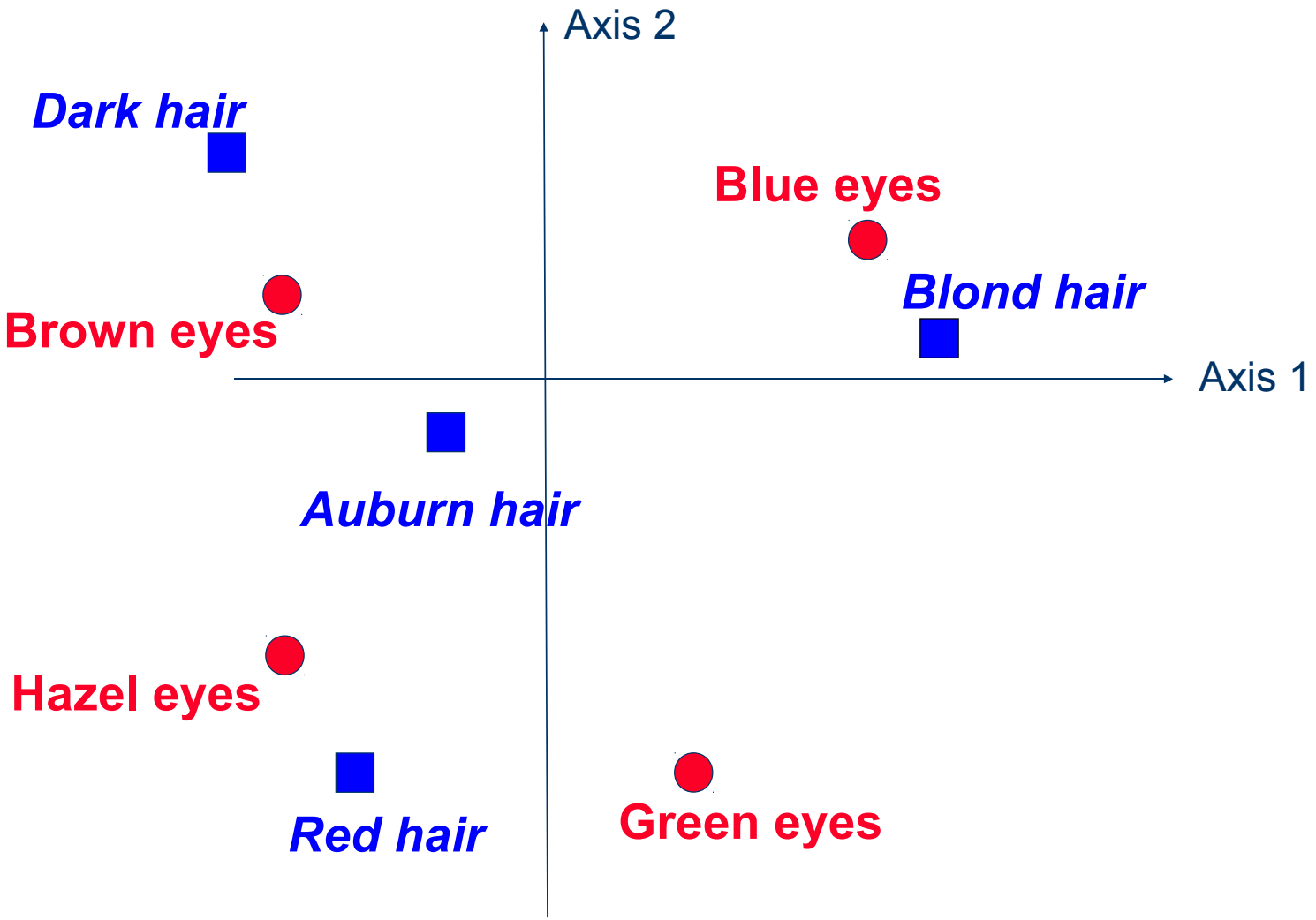
		Hair color				
		Dark	Auburn	Red	Blond	*Totals*
Eyes color	Brown	68	119	26	7	***220***
	Hazel	15	54	14	10	***93***
	Green	5	29	14	16	***64***
	Blue	20	84	17	94	***215***
	Totals	***108***	***286***	***71***	***127***	***592***

p

Contingency table $[x_{ij}]$

$$x_{i\bullet} = \sum_{j=1}^{p} x_{ij} \qquad x_{\bullet j} = \sum_{i=1}^{n} x_{ij} \qquad x_{\bullet\bullet} = \sum_{i=1}^{n} \sum_{j=1}^{p} x_{ij}$$

Simultaneous representation



Semiometric plane (2,6)

Generalization and Capacity Control

Léon Bottou

COS 424 – 3/30/2010

Decomposition for the general case

$$\mathrm{Fr}\left\{\mu_T - \mu_L > \epsilon\right\} =$$

$$\sum_{L,S} \frac{1}{N_{\text{splits}}} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \times \begin{pmatrix} \mathbb{1}\left\{\mu_T(m_1) - \mu_L(m_1) > \epsilon\right\} \\ \mathbb{1}\left\{\mu_T(m_2) - \mu_L(m_2) > \epsilon\right\} \\ \vdots \\ 0 \\ 1 \\ \vdots \\ \mathbb{1}\left\{\mu_T(m_{\mathcal{N}}) - \mu_L(m_{\mathcal{N}}) > \epsilon\right\} \end{pmatrix}$$

- The sum runs over all the possible splits.
- The green vector indicates which error vector is produced by the classifier returned by running learning algorithm on that split.
- The purple vector indicates which error vectors have an error deviation greater than ϵ.

The infinite case

Vapnik-Chervonenkis combinatorial lemma

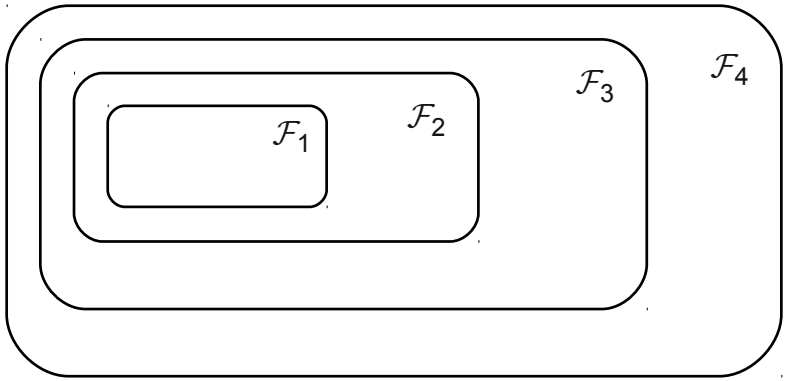
Let $m_{\mathcal{F}}(l) = \max_{\{x_1, c_1 \ldots x_l, c_l\}} \mathcal{N}(\mathcal{F}, \{x_1, c_1 \ldots x_l, c_l\})$.

- Either $m_{\mathcal{F}}(l) = 2^l$ for all l.
- Or $m_{\mathcal{F}}(l) \leq \left(\frac{le}{h}\right)^h$ where h is the last value such that $m_{\mathcal{F}}(h) = 2^h$.

– Quantity h is called the Vapnik-Chervonenkis of the family $\mathcal{F}$.
It measures the "capacity" of a family of functions.

(Vapnik and Chervonenkis, 1968) (Sauer, 1972)
Some people unfairly call this lemma "Sauer's lemma".

What is a "structure"



The structure defines a preorder on the functions.

All other things being equal:
– We'll prefer a function from $\mathcal{F}_1$ over a function of $\mathcal{F}_2$.
– We'll prefer a function from $\mathcal{F}_2$ over a function of $\mathcal{F}_3$.
– We'll prefer a function from $\mathcal{F}_3$ over a function of $\mathcal{F}_4$.
– etc.

Very similar to a Bayesian prior!

III. Learning algorithms in little pieces

Support Vector Machines

Léon Bottou

COS 424 – 4/1/2010

Primal and dual formulation

Karush-Kuhn-Tucker theory

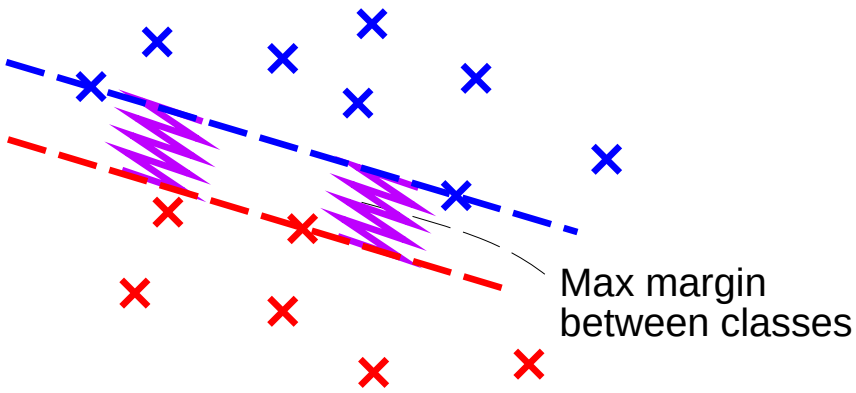
– Refined theory for convex otimization under constraints.

– Construct a *dual optimization problem*

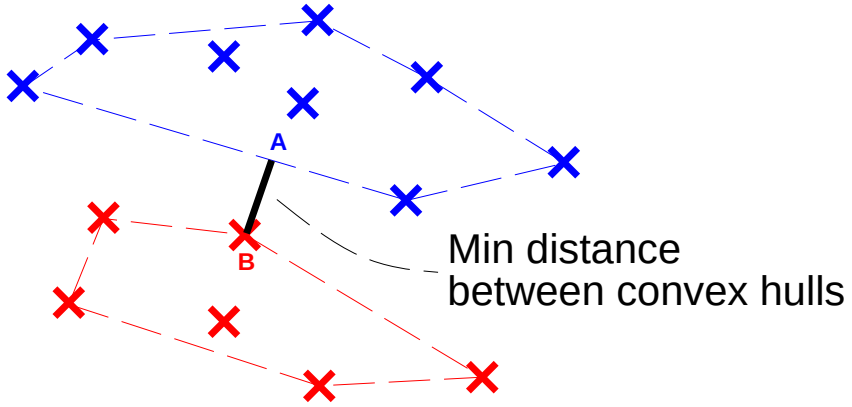
whose constraints are simpler,

and whose solution is related to the solution we seek.

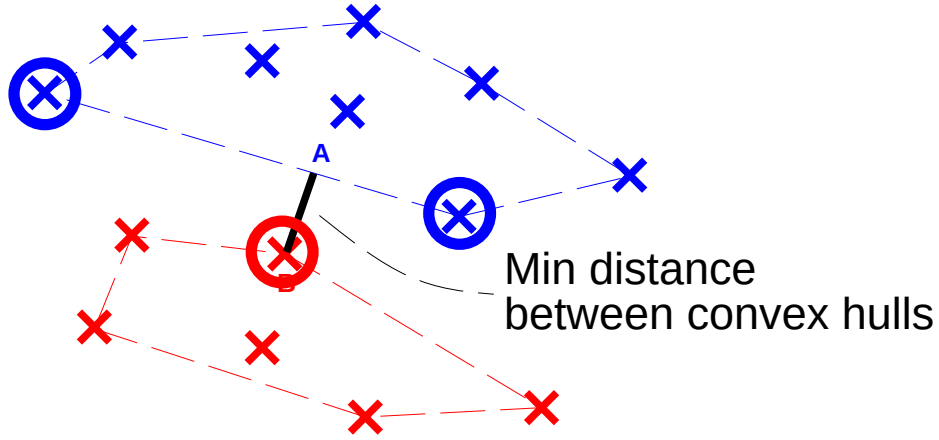
Primal formulation



Dual formulation



Support Vectors Machines



$$\min_{\boldsymbol{\beta}} \sum_{ij} y_i y_j \, \beta_i \beta_j \, \mathbf{x}_i^\top \mathbf{x}_j \quad \text{subject to} \quad \begin{cases} \forall i \quad \beta_i \geq 0 \\ \sum_i y_i \beta_i = 0 \\ \sum_i \beta_i = 2 \end{cases}$$

The only non zero β_i are those corresponding to support vectors.

Quadratic Kernel

Quadratic basis

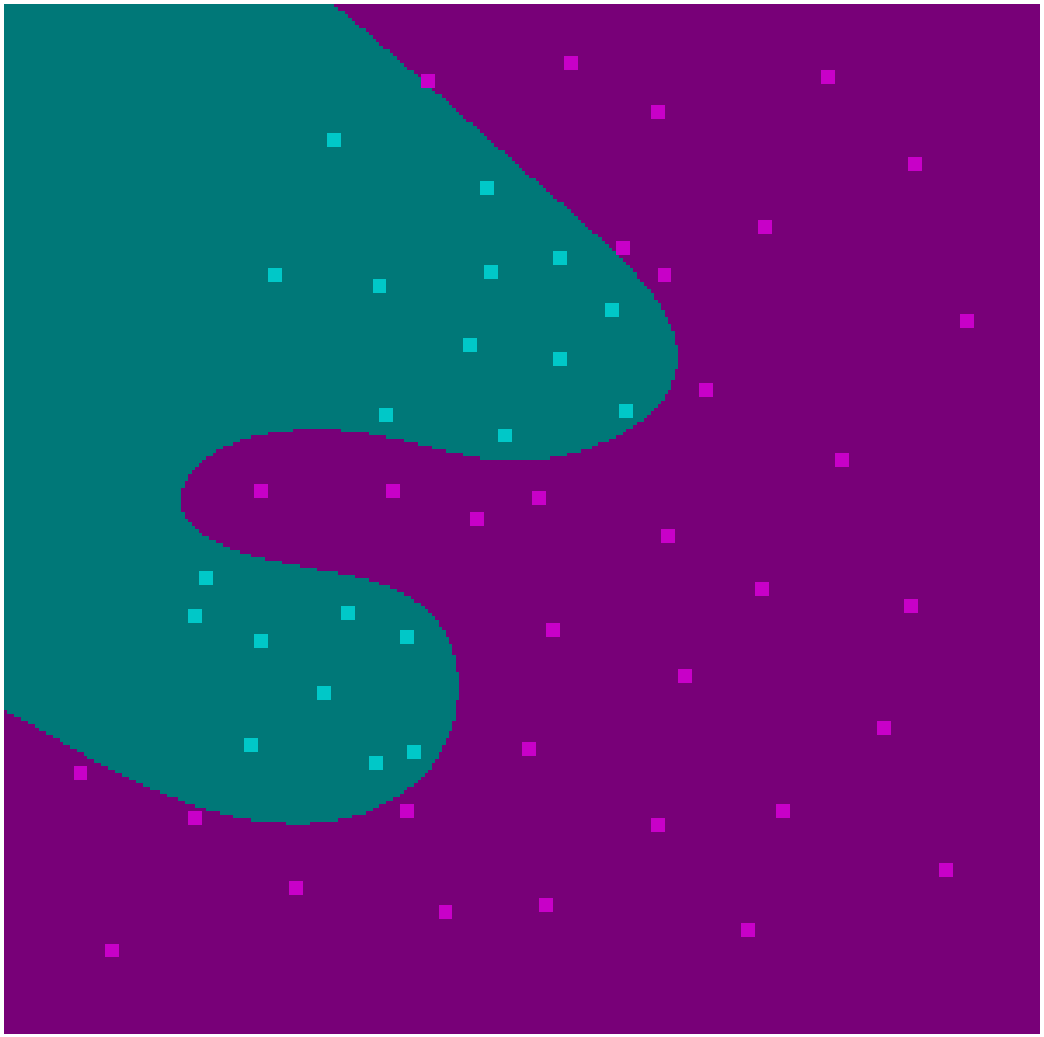
$$\Phi(\mathbf{x}) = \left(\left[x_i \right]_i , \left[x_i^2 \right]_i , \left[\sqrt{2}\, x_i x_j \right]_{i<j} \right)$$

Dot product

$$\Phi(\mathbf{x})^\top \Phi(\mathbf{v}) = \sum_i x_i v_i + \sum_i x_i^2 v_i^2 + \sum_{i<j} 2\, x_i v_i x_j v_j$$

– Are there $\boldsymbol{d(d+3)/2}$ terms to add ?

Radial Basis (gamma = 1)



Outlook

Success stories

– Text categorization

– Classification tasks in general
  the best classifier can change a lot,
  but the SVM is rarely far away.

Weak points

– Computationally costly with noisy data

– L2 regularization works poorly when irrelevant inputs abound.

Information Theory, Statistics, and Decision Trees

Léon Bottou

COS 424 – 4/6/2010

Quantity of information

Optimal code length: $l_i = -\log_r(p_i)$.

Optimal expected code length: $\sum p_i \, l_i = -\sum p_i \log_r(p_i)$.

Receiving a message x with probability p_x:

– The *acquired information* is $h(x) = -log_2(p_x)$ bits.

– An informative message is a surprising message!

Expecting a message X with distribution $p_1 \ldots p_n$:

– The *expected information* is $H(X) = -\sum_{x \in \mathcal{X}} p_x \log_2(p_x)$ bits.

– This is also called *entropy*.

These are two distinct definitions!

Note how we switched to logarithms in base two.
This is a multiplicative factor: $\log_2(p) = \log_r(p) \log_2(r)$.
Choosing base 2 defines a unit of information: the bit.

Questions

Many questions can distinguish cars

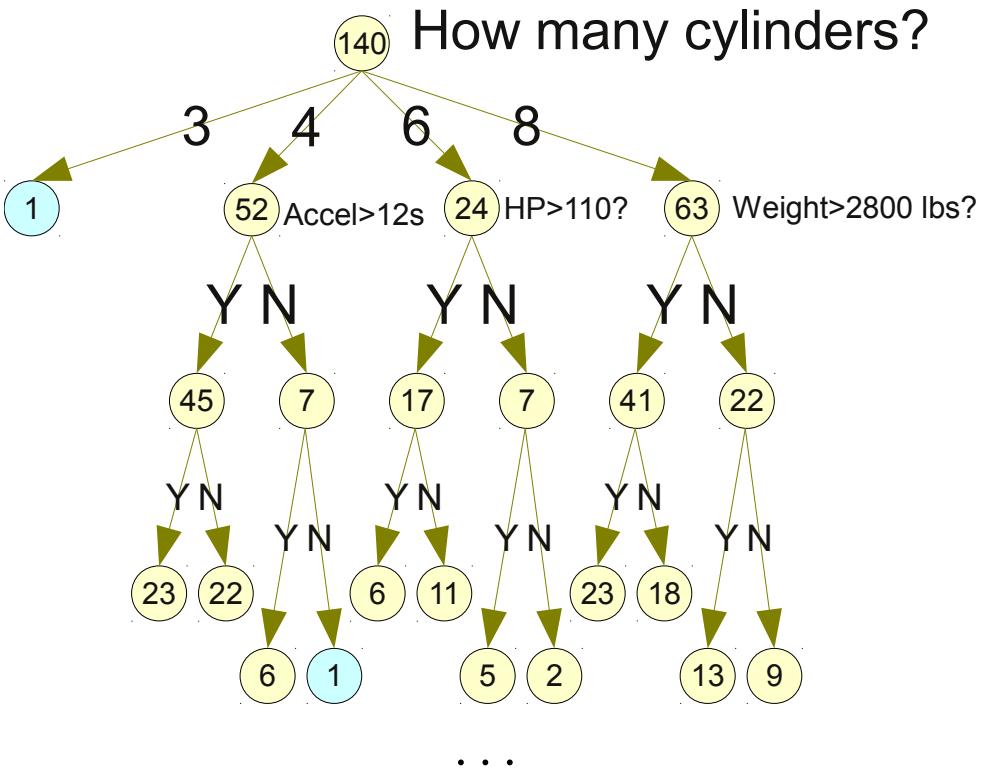
– How many cylinders? (3,4,5,8)
– Displacement greater than 200 cu in? (yes, no)
– Displacement greater than x cu in? (yes, no)
– Weight greater than x lbs? (yes, no)
– Model name longer than x characters (yes, no)
– etc. . .

Which question brings the most information about the task?

– Build contingency table.
– Compare mutual informations $I(Question, Mpg > 19)$.

	Possible answers			
	ansA	**ansB**	**ansC**	**ansD**
mpg>19	12	23	65	5
mpg≤19	18	12	4	4

Decision trees



Then label each leaf with class $MPG > 19$ or $MPG \leq 19$.

We can now say if a car does more than 19mpg by asking a few questions.

But that is learning by heart!

Revisiting decision trees : likelihoods

The tree as a model of $P(Y|X)$

– Estimate $P(Y|X)$ by the target frequencies in the leaf for X.

– We can compute the likelihood of the data in this model.

Likelihood gain when splitting a node

– Let x_{ij} be the contingency table for a node and a question.

– Splitting the node with a question increases the likelihood:

$$\begin{aligned}
\log L_{after} - \log L_{before} &= \sum_{ij} x_{ij} \log \frac{x_{ij}}{x_{\bullet j}} - \sum_{i} x_{i\bullet} \log \frac{x_{i\bullet}}{x_{\bullet\bullet}} \\
&= \sum_{ij} x_{ij} \log \frac{x_{ij}\, x_{\bullet\bullet}}{x_{\bullet\bullet}\, x_{\bullet j}} - \sum_{i} x_{i\bullet} \log \frac{x_{i\bullet}}{x_{\bullet\bullet}} \\
&= \sum_{ij} x_{ij} \log \frac{x_{ij}}{x_{\bullet\bullet}} - \sum_{j} x_{\bullet j} \log \frac{x_{\bullet j}}{x_{\bullet\bullet}} - \sum_{i} x_{i\bullet} \log \frac{x_{i\bullet}}{x_{\bullet\bullet}}
\end{aligned}$$

Compare with slide 19.

Ensembles

Léon Bottou

COS 424 – 4/8/2010

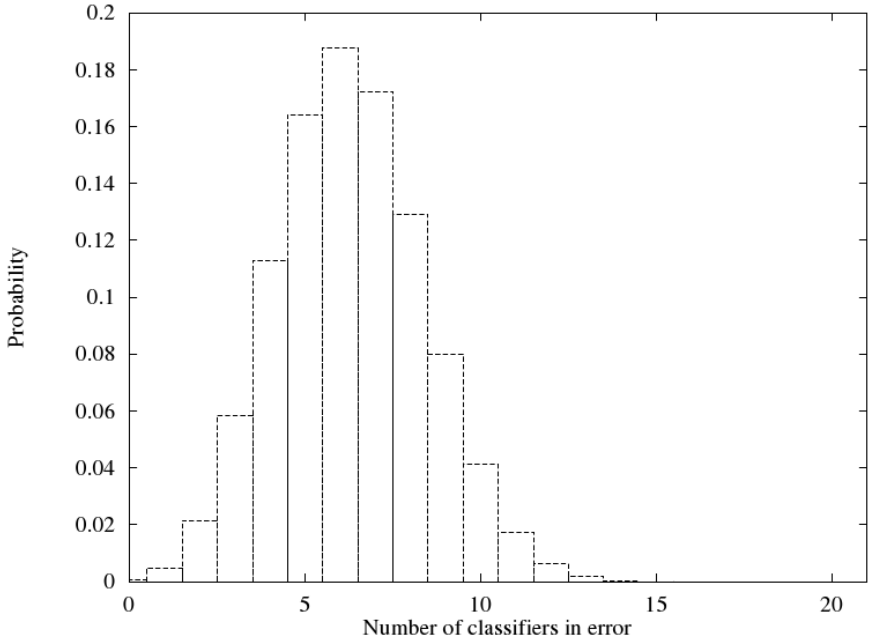
Uncorrelated classifiers

Assume $\forall r \neq s \quad \text{Cov}\left[\, \mathbb{1}\{h_r(x) = y\}\,,\, \mathbb{1}\{h_s(x) = y\}\,\right] = 0$

The tally of classifier votes follows a binomial distribution.

Example

Twenty-one uncorrelated classifiers with 30% error rate.



Adaboost

Given examples $(x_1, y_1) \ldots (x_n, y_n)$ with $y_i = \pm 1$.

Let $D_1(i) = 1/n$ for $i = 1 \ldots n$.

For $t = 1 \ldots T$ do

- Run weak learner using examples with weights D_t.
- Get weak classifier h_t
- Compute error: $\varepsilon_t = \sum_i D_t(i)\, \mathbb{1}(h_t(x_i) \neq y_i)$
- Compute magic coefficient $\alpha_t = \frac{1}{2} \log \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right)$
- Update weights $D_{t+1}(i) = \frac{D_t(i)\, e^{-\alpha_t\, y_i\, h_t(x_i)}}{Z_t}$

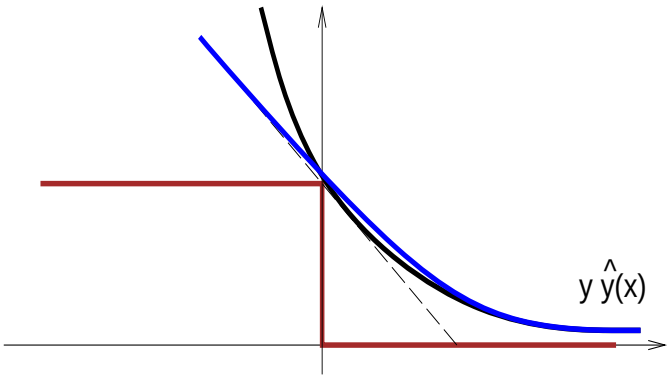
Output the final classifier $f_T(x) = \text{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right)$

Boosting and exponential loss

Proofs are instructive

We obtain the bound

$$\text{TrainingError}(f_T) \;\leq\; \frac{1}{n}\sum_i e^{-y_i H(x_i)} \;=\; \prod_{t=1}^{T} Z_t$$



- without saying how D_t relates to h_t
- without using the value of α_t

Conclusion

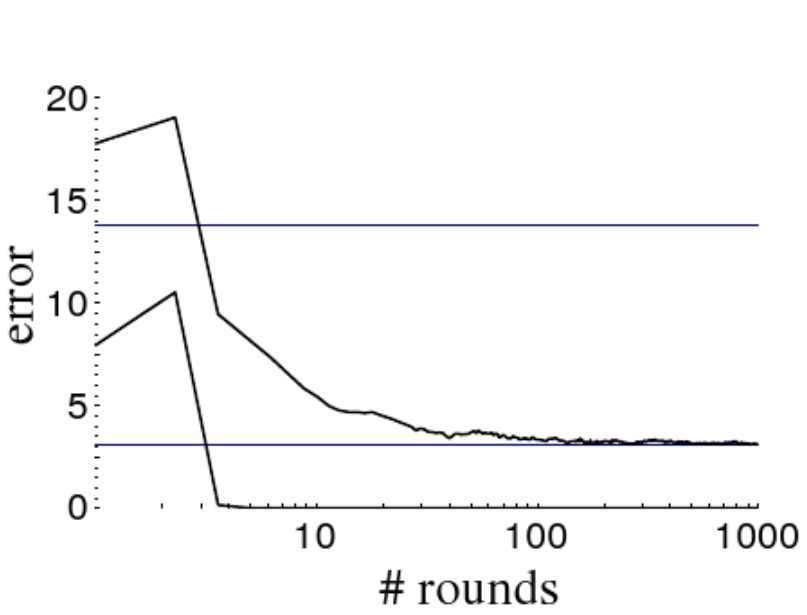
- Round T chooses the h_T and α_T that maximize the exponential loss reduction from f_{T-1} to f_T.

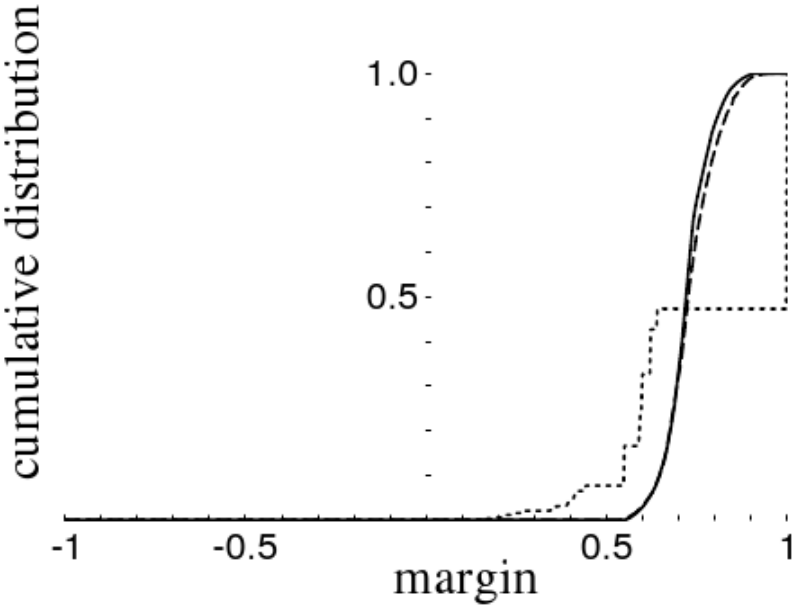
Exercise

- Tweak Adaboost to minimize the log loss instead of the exp loss.

Boosting and margins

$$margin_H(x,y) = \frac{y\,H(x)}{\sum_t |\alpha_t|} = \frac{\sum_t \alpha_t\, y\, h_t(x)}{\sum_t |\alpha_t|}$$





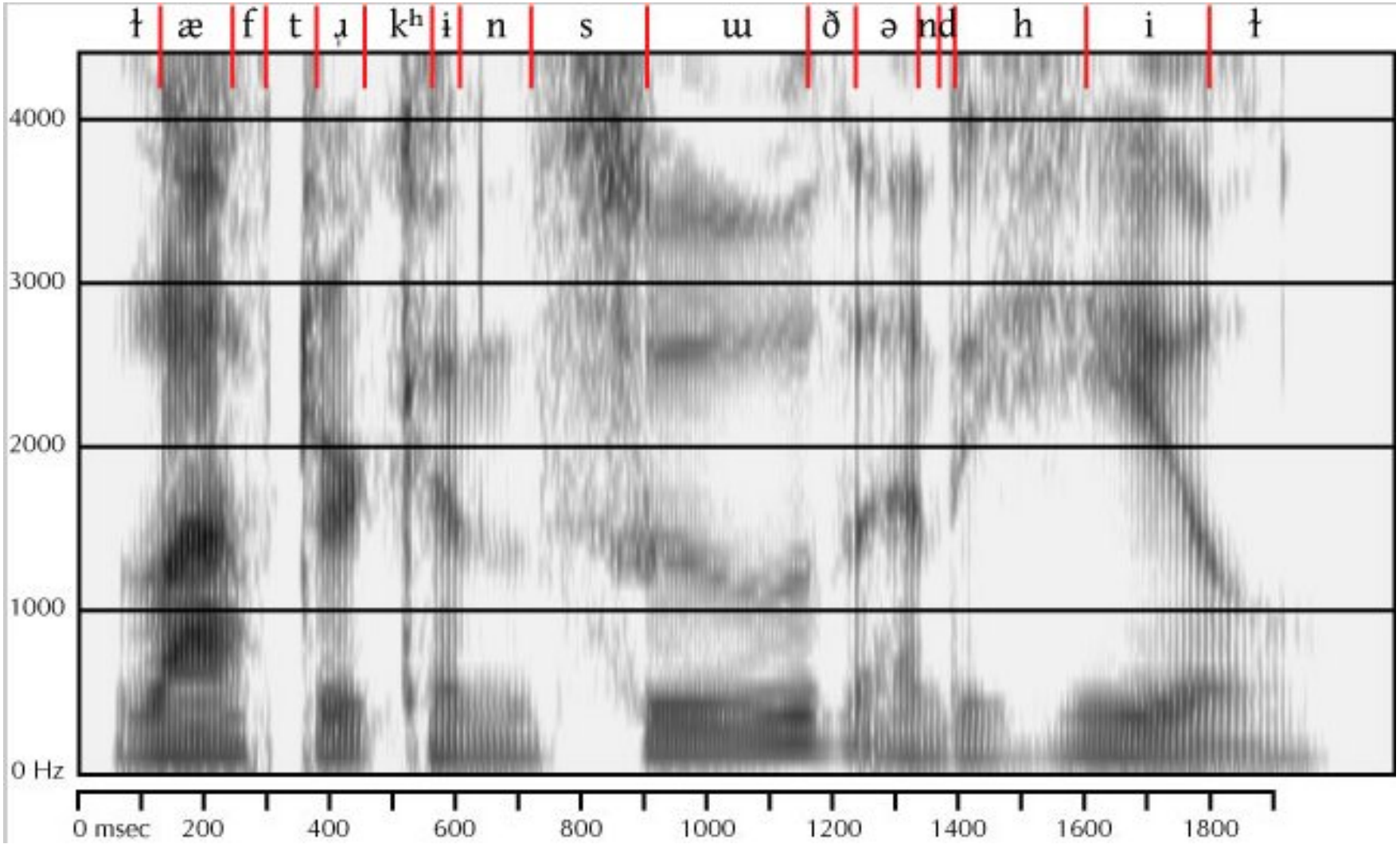
Remember support vector machines?

Hidden Markov Models

Léon Bottou

COS 424 – 4/13/2010

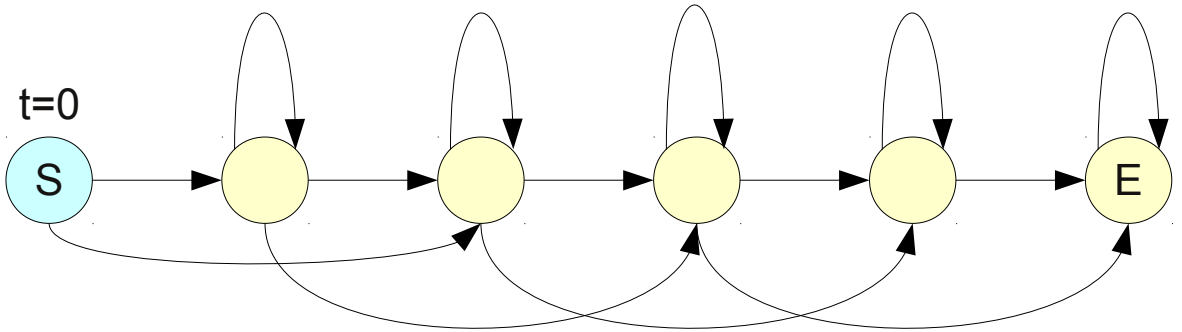
Spectrogram



"Laughter can soothe and heal."

Generative word model

Markov state machine



Transition probabilities

– Markov assumption: s_t depends only on s_{t-1}.

– Invariance assumption: $P_\theta(s_t \mid s_{t-1}) \stackrel{\Delta}{=} a_{s_t,s_{t-1}}$ does not depend on t.

Emission probabilities

– Independence assumption: x_t depends only s_t (and sometimes s_{t-1})

– Continuous HMM: $P_\theta(x_t \mid s_t = s)$ is $\mathcal{N}(\mu_s, \Sigma_s)$.

– Discrete HMM: $P_\theta(x_t \in \mathcal{X}_c \mid s_t = s) \stackrel{\Delta}{=} b_{cs}$ with $\mathcal{X}_c$ defined by clustering.

The Ferguson problems

Likelihood

– Given a specific HMM,
compute the likelihood of an observation sequence.

$\Longrightarrow$ Forward algorithm

Decoding

– Given an observation sequence and an HMM,
discover the most probable hidden state sequence.

$\Longrightarrow$ Viterbi algorithm

Learning

– Given an observation sequence, learn the HMM parameters.

$\Longrightarrow$ Forward-Backward algorithm

Factoring the likelihood (2bis)

Equivalent derivation:

$$\begin{aligned}
\alpha_t(s_t) &= \sum_{s_1 \dots s_{t-1}} \prod_{t'=1}^{t} a_{s_{t'-1}s_{t'}} P_\theta(x_{t'} \mid s_{t'}) \\
&= \sum_{s_{t-1}} P_\theta(x_t|s_t)\, a_{s_{t-1}s_t} \sum_{s_1 \dots s_{t-2}} \prod_{t'=1}^{t-1} a_{s_{t'-1}s_{t'}} P_\theta(x_{t'} \mid s_{t'}) \\
&= \sum_{s_{t-1}} \alpha_{t-1}(s_{t-1})\, a_{s_{t-1}s_t} P_\theta(x_t|s_t)
\end{aligned}$$

We have only used the arithmetic relations: $AB + AC = A(B + C)$

Decoding

Forward works because $AB + AC = A(B + C)$.

But we also have $\max(AB, AC) = A \max(B, C)$ when $A, B, C \geq 0$.

$$\alpha_t(i) \stackrel{\Delta}{=} \sum_{s_1 \ldots s_{t-1}} \prod_{t'=1}^{t} a_{s_{t'-1} s_{t'}} P_\theta(x_{t'} \,|\, s_{t'})$$

$$\alpha^\star_t(i) \stackrel{\Delta}{=} \max_{s_1 \ldots s_{t-1}} \prod_{t'=1}^{t} a_{s_{t'-1} s_{t'}} P_\theta(x_{t'} \,|\, s_{t'})$$

Viterbi algorithm

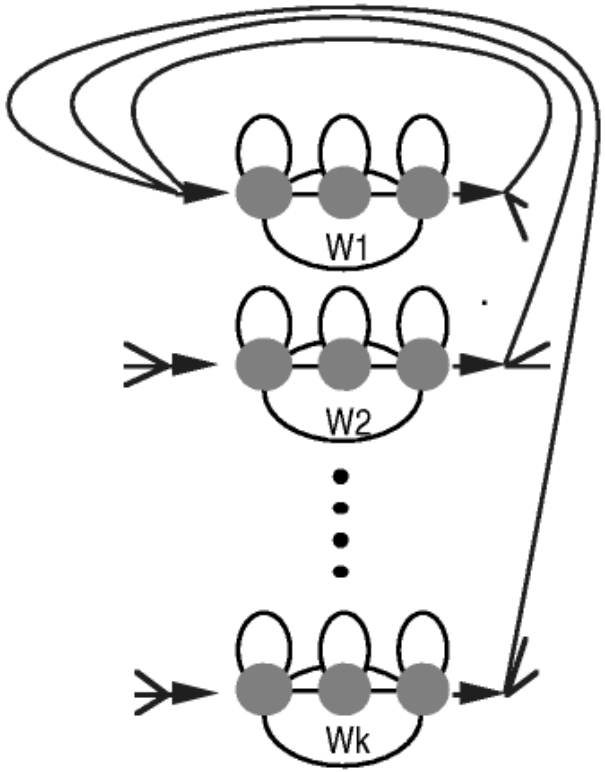
$$\alpha^\star_o(i) = \mathbb{1}\{i = \mathsf{Start}\}$$

$$\alpha^\star_t(i) = \max_j \alpha^\star_{t-1}(j)\, a_{ji}\, P_\theta(x_t | s_t = i)$$

$$\max_{s_1 \ldots s_T} P_\theta(s_1 \ldots s_T\,,\, x_1 \ldots x_T) = \max_{i \in \mathsf{End}} \alpha^\star_T(i)$$

Simultaneous recognition and segmentation

Construct a super model



Graphical Models

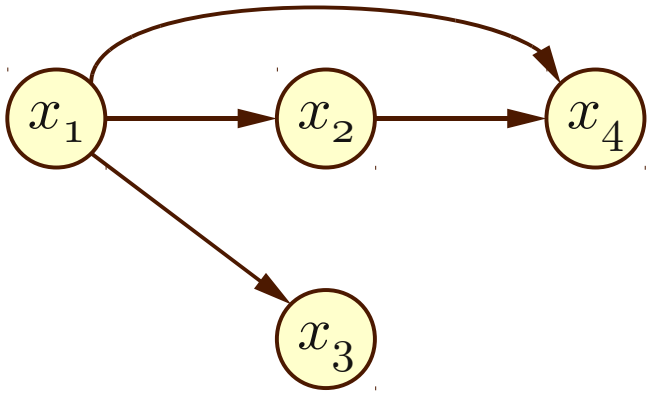
Léon Bottou

COS 424 – 4/15/2010

Graphical representation

Independence assumptions

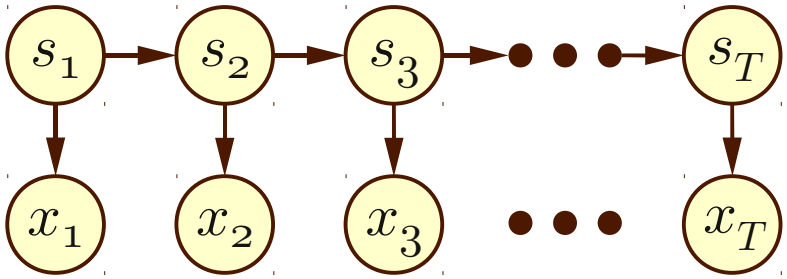
$$\begin{aligned} P(x_1, x_2, x_3, x_4) &= P(x_1)\, P(x_2|x_1)\, P(x_3|x_1, x_2)\, P(x_4|x_1, x_2, x_3) \\ &= P(x_1)\, P(x_2|x_1)\, P(x_3|x_1)\, P(x_4|x_1, x_2) \end{aligned}$$



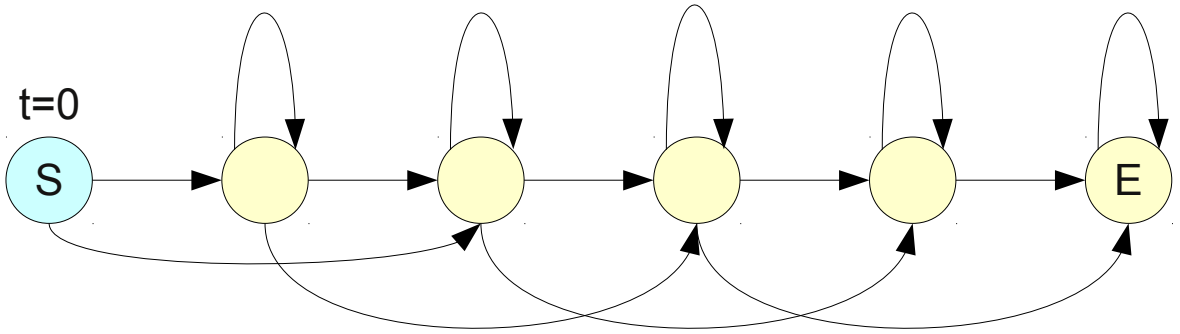
Missing links represent independence assumptions

Hidden Markov Models

$$P(x_1 \dots x_T,\, s_1 \dots s_T) = P(s_1)\, P(x_1|s_1)\, P(s_2|s_1)\, P(x_2|s_2)\, \dots P(s_T|s_{T-1})\, P(x_T|s_T)$$



What is the relation between this graph and that graph?



D-separation

Problem

– Consider three disjoint sets of nodes: A, B, C.

– When do we have $A \perp\!\!\!\perp B \mid C$?

Definition

A and B are *d-separated* by C if all paths from $a \in A$ to $b \in B$

– contain a head-to-tail or tail-to-tail node $c \in C$, or

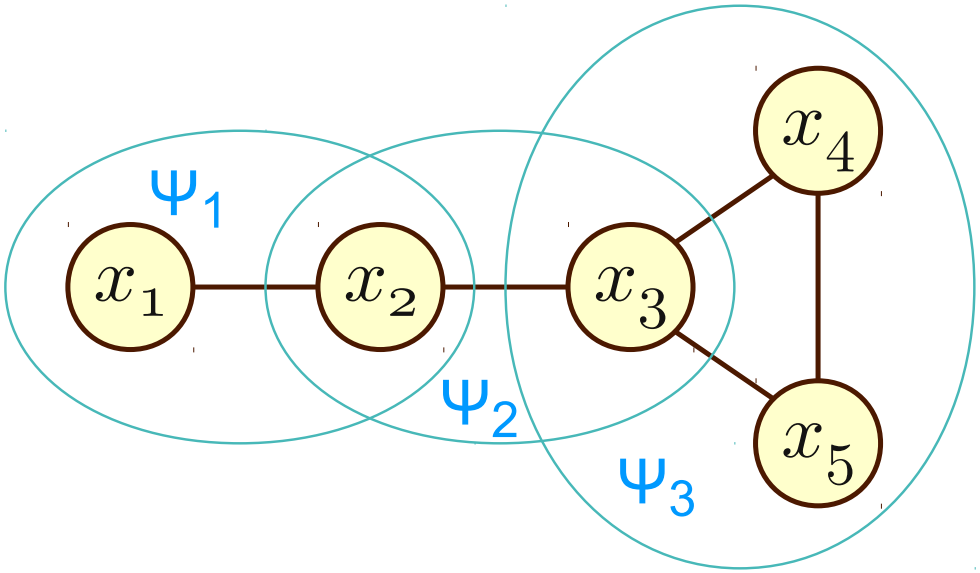
– contain a head-to-head node c such that neither c nor any of its descendants belongs to C.

Theorem

A and B are *d-separated* by $C \quad \Longleftrightarrow \quad A \perp\!\!\!\perp B \mid C$

Graphical representation

$$P(x_1, x_2, x_3, x_4, x_5) = \frac{1}{Z} \Psi_1(x_1, x_2) \Psi_2(x_2, x_3) \Psi_3(x_3, x_4, x_5)$$



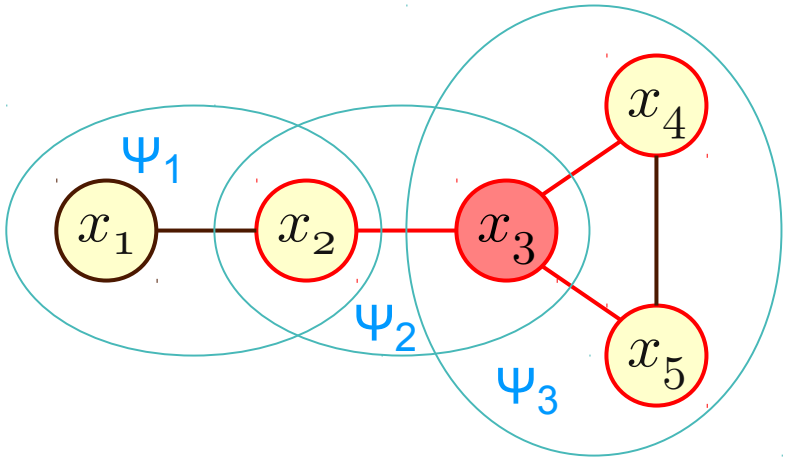
– Completely connect the nodes belonging to each $\mathbf{x}_C$.

– Each subset $\mathbf{x}_C$ forms a *clique* of the graph.

Graph and Markov blanket

The Markov blanket of a MRF variable is the set of its neighbors.

$$P(x_3 \,|\, x_1, x_2, x_4, x_5) = P(x_3 \,|\, x_2, x_4, x_5)$$



Consequence

– Consider three disjoint sets of nodes: A, B, C.

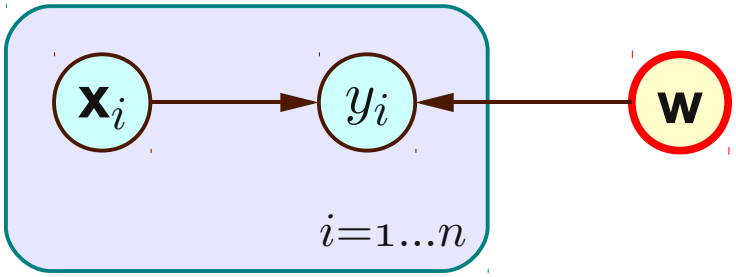
$$A \perp\!\!\!\perp B \mid C \quad \Longleftrightarrow \quad \begin{cases} \text{Any path between } a \in A \text{ and } b \in B \\ \text{passes through a node } c \in C. \end{cases}$$

Conversely (Hammersley-Clifford theorem)

– Any distribution that satisfies such properties with respect to an undirected graph is a Markov Random Field.

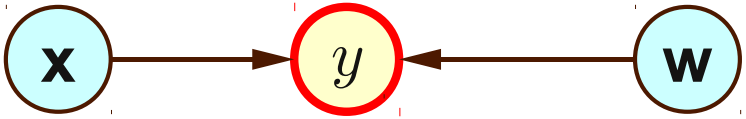
Inference

Inference for learning



$P(\mathbf{w} \mid \mathcal{D})$?

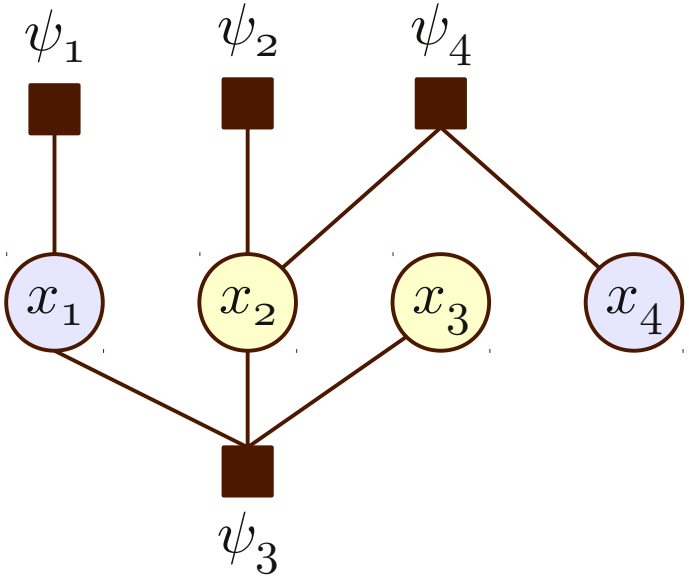
Inference for recognition



$P(\mathrm{y} \mid \mathbf{x}, \mathbf{w})$?

Gibbs sampling

A computationally intensive inference algorithm



Clamp the observed variables.

Randomly initialize the other variables.

Repeat:

– Pick one unobserved variable x.

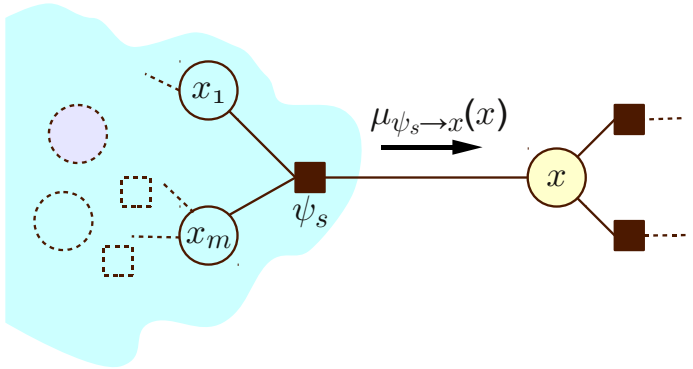
– Compute $P(\, x \,|\, \mathsf{ne}(\mathsf{ne}(x))\,)$.

– Pick a new value for x accordingly.

Observe the empirical distribution of the variables of interest.

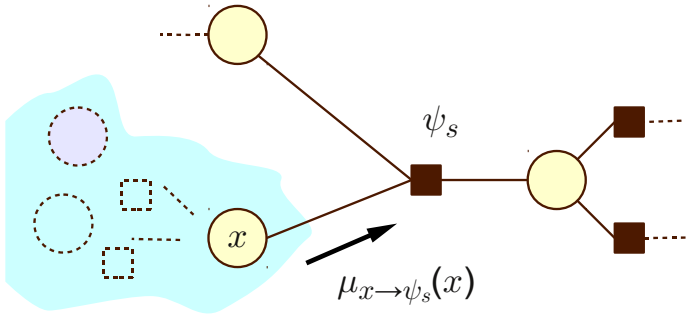
Sum-product algorithm (2)

Recursions



$$\mu_{\Psi_s \to x}(x) = \sum_{x_1..x_m..x_M} \Psi_s(\mathbf{x}_s) \prod_m \mu_{x_m \to \Psi_s}(x_m)$$

$$\mu_{\Psi_s \to x}(x) = \Psi_s(x) \quad \text{if } \Psi_s \text{ is a leaf.}$$



$$\mu_{x \to \Psi_s}(x) = \prod_{l \in \mathsf{ne}(x) \setminus s} \mu_{\Psi_l \to x}(x)$$

$$\mu_{x \to \Psi_s}(x) = 1 \text{ if } x \text{ is a leaf.}$$

– These recursion work because we assume the factor graph is a tree.
– Starting from the leafs, compute the messages μ everywhere.

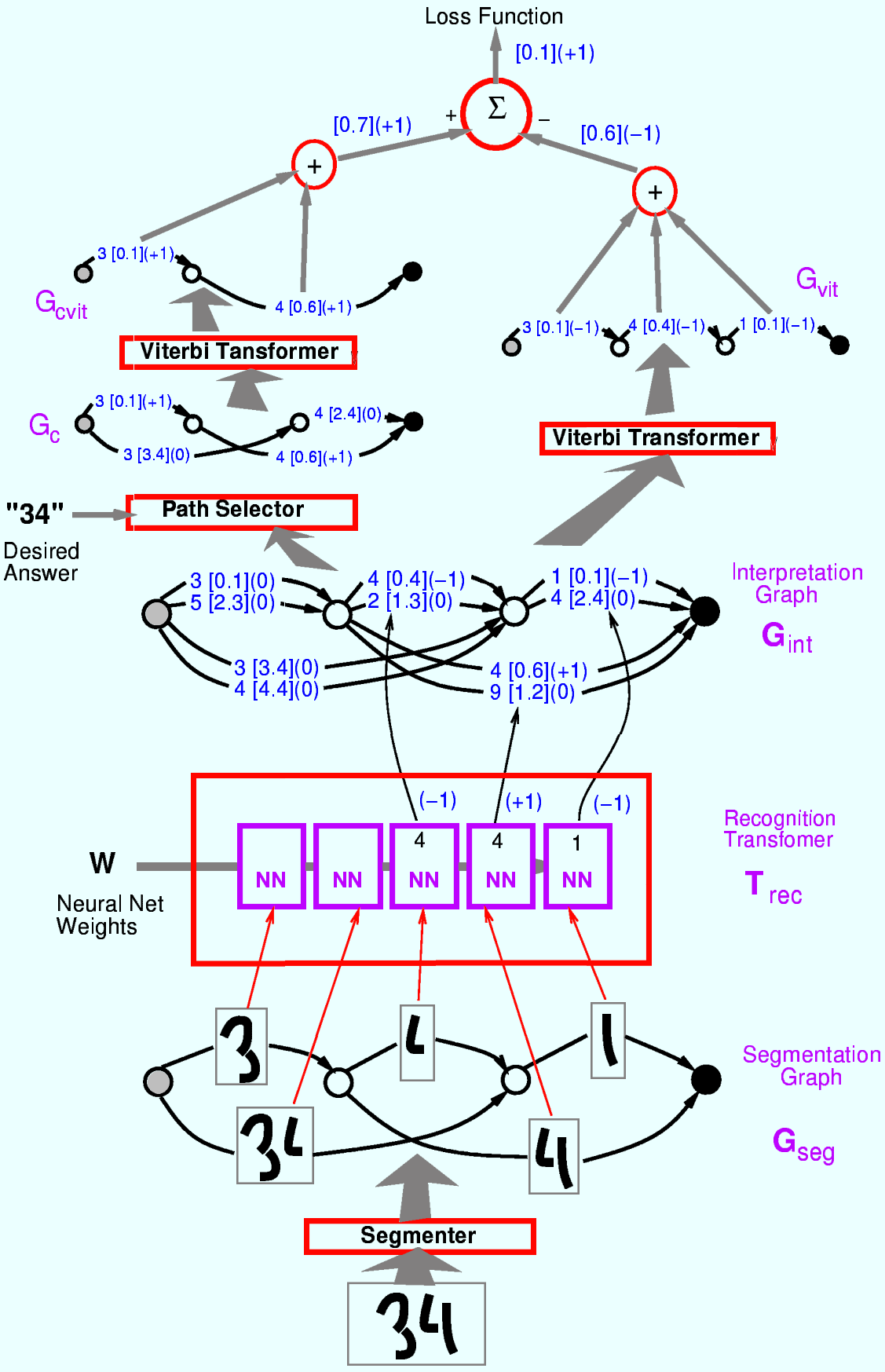
Graph Transformer Networks

Léon Bottou

Joint work with
Yann Le Cun
Yoshua Bengio
Patrick Haffner

AT&T Labs – Research

Middletown, NJ

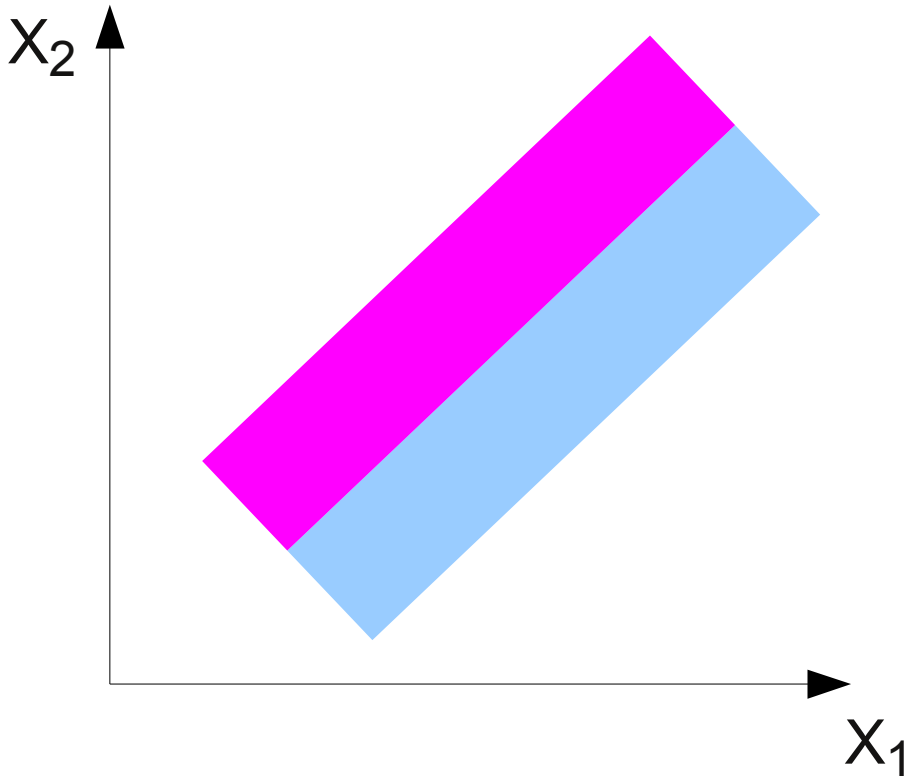

Loss Function
[0.1](+1)
Σ
+
–
[0.7](+1)
[0.6](–1)
+
+
3 [0.1](+1)
4 [0.6](+1)
G_{cvit}
Viterbi Tansformer
3 [0.1](–1)
4 [0.4](–1)
1 [0.1](–1)
G_{vit}
3 [0.1](+1)
4 [2.4](0)
3 [3.4](0)
4 [0.6](+1)
G_c
Viterbi Transformer
"34"
Desired Answer
Path Selector
3 [0.1](0)
5 [2.3](0)
4 [0.4](–1)
2 [1.3](0)
1 [0.1](–1)
4 [2.4](0)
3 [3.4](0)
4 [4.4](0)
4 [0.6](+1)
9 [1.2](0)
Interpretation Graph
G_{int}
(–1)
(+1)
(–1)
W
Neural Net Weights
NN
NN
4
NN
4
NN
1
NN
Recognition Transfomer
T_{rec}
Segmentation Graph
G_{seg}
Segmenter
34

Feature engineering

Léon Bottou

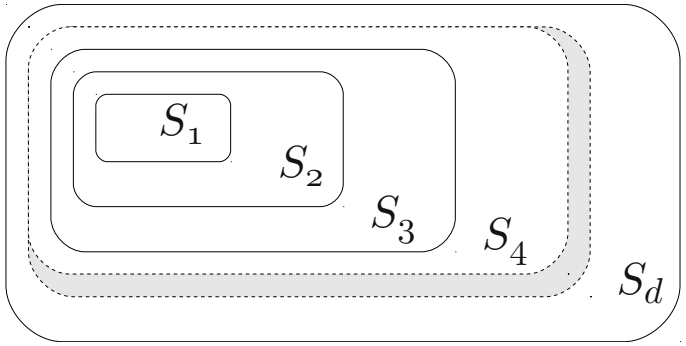
COS 424 – 4/22/2010

Interesting example



Correlated variables may be useless by themselves.

L_0 structural risk minimization



S_r : classifiers with at most r features.

Let $E_r = \min_{f \in S_r} E_{\text{test}}(f)$. The following result holds (Ng 1998):

$$E_{\text{test}}(f^\star) \leq \min_{r=1\ldots d} \left\{ E_r + \tilde{\mathcal{O}}\left(\sqrt{\frac{h_r}{n_{\text{train}}}}\right) + \tilde{\mathcal{O}}\left(\sqrt{\frac{r \log d}{n_{\text{train}}}}\right) \right\} + \mathcal{O}\left(\sqrt{\frac{\log d}{n_{\text{valid}}}}\right)$$

Assume E_r is quite good for a low number of features r.

Meaning that few features are relevant.

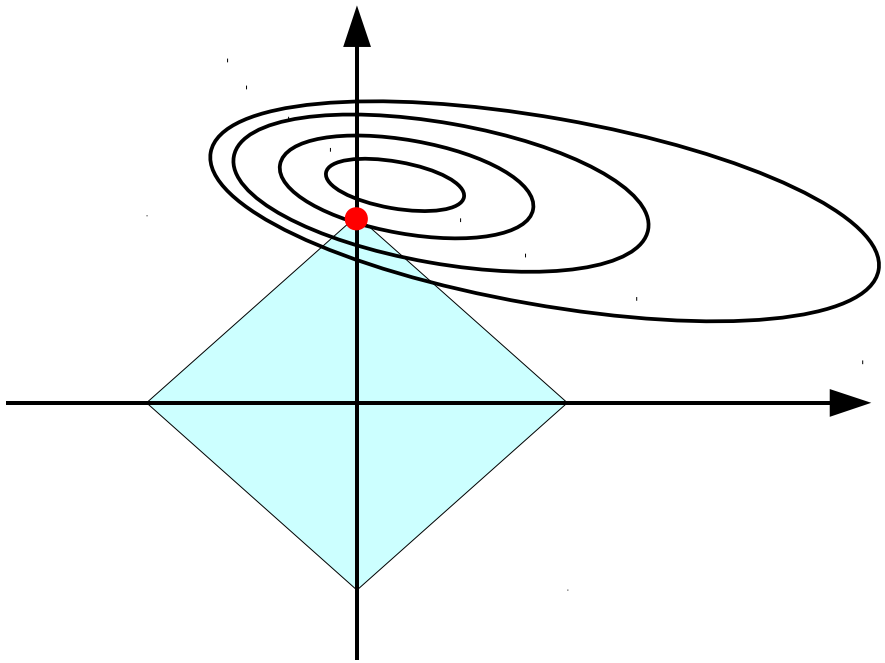
Then we can still find a good classifier if h_r and $\log d$ are reasonable.

We can filter an exponential number of irrelevant features.

L_1 regularisation

The L_1 norm is the first convex L_p norm.

$$\min_{\mathbf{w}} \quad \frac{1}{n}\sum_{i=1}^{n} \ell(y, f_{\mathbf{w}}(x)) \;+\; \lambda|\mathbf{w}|_1$$



Same logarithmic property (Tsybakov 2006).

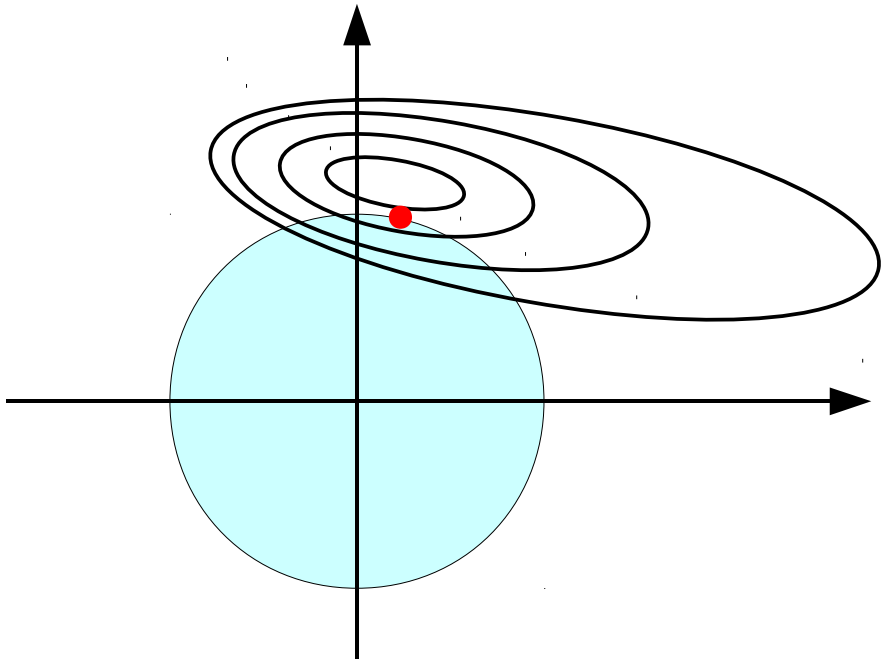
L_1 regulatization can weed an exponential number of irrelevant features.

See also "*compressed sensing*".

L_2 regularisation

The L_2 norm is the same as the maximum margin idea.

$$\min_{\mathbf{w}} \quad \frac{1}{n}\sum_{i=1}^{n} \ell(y, f_{\mathbf{w}}(x)) \;+\; \lambda\|\mathbf{w}\|_2$$



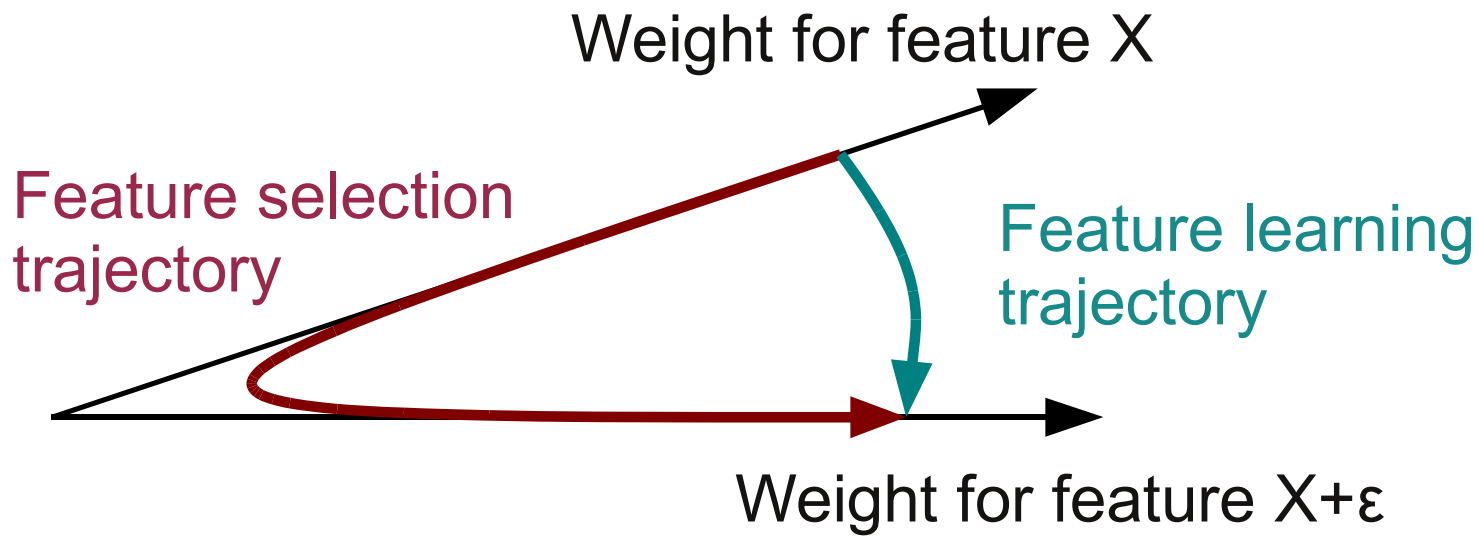
Logarithmic property is lost.

Rotationally invariant regularizer!

SVMs do not have magic properties for filtering out irrelevant features. They perform best when dealing with lots of relevant features.

Feature learning in one slide

Suppose we have weight on a feature X.
Suppose we prefer a closely related feature $X + \epsilon$.

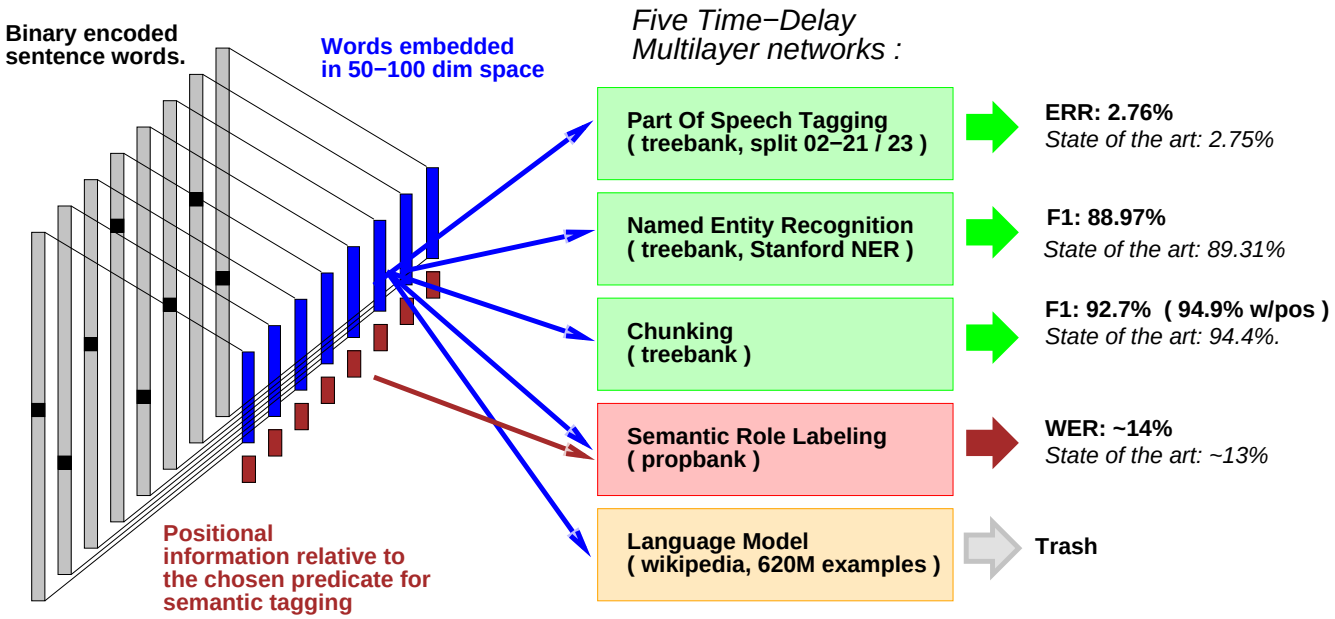


Natural language processing and weak supervision

Léon Bottou

COS 424 – 4/27/2010

SGD in Real Life: Sentence Analysis



– (Collobert and Weston, 2007, 2008) [not my work]
– No hand-tuned parsing tricks.
– No hand-tuned linguistic features.
– State-of-the-art accuracies.
– Analyzes a sentence in 50 milliseconds *(instead of seconds.)*
– Trains with SGD in about 3 weeks.