A tutorial on active learning

Sanjoy Dasgupta[1] John Langford[2]

UC San Diego[1]

Yahoo Labs[2]

Exploiting unlabeled data

A lot of unlabeled data is plentiful and cheap, eg.

- documents off the web
- speech samples
- images and video

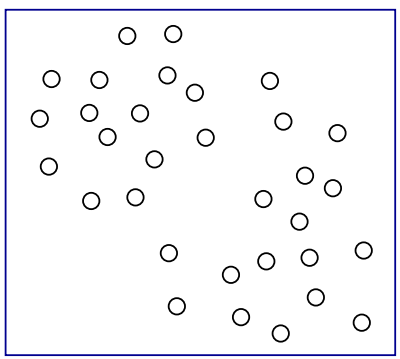
But labeling can be expensive.

Exploiting unlabeled data

A lot of unlabeled data is plentiful and cheap, eg.

- documents off the web
- speech samples
- images and video

But labeling can be expensive.



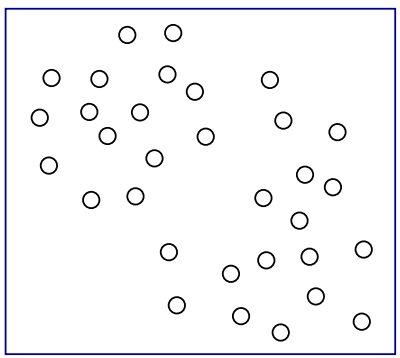
Unlabeled points

Exploiting unlabeled data

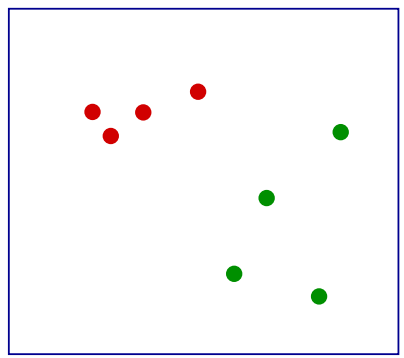
A lot of unlabeled data is plentiful and cheap, eg.

- documents off the web
- speech samples
- images and video

But labeling can be expensive.



Unlabeled points



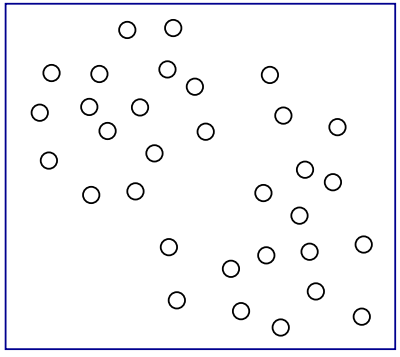
Supervised learning

Exploiting unlabeled data

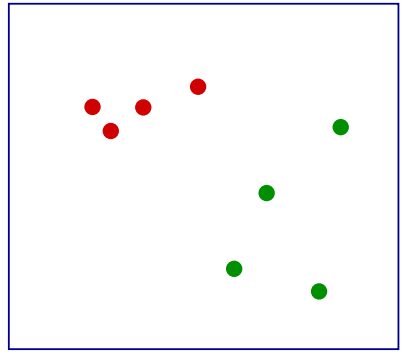
A lot of unlabeled data is plentiful and cheap, eg.

- documents off the web
- speech samples
- images and video

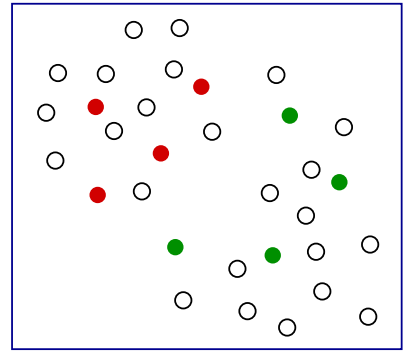
But labeling can be expensive.



Unlabeled points



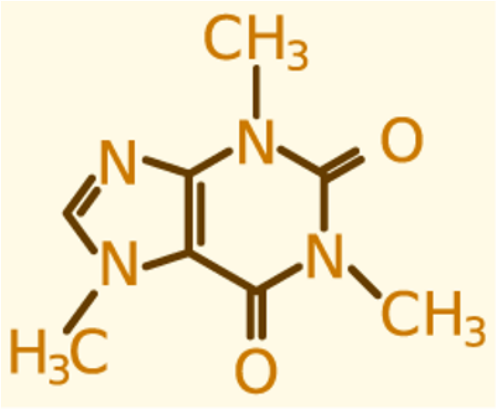
Supervised learning



Semisupervised and active learning

Active learning example: drug design [Warmuth et al 03]

Goal: find compounds which bind to a particular target

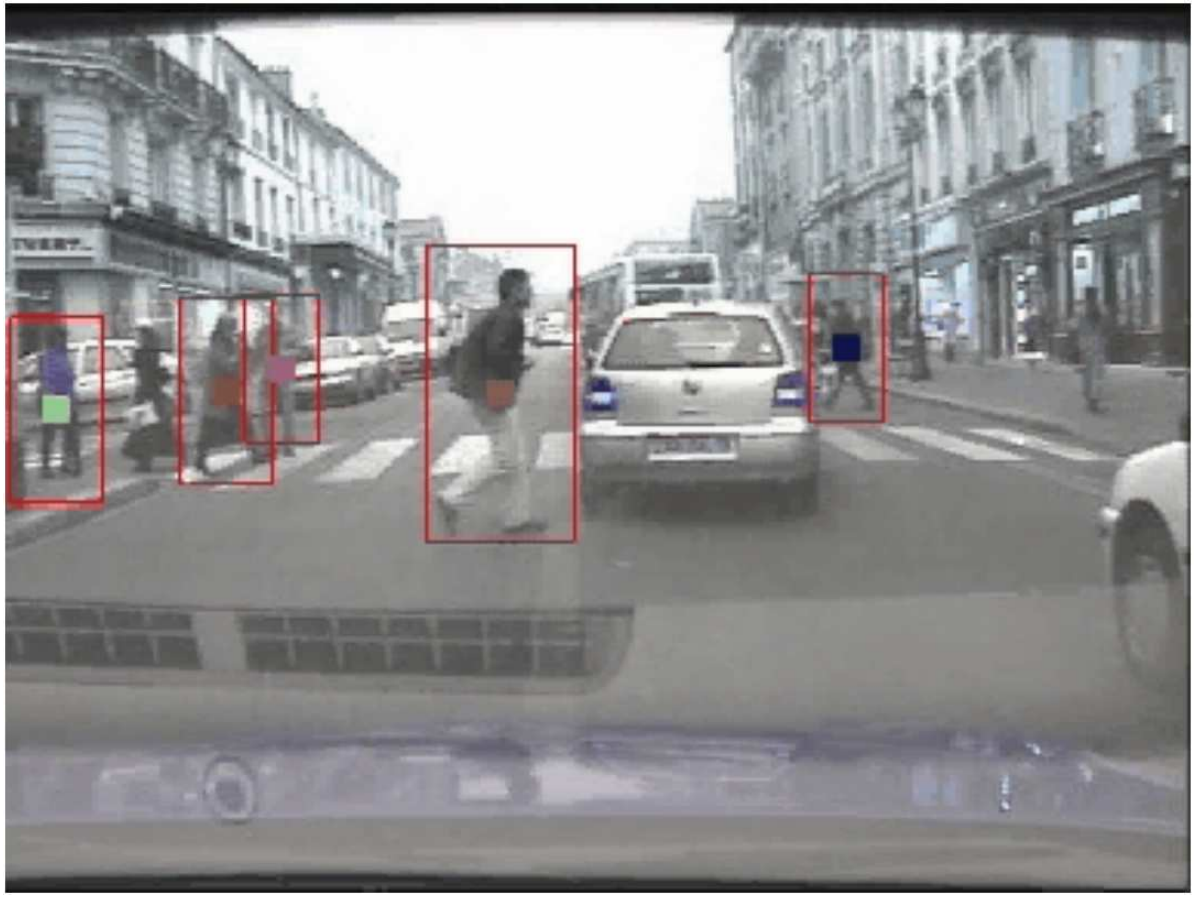


Large collection of compounds, from:

- vendor catalogs
- corporate collections
- combinatorial chemistry

unlabeled point	$\equiv$	description of chemical compound
label	$\equiv$	*active* (binds to target) vs. *inactive*
getting a label	$\equiv$	chemistry experiment

Active learning example: pedestrian detection [Freund et al 03]



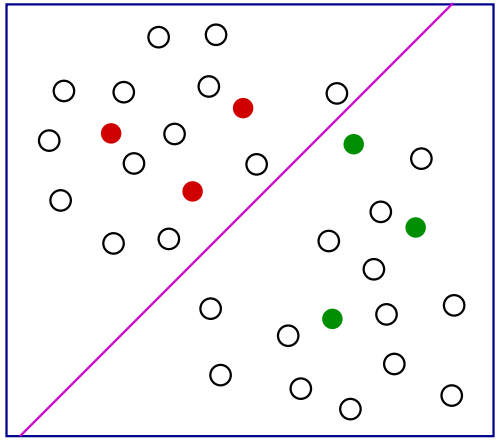
Typical heuristics for active learning

Start with a pool of unlabeled data

Pick a few points at random and get their labels

Repeat

> Fit a classifier to the labels seen so far
> Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)



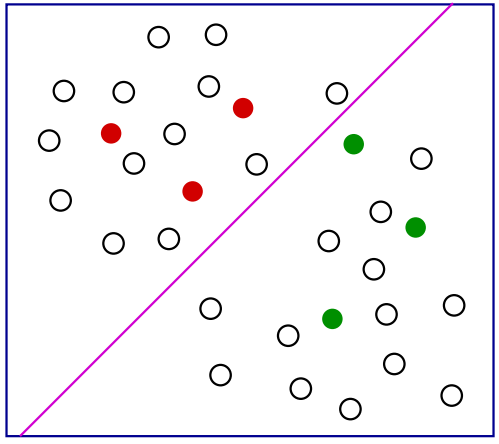
Typical heuristics for active learning

Start with a pool of unlabeled data

Pick a few points at random and get their labels

Repeat

> Fit a classifier to the labels seen so far
> Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)



Biased sampling: the labeled points are not representative of the underlying distribution!

Sampling bias

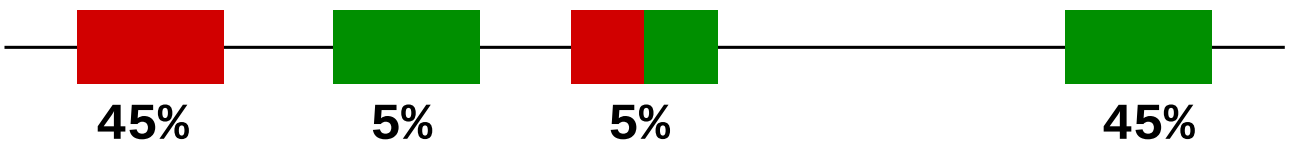
Start with a pool of unlabeled data

Pick a few points at random and get their labels

Repeat

Fit a classifier to the labels seen so far
Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)

Example:



Sampling bias

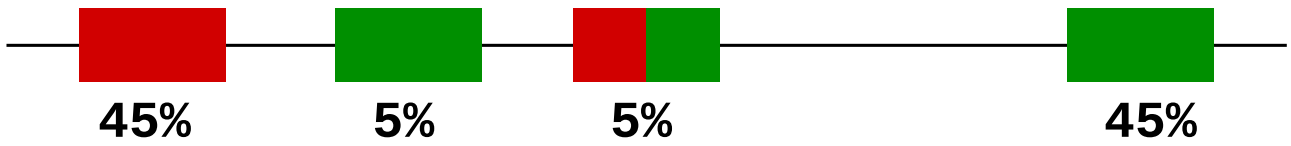
Start with a pool of unlabeled data

Pick a few points at random and get their labels

Repeat

Fit a classifier to the labels seen so far
Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)

Example:



Even with infinitely many labels, converges to a classifier with 5% error instead of the best achievable, 2.5%. *Not consistent!*

Sampling bias

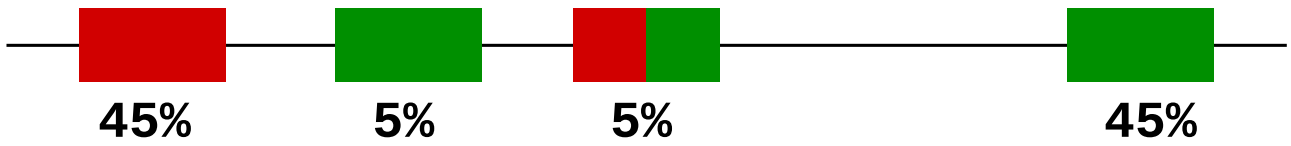
Start with a pool of unlabeled data

Pick a few points at random and get their labels

Repeat

Fit a classifier to the labels seen so far
Query the unlabeled point that is closest to the boundary (or most uncertain, or most likely to decrease overall uncertainty,...)

Example:



Even with infinitely many labels, converges to a classifier with 5% error instead of the best achievable, 2.5%. *Not consistent!*

Manifestation in practice, eg. Schutze et al 03.

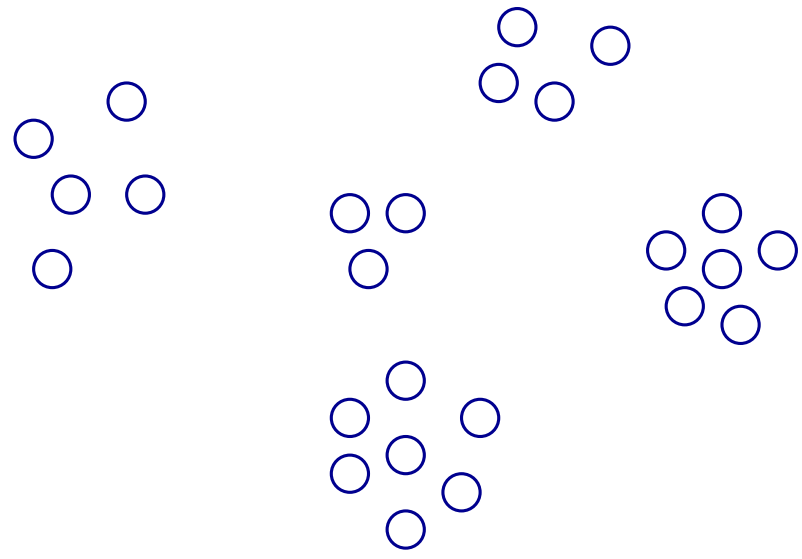
Can adaptive querying really help?

There are two distinct narratives for explaining how adaptive querying can help.

- Case I: Exploiting (cluster) structure in data
- Case II: Efficient search through hypothesis space

Case I: Exploiting cluster structure in data

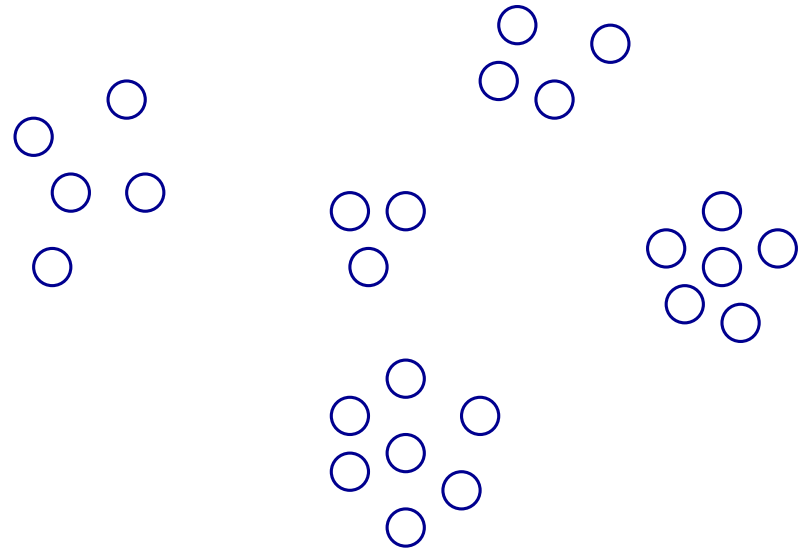
Suppose the unlabeled data looks like this.



Then perhaps we just need five labels!

Case I: Exploiting cluster structure in data

Suppose the unlabeled data looks like this.

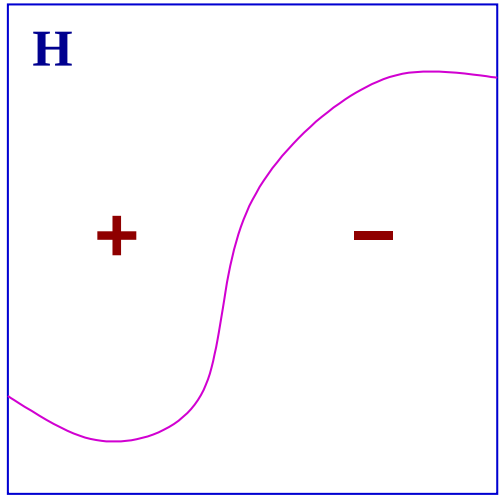


Then perhaps we just need five labels!

Challenges: In general, the cluster structure (i) is not so clearly defined and (ii) exists at many levels of granularity. And the clusters themselves might not be pure in their labels. How to exploit whatever structure happens to exist?

Case II: Efficient search through hypothesis space

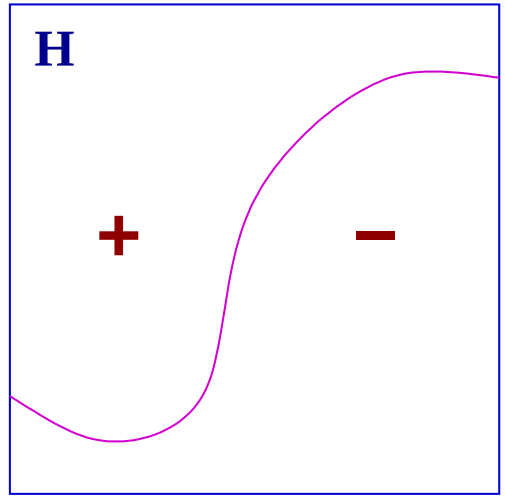
Ideal case: each query cuts the version space in two.



Then perhaps we need just $\log |H|$ labels to get a perfect hypothesis!

Case II: Efficient search through hypothesis space

Ideal case: each query cuts the version space in two.



Then perhaps we need just $\log |H|$ labels to get a perfect hypothesis!

Challenges: (1) Do there always exist queries that will cut off a good portion of the version space? (2) If so, how can these queries be found? (3) What happens in the nonseparable case?

Outline of tutorial

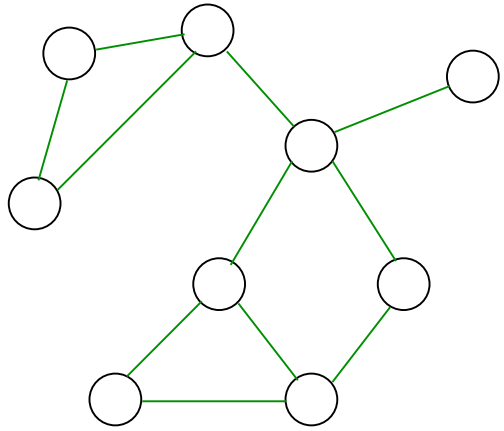
I. Exploiting (cluster) structure in data

II. Efficient search through hypothesis space

Sine qua non: *statistical consistency*.

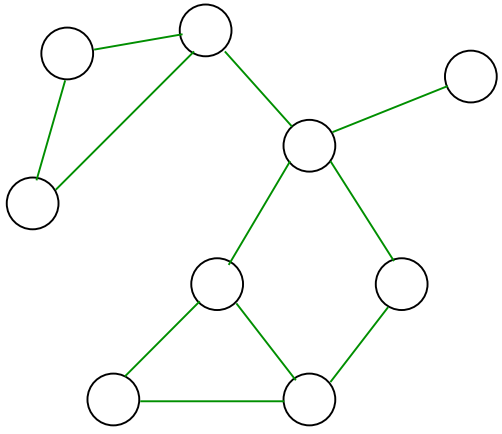
A cluster-based active learning scheme [ZGL 03]

(1) Build neighborhood graph

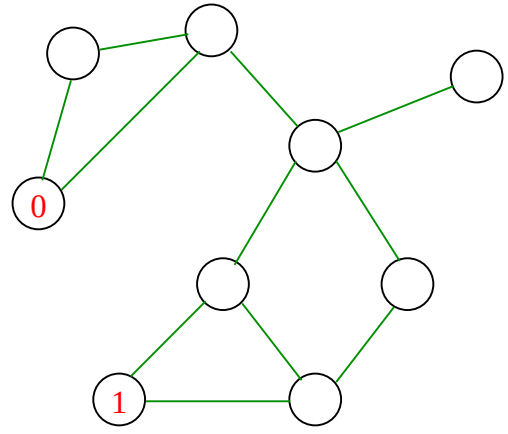


A cluster-based active learning scheme [ZGL 03]

(1) Build neighborhood graph

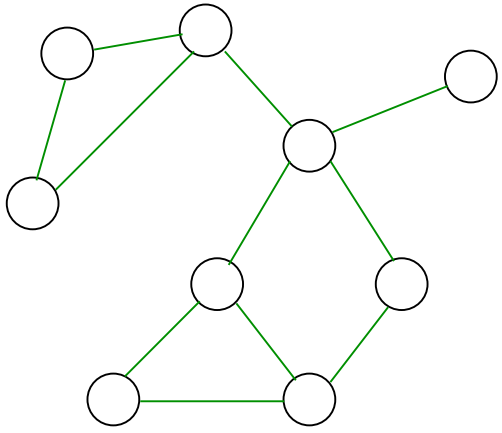


(2) Query some random points

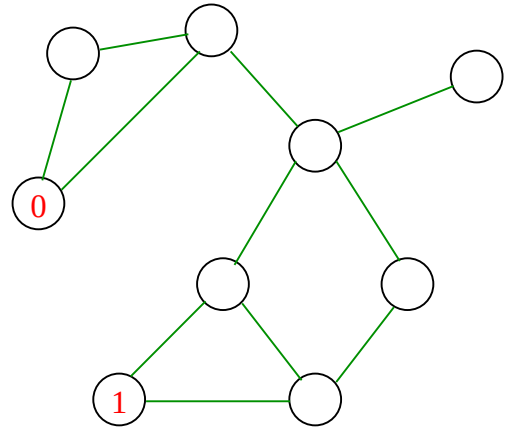


A cluster-based active learning scheme [ZGL 03]

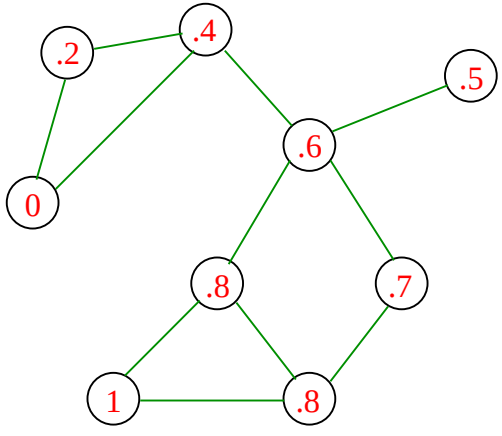
(1) Build neighborhood graph



(2) Query some random points

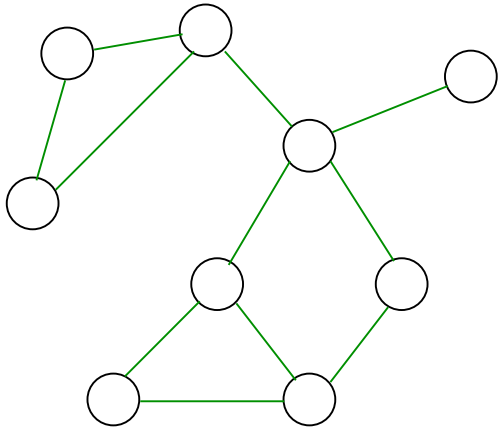


(3) Propagate labels

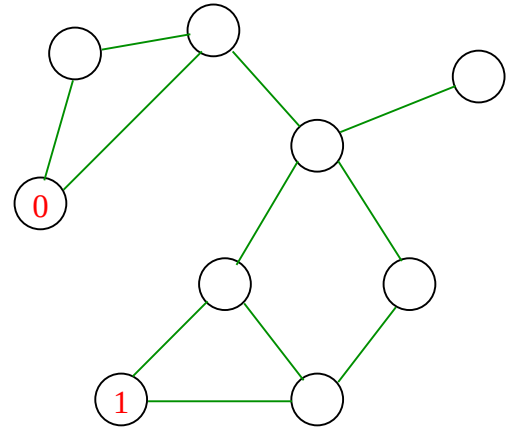


A cluster-based active learning scheme [ZGL 03]

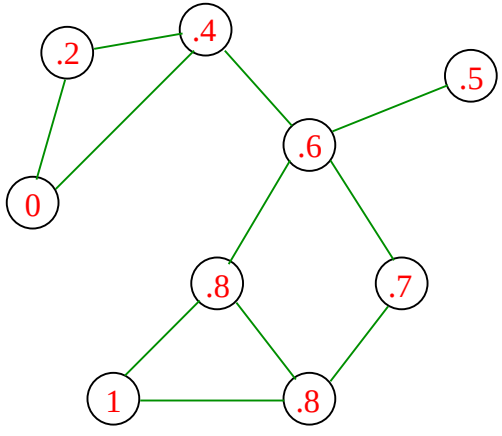
(1) Build neighborhood graph



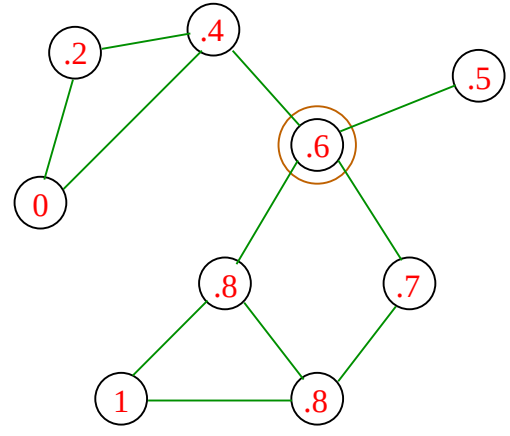
(2) Query some random points



(3) Propagate labels

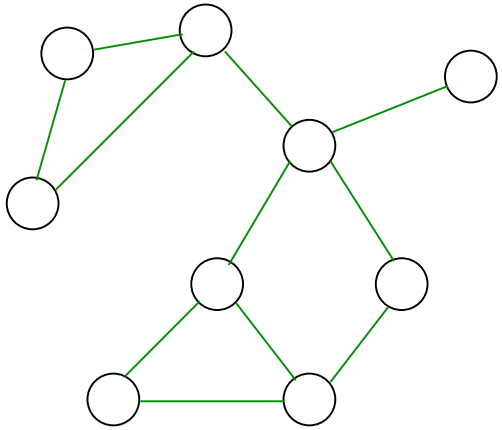


(4) Make query and go to (3)

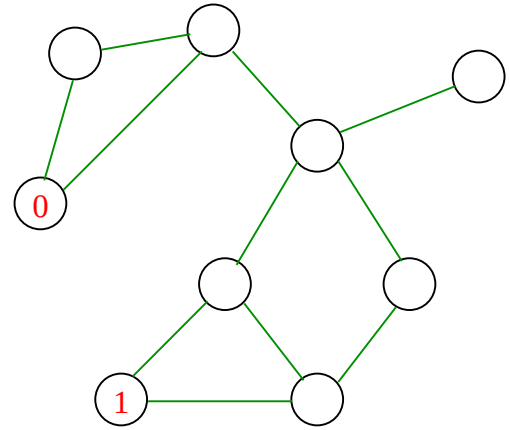


A cluster-based active learning scheme [ZGL 03]

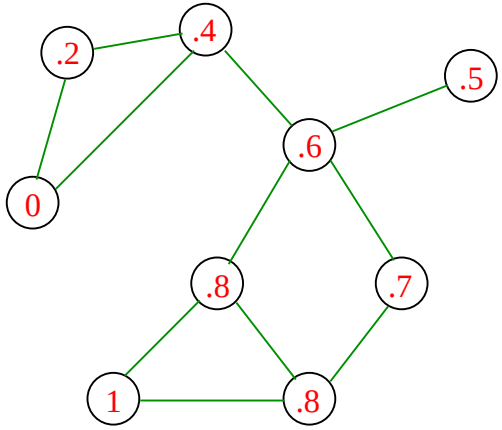
(1) Build neighborhood graph



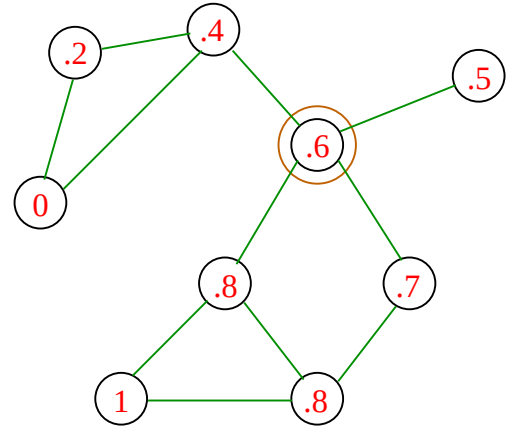
(2) Query some random points



(3) Propagate labels



(4) Make query and go to (3)

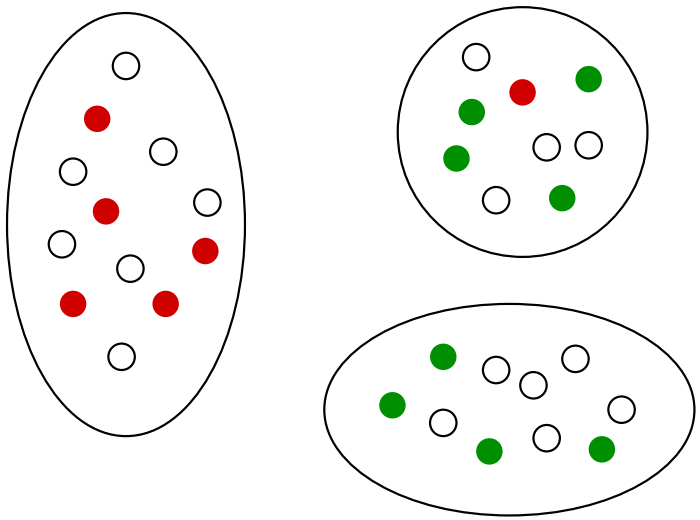


Clusters in data $\Rightarrow$ graph cut which curtails propagation of influence

Exploiting cluster structure in data [DH 08]

Basic primitive:

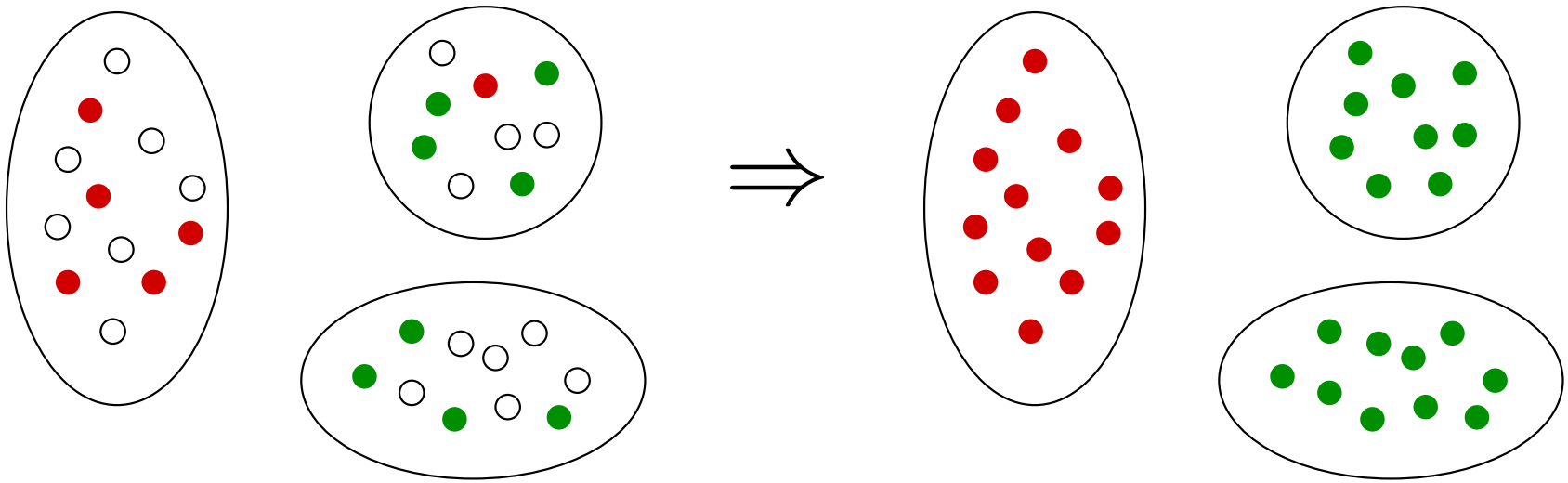
- Find a clustering of the data
- Sample a few *randomly-chosen* points in each cluster
- Assign each cluster its majority label
- Now use this fully labeled data set to build a classifier



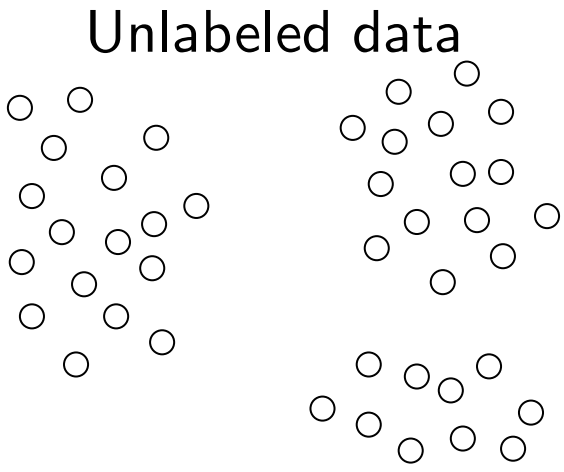
Exploiting cluster structure in data [DH 08]

Basic primitive:

- Find a clustering of the data
- Sample a few *randomly-chosen* points in each cluster
- Assign each cluster its majority label
- Now use this fully labeled data set to build a classifier

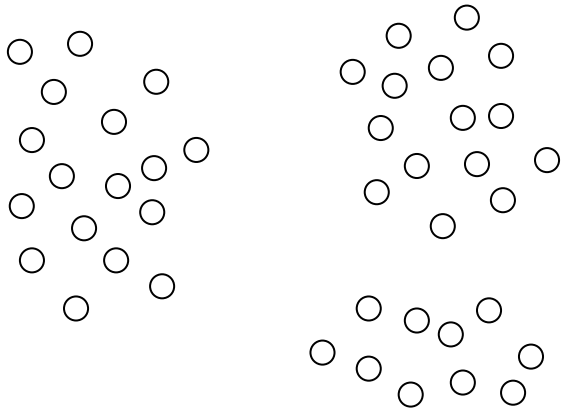


Finding the right granularity

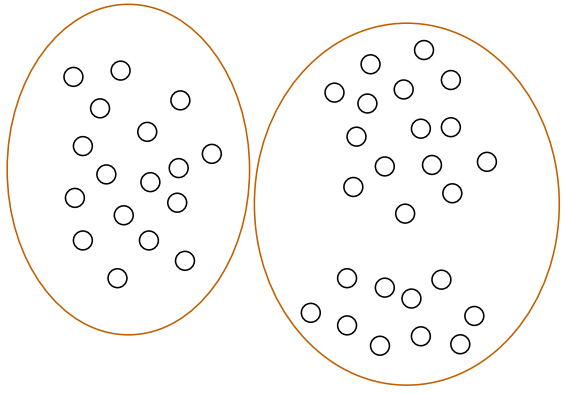


Finding the right granularity

Unlabeled data

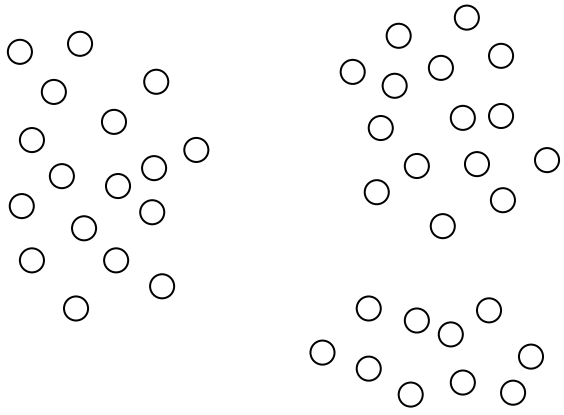


Find a clustering

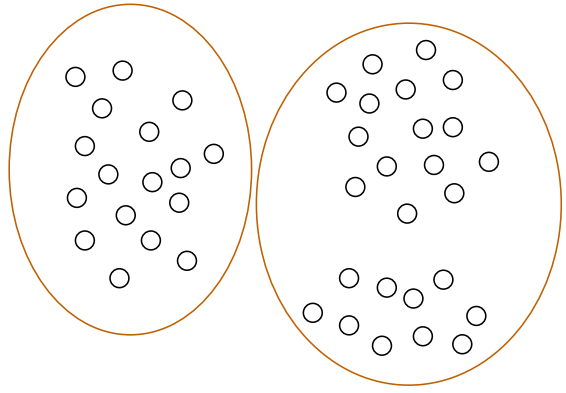


Finding the right granularity

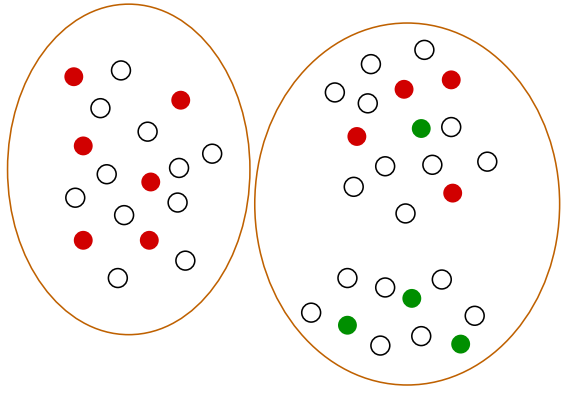
Unlabeled data



Find a clustering

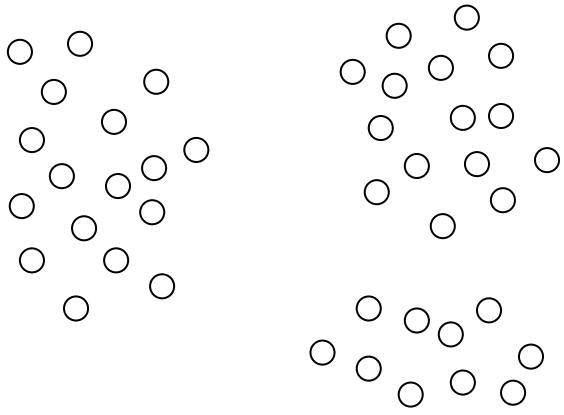


Ask for some labels

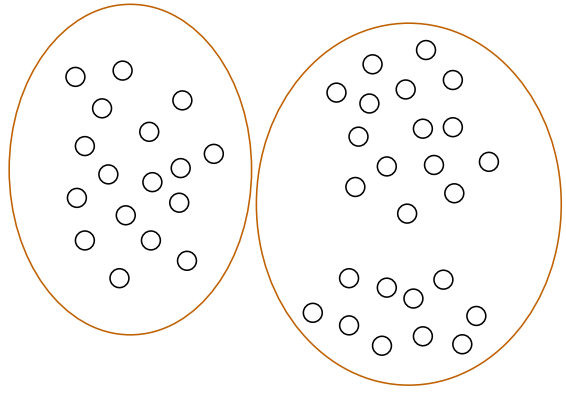


Finding the right granularity

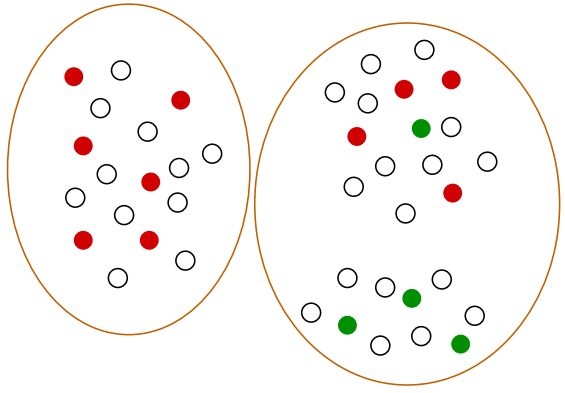
Unlabeled data



Find a clustering



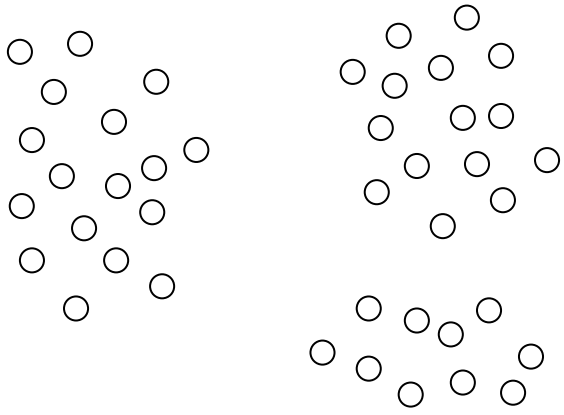
Ask for some labels



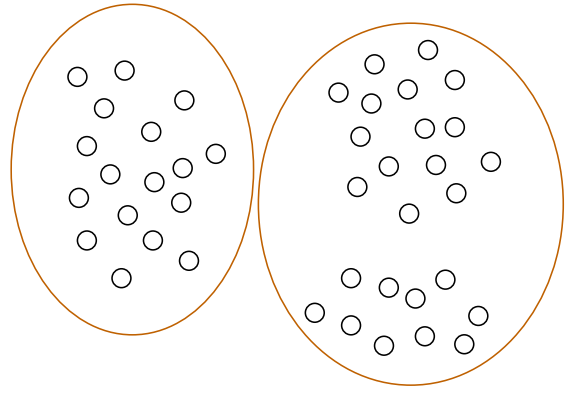
Now what?

Finding the right granularity

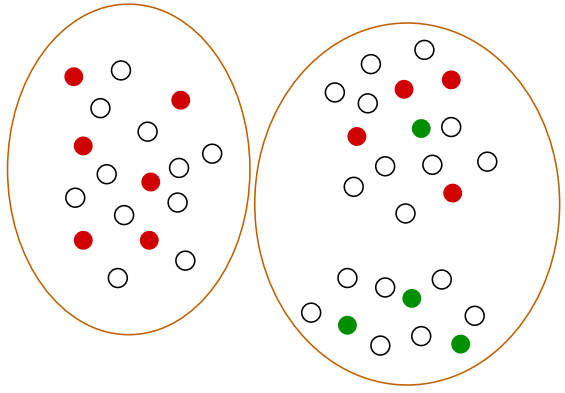
Unlabeled data



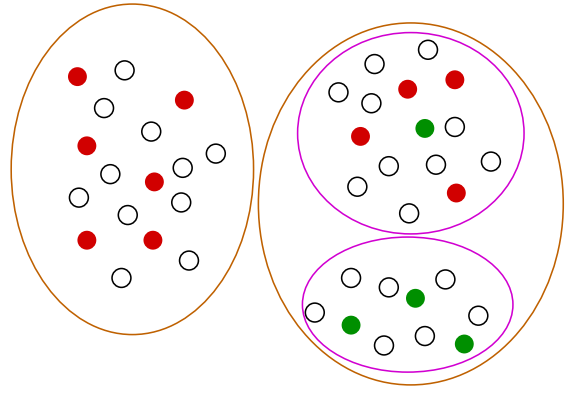
Find a clustering



Ask for some labels

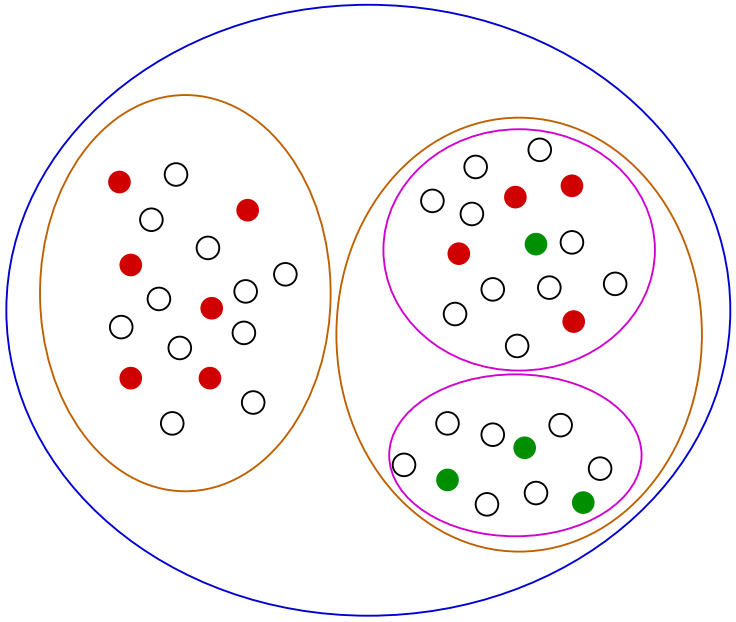


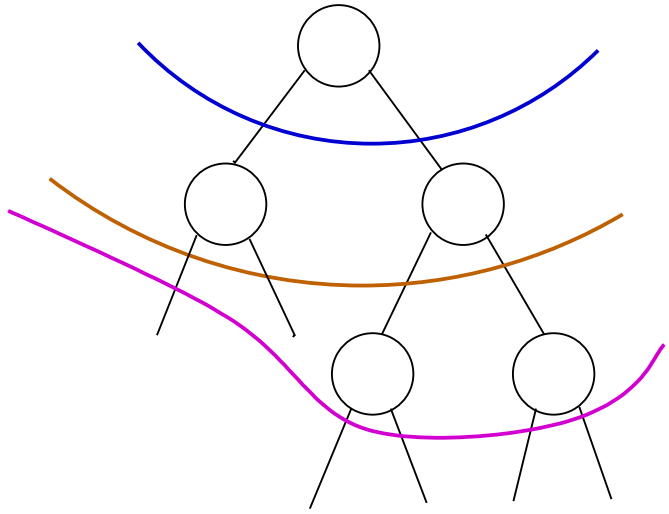
Refine the clustering



Now what?

Using a hierarchical clustering





Rules:

- Always work with some pruning of the hierarchy: a clustering induced by the tree. Pick a cluster (intelligently) and query a *random* point in it.
- For each tree node (i.e. cluster) v maintain: (i) majority label $L(v)$; (ii) empirical label frequencies $\widehat{p}_{v,l}$; and (iii) confidence interval $[p^{lb}_{v,l}, p^{ub}_{v,l}]$

Algorithm: hierarchical sampling

Input: Hierarchical clustering T

For each node v maintain: (i) majority label $L(v)$; (ii) empirical label frequencies $\widehat{p}_{v,l}$; and (iii) confidence interval $[p^{lb}_{v,l}, p^{ub}_{v,l}]$

Initialize: pruning $P = \{\text{root}\}$, labeling $L(\text{root}) = \ell_0$

for t = 1, 2, 3, . . . :

- $v = \text{select-node}(P)$
- pick a random point z in subtree T_v and query its label
- update empirical counts for all nodes along path from z to v
- choose best pruning and labeling (P', L') of T_v
- $P = (P \setminus \{v\}) \cup P'$ and $L(u) = L'(u)$ for all u in P'

for each v in P: assign each leaf in T_v the label $L(v)$

return the resulting fully labeled data set

$$v = \text{select-node}(P) \equiv \begin{cases} \text{Prob}[v] \propto |T_v| & \textit{random sampling} \\ \text{Prob}[v] \propto |T_v|(1 - p^{lb}_{v,l}) & \textit{active sampling} \end{cases}$$

Outline of tutorial

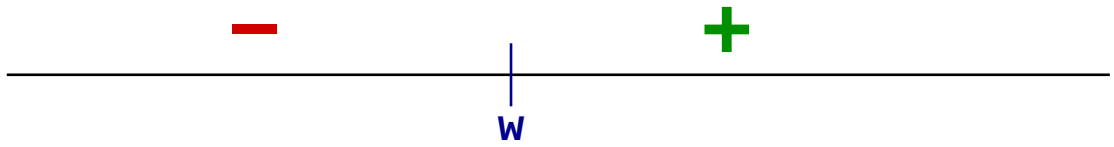
I. Exploiting (cluster) structure in data
II. Efficient search through hypothesis space
    (a) The separable case
    (b) The general case

Efficient search through hypothesis space

Threshold functions on the real line:

$H = \{h_w : w \in \mathbb{R}\}$

$h_w(x) = 1(x \geq w)$



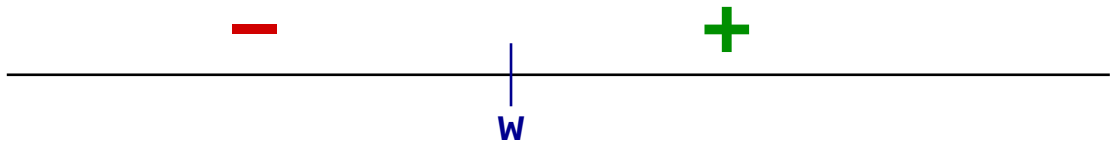
Supervised: for misclassification error $\leq \epsilon$, need $\approx 1/\epsilon$ labeled points.

Efficient search through hypothesis space

Threshold functions on the real line:

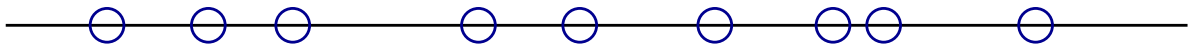
$H = \{h_w : w \in \mathbb{R}\}$

$h_w(x) = 1(x \geq w)$



Supervised: for misclassification error $\leq \epsilon$, need $\approx 1/\epsilon$ labeled points.

Active learning: instead, start with $1/\epsilon$ *unlabeled* points.

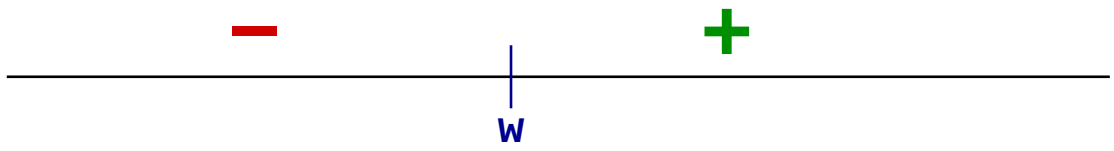


Efficient search through hypothesis space

Threshold functions on the real line:

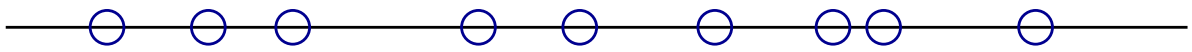
$H = \{h_w : w \in \mathbb{R}\}$

$h_w(x) = 1(x \geq w)$



Supervised: for misclassification error $\leq \epsilon$, need $\approx 1/\epsilon$ labeled points.

Active learning: instead, start with $1/\epsilon$ *unlabeled* points.



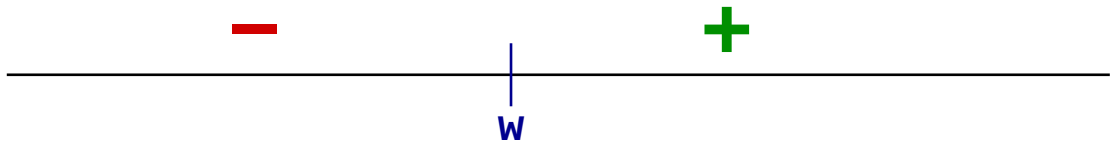
Binary search: need just $\log 1/\epsilon$ labels, from which the rest can be inferred. *Exponential improvement in label complexity!*

Efficient search through hypothesis space

Threshold functions on the real line:

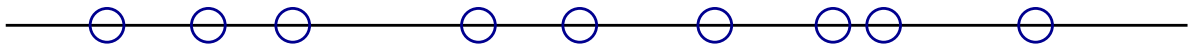
$H = \{h_w : w \in \mathbb{R}\}$

$h_w(x) = 1(x \geq w)$



Supervised: for misclassification error $\leq \epsilon$, need $\approx 1/\epsilon$ labeled points.

Active learning: instead, start with $1/\epsilon$ *unlabeled* points.



Binary search: need just $\log 1/\epsilon$ labels, from which the rest can be inferred. *Exponential improvement in label complexity!*

Challenges: Nonseparable data? Other hypothesis classes?

Some results of active learning theory

	Separable data	General (nonseparable) data
Aggressive	Query by committee (Freund, Seung, Shamir, Tishby, 97) Splitting index (D, 05)	
Mellow	Generic active learner (Cohn, Atlas, Ladner, 91)	A^2 algorithm (Balcan, Beygelzimer, L, 06) Disagreement coefficient (Hanneke, 07) Reduction to supervised (D, Hsu, Monteleoni, 2007) Importance-weighted approach (Beygelzimer, D, L, 2009)

Some results of active learning theory

	Separable data	General (nonseparable) data
Aggressive	Query by committee (Freund, Seung, Shamir, Tishby, 97) Splitting index (D, 05)	
Mellow	Generic active learner (Cohn, Atlas, Ladner, 91)	A^2 algorithm (Balcan, Beygelzimer, L, 06) Disagreement coefficient (Hanneke, 07) Reduction to supervised (D, Hsu, Monteleoni, 2007) Importance-weighted approach (Beygelzimer, D, L, 2009)

Issues:

Computational tractability

Are labels being used as efficiently as possible?

A generic mellow learner [CAL '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

Repeat for $t = 1, 2, \ldots$

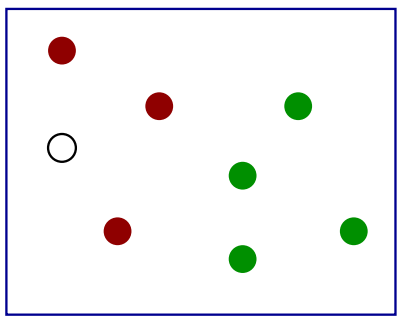
Receive unlabeled point x_t

If there is any disagreement within H_t about x_t's label:

query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

$H_{t+1} = H_t$



Is a label needed?

A generic mellow learner [CAL '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

Repeat for $t = 1, 2, \ldots$

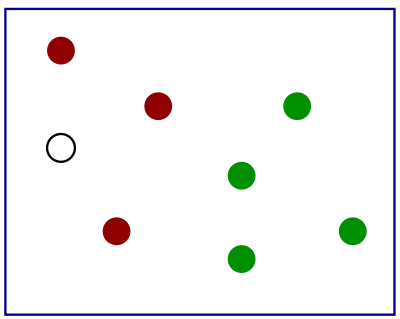
Receive unlabeled point x_t

If there is any disagreement within H_t about x_t's label:

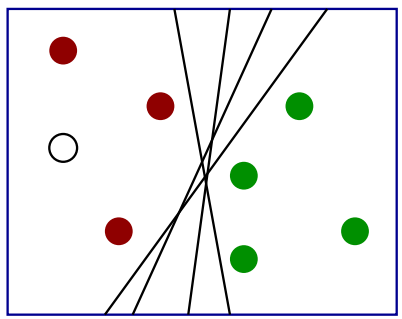
query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

$H_{t+1} = H_t$



Is a label needed?



$H_t =$ current candidate hypotheses

A generic mellow learner [CAL '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

Repeat for $t = 1, 2, \ldots$

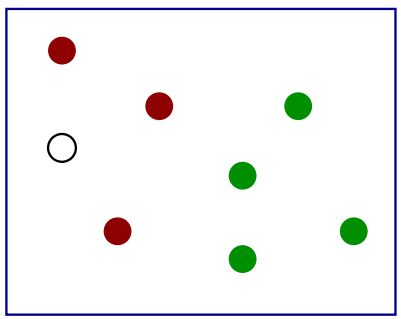
Receive unlabeled point x_t

If there is any disagreement within H_t about x_t's label:

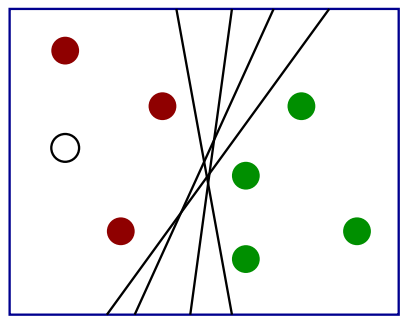
query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

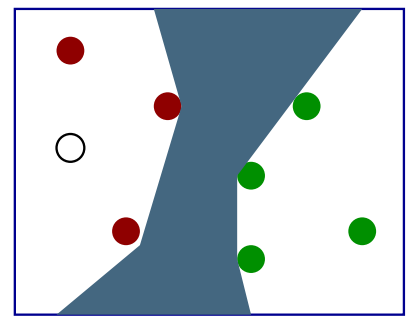
$H_{t+1} = H_t$



Is a label needed?



$H_t =$ current candidate hypotheses



Region of uncertainty

A generic mellow learner [CAL '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

Repeat for $t = 1, 2, \ldots$

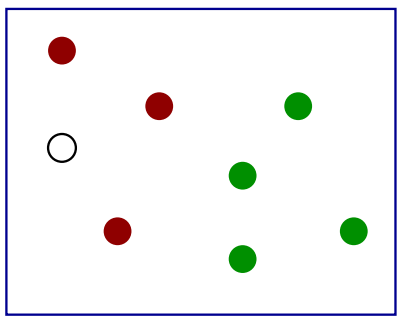
Receive unlabeled point x_t

If there is any disagreement within H_t about x_t's label:

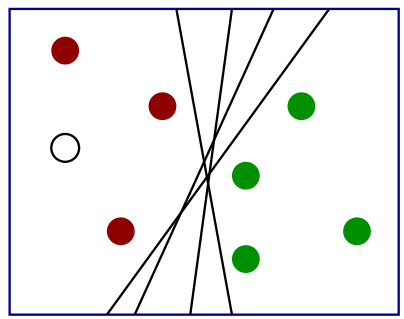
query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

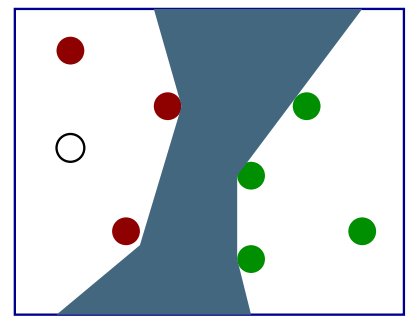
$H_{t+1} = H_t$



Is a label needed?



$H_t =$ current candidate hypotheses



Region of uncertainty

Problems: (1) intractable to maintain H_t; (2) nonseparable data.

Maintaining H_t

Explicitly maintaining H_t is intractable. Do it implicitly, by reduction to supervised learning.

Explicit version

$H_1 =$ hypothesis class
For $t = 1, 2, \ldots$:
  Receive unlabeled point x_t
  If disagreement in H_t about x_t's label:
    query label y_t of x_t
    $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$
  else:
    $H_{t+1} = H_t$

Implicit version

$S = \{\}$ (points seen so far)
For $t = 1, 2, \ldots$:
  Receive unlabeled point x_t
  If learn$(S \cup (x_t, 1))$ and learn$(S \cup (x_t, 0))$ both return an answer:
    query label y_t
  else:
    set y_t to whichever label succeeded
  $S = S \cup \{(x_t, y_t)\}$

Maintaining H_t

Explicitly maintaining H_t is intractable. Do it implicitly, by reduction to supervised learning.

Explicit version

$H_1 =$ hypothesis class
For $t = 1, 2, \ldots$:
  Receive unlabeled point x_t
  If disagreement in H_t about x_t's label:
    query label y_t of x_t
    $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$
  else:
    $H_{t+1} = H_t$

Implicit version

$S = \{\}$ (points seen so far)
For $t = 1, 2, \ldots$:
  Receive unlabeled point x_t
  If learn$(S \cup (x_t, 1))$ and learn$(S \cup (x_t, 0))$ both return an answer:
    query label y_t
  else:
    set y_t to whichever label succeeded
  $S = S \cup \{(x_t, y_t)\}$

This scheme is no worse than straight supervised learning. But can one bound the number of labels needed?

Label complexity [Hanneke]

The label complexity of CAL (mellow, separable) active learning can be captured by the the VC dimension d of the hypothesis and by a parameter θ called the *disagreement coefficient*.

Label complexity [Hanneke]

The label complexity of CAL (mellow, separable) active learning can be captured by the the VC dimension d of the hypothesis and by a parameter θ called the *disagreement coefficient*.

- Regular supervised learning, separable case.

  Suppose data are sampled iid from an underlying distribution. To get a hypothesis whose misclassification rate (on the underlying distribution) is $\leq \epsilon$ with probability ≥ 0.9, it suffices to have

  $$\frac{d}{\epsilon}$$

  labeled examples.

Label complexity [Hanneke]

The label complexity of CAL (mellow, separable) active learning can be captured by the the VC dimension d of the hypothesis and by a parameter θ called the *disagreement coefficient*.

- Regular supervised learning, separable case.

  Suppose data are sampled iid from an underlying distribution. To get a hypothesis whose misclassification rate (on the underlying distribution) is $\leq \epsilon$ with probability ≥ 0.9, it suffices to have

  $$\frac{d}{\epsilon}$$

  labeled examples.

- CAL active learner, separable case.

  Label complexity is

  $$\theta d \log \frac{1}{\epsilon}$$

Label complexity [Hanneke]

The label complexity of CAL (mellow, separable) active learning can be captured by the the VC dimension d of the hypothesis and by a parameter θ called the *disagreement coefficient*.

- Regular supervised learning, separable case.

  Suppose data are sampled iid from an underlying distribution. To get a hypothesis whose misclassification rate (on the underlying distribution) is $\leq \epsilon$ with probability ≥ 0.9, it suffices to have

  $$\frac{d}{\epsilon}$$

  labeled examples.

- CAL active learner, separable case.

  Label complexity is

  $$\theta d \log \frac{1}{\epsilon}$$

- There is a version of CAL for nonseparable data. (More to come!)

  If best achievable error rate is ν, suffices to have

  $$\theta \left(d \log^2 \frac{1}{\epsilon} + \frac{d\nu^2}{\epsilon^2} \right)$$

  labels. Usual supervised requirement: d/ϵ^2.

Disagreement coefficient [Hanneke]

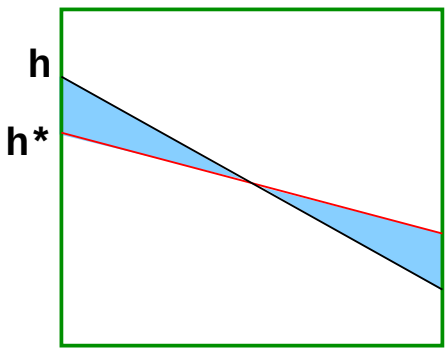
Let $\mathbb{P}$ be the underlying probability distribution on input space $\mathcal{X}$.
Induces (pseudo-)metric on hypotheses: $d(h, h') = \mathbb{P}[h(X) \neq h'(X)]$.
Corresponding notion of *ball* $B(h, r) = \{h' \in H : d(h, h') < r\}$.

Disagreement region of any set of candidate hypotheses $V \subseteq H$:

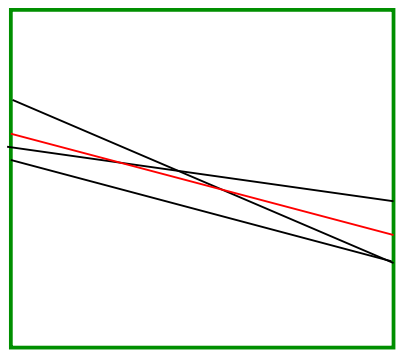
$$\mathrm{DIS}(V) = \{x : \exists h, h' \in V \text{ such that } h(x) \neq h'(x)\}.$$

Disagreement coefficient for target hypothesis $h^* \in H$:

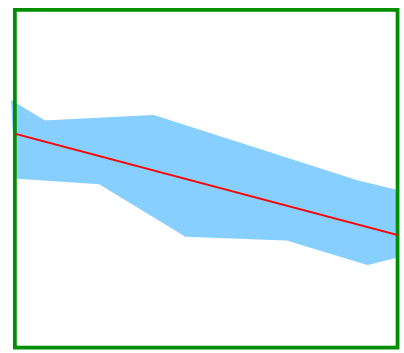
$$\theta = \sup_r \frac{\mathbb{P}[\mathrm{DIS}(B(h^*, r))]}{r}.$$



$d(h^*, h) = \mathbb{P}[\text{shaded region}]$



Some elements of $B(h^*, r)$



$\mathrm{DIS}(B(h^*, r))$

Disagreement coefficient: separable case

Let $\mathbb{P}$ be the underlying probability distribution on input space $\mathcal{X}$.
Let H_ϵ be all hypotheses in H with error $\leq \epsilon$. Disagreement region:

$$\mathrm{DIS}(H_\epsilon) = \{x : \exists h, h' \in H_\epsilon \text{ such that } h(x) \neq h'(x)\}.$$

Then disagreement coefficient is

$$\theta = \sup_\epsilon \frac{\mathbb{P}[\mathrm{DIS}(H_\epsilon)]}{\epsilon}.$$

Disagreement coefficient: separable case

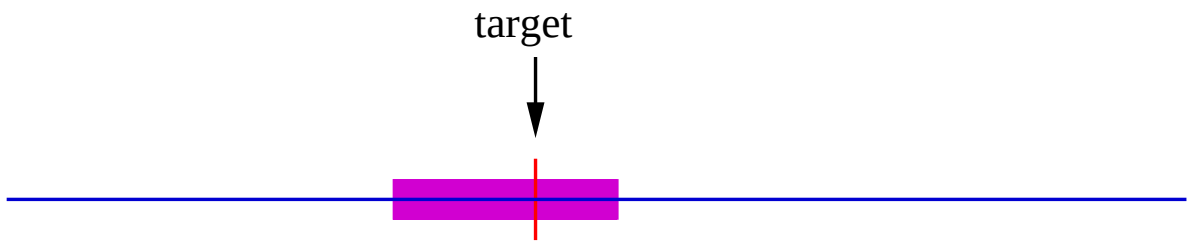
Let $\mathbb{P}$ be the underlying probability distribution on input space $\mathcal{X}$.
Let H_ϵ be all hypotheses in H with error $\leq \epsilon$. Disagreement region:

$$\mathrm{DIS}(H_\epsilon) = \{x : \exists h, h' \in H_\epsilon \text{ such that } h(x) \neq h'(x)\}.$$

Then disagreement coefficient is

$$\theta = \sup_\epsilon \frac{\mathbb{P}[\mathrm{DIS}(H_\epsilon)]}{\epsilon}.$$

Example: $H = \{\text{thresholds in } \mathbb{R}\}$, any data distribution.



Therefore $\theta = 2$.

Disagreement coefficient: examples [H '07, F '09]

Disagreement coefficient: examples [H '07, F '09]

- Thresholds in $\mathbb{R}$, any data distribution.

  $$\theta = 2.$$

  Label complexity $O(\log 1/\epsilon)$.

Disagreement coefficient: examples [H '07, F '09]

- Thresholds in $\mathbb{R}$, any data distribution.

  $$\theta = 2.$$

  Label complexity $O(\log 1/\epsilon)$.

- Linear separators through the origin in $\mathbb{R}^d$, uniform data distribution.

  $$\theta \leq \sqrt{d}.$$

  Label complexity $O(d^{3/2} \log 1/\epsilon)$.

Disagreement coefficient: examples [H '07, F '09]

- Thresholds in $\mathbb{R}$, any data distribution.

  $$\theta = 2.$$

  Label complexity $O(\log 1/\epsilon)$.

- Linear separators through the origin in $\mathbb{R}^d$, uniform data distribution.

  $$\theta \leq \sqrt{d}.$$

  Label complexity $O(d^{3/2} \log 1/\epsilon)$.

- Linear separators in $\mathbb{R}^d$, smooth data density bounded away from zero.

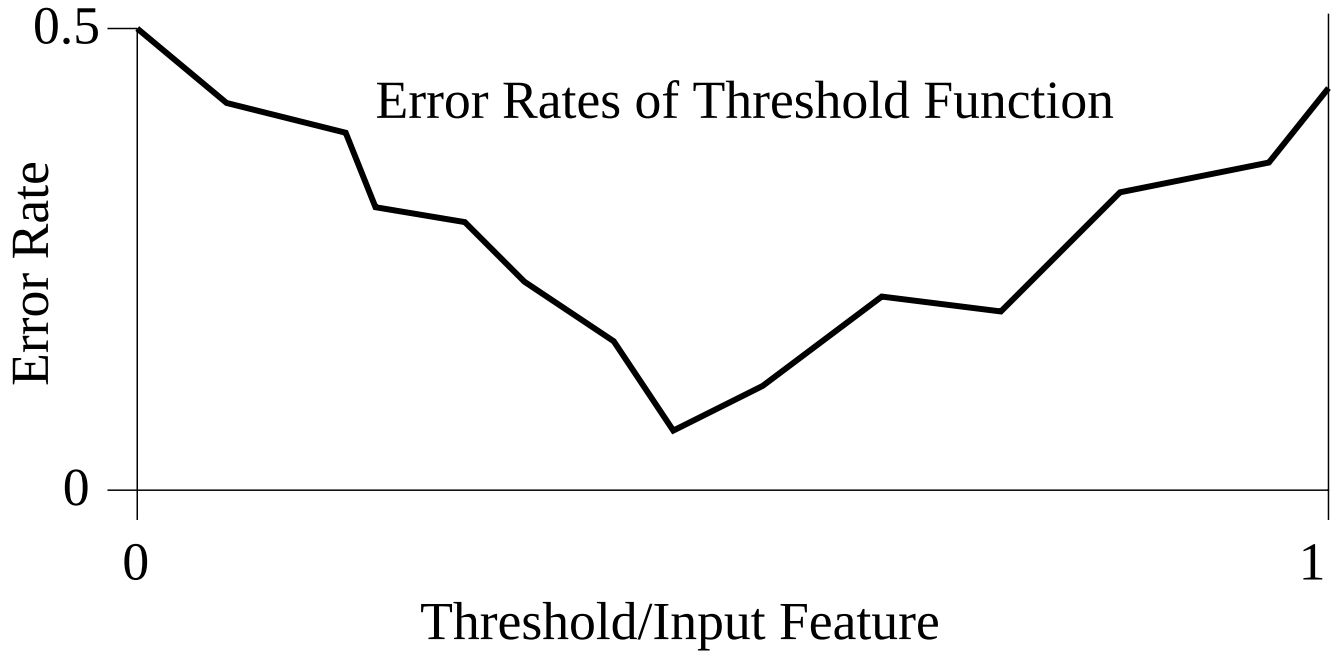
  $$\theta \leq c(h^*)d$$

  where $c(h^*)$ is a constant depending on the target h^*.
  Label complexity $O(c(h^*)d^2 \log 1/\epsilon)$.

Gimme the algorithms

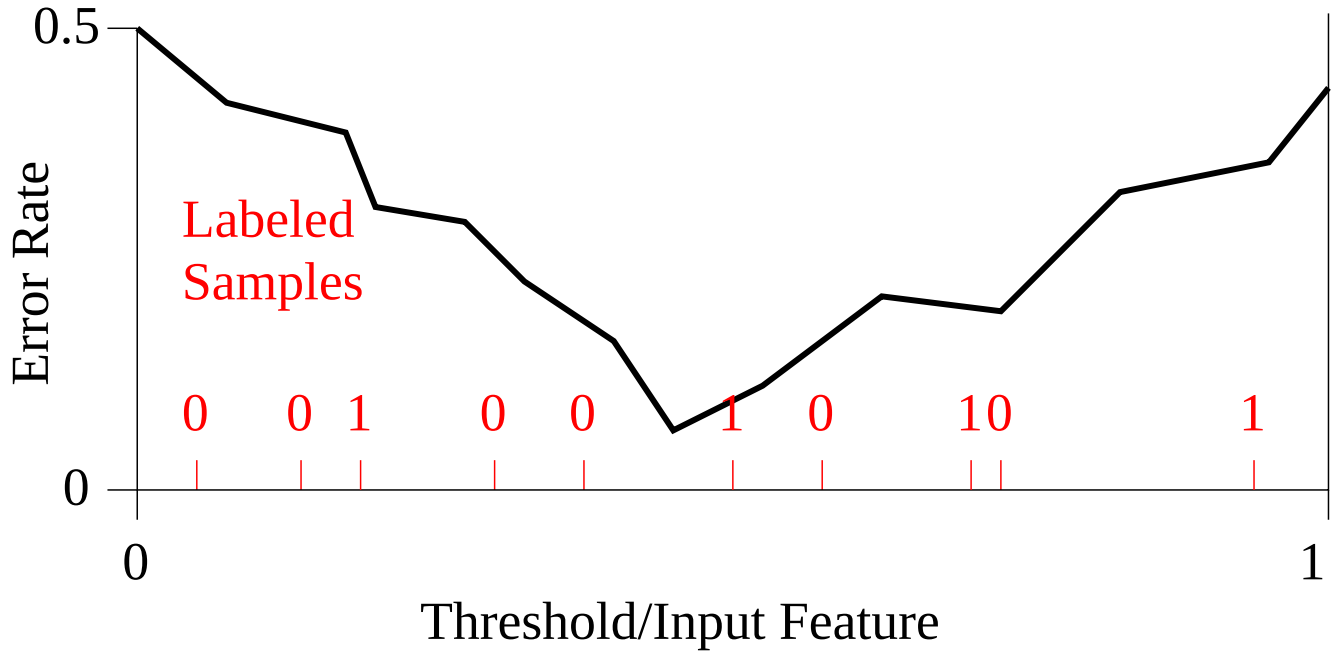
1. The A^2 algorithm.
2. Limitations
3. IWAL

The A^2 Algorithm in Action



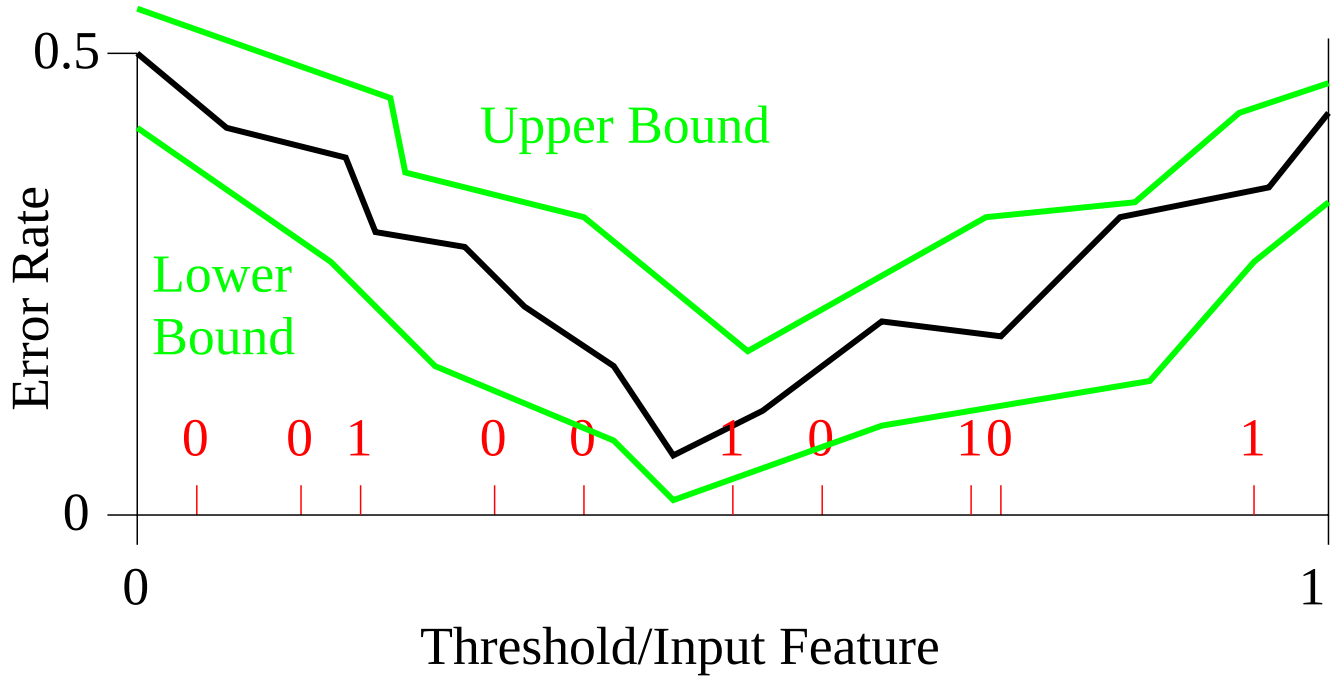
Problem: find the optimal threshold function on the $[0, 1]$ interval in a noisy domain.

Sampling



Label samples at random.

Bounding

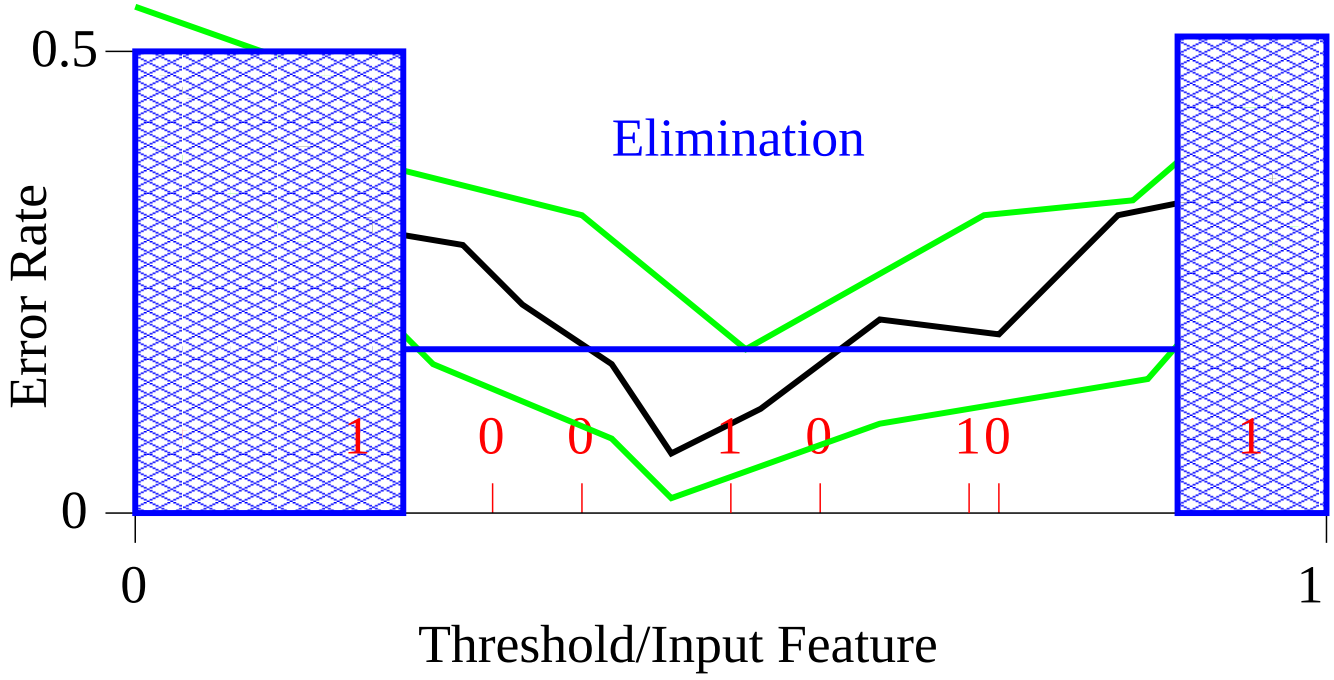


Compute upper and lower bounds on the error rate of each hypothesis.

Theorem: For all H, for all D, for all numbers of samples m,

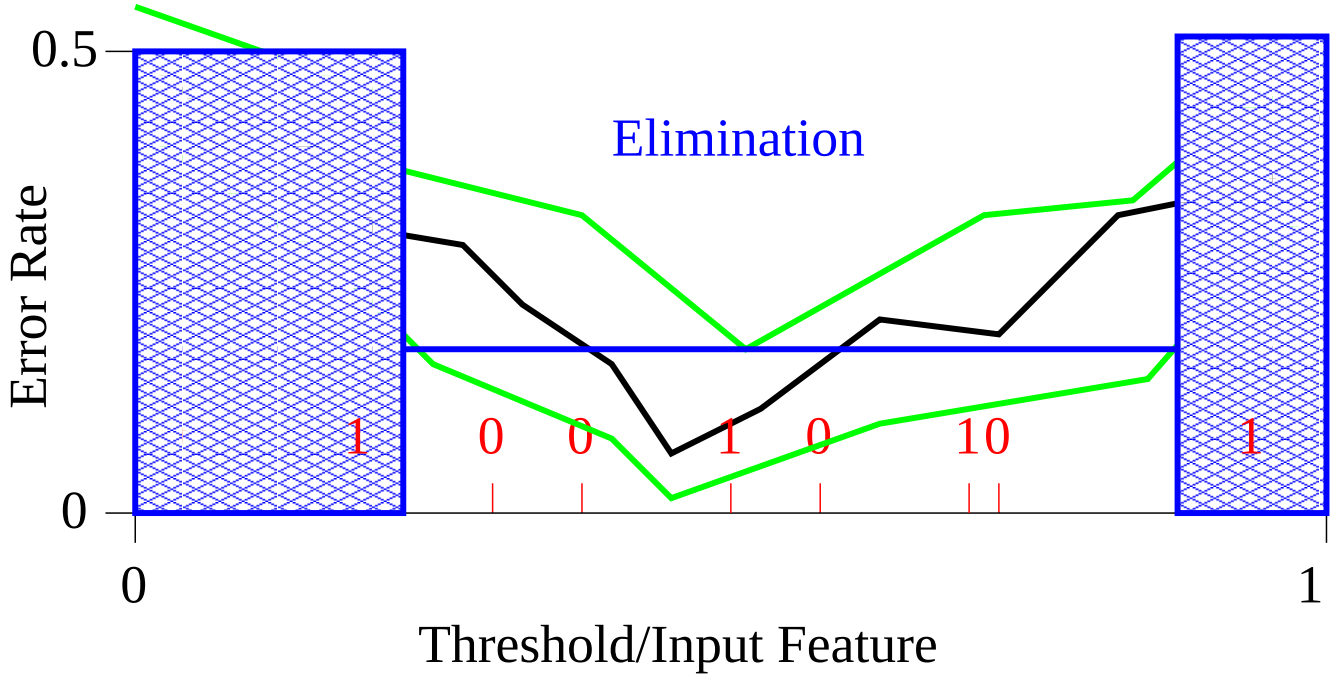
$$\Pr(|\text{true error rate} - \text{empirical error rate}| \leq f(H, \delta, m)) \geq 1 - \delta$$

Eliminating



Chop away part of the hypothesis space, implying that you cease to care about part of the feature space.

Eliminating



Chop away part of the hypothesis space, implying that you cease to care about part of the feature space. Recurse!

What is this master algorithm?

Let $e(h, D) = \Pr_{x,y \sim D}(h(x) \neq y)$

What is this master algorithm?

Let $e(h, D) = \Pr_{x,y \sim D}(h(x) \neq y)$

Let $\mathrm{UB}(S, h) =$ upper bound on $e(h, D)$ assuming S IID from D.
Let $\mathrm{LB}(S, h) =$ lower bound on $e(h, D)$ assuming S IID from D.

What is this master algorithm?

Let $e(h, D) = \Pr_{x,y \sim D}(h(x) \neq y)$

Let $\mathrm{UB}(S, h) =$ upper bound on $e(h, D)$ assuming S IID from D.
Let $\mathrm{LB}(S, h) =$ lower bound on $e(h, D)$ assuming S IID from D.

$\mathrm{Disagree}(H, D) = \Pr_{x,y \sim D}(\exists h, h' \in H : h(x) \neq h'(x))$

What is this master algorithm?

Let $e(h, D) = \Pr_{x,y \sim D}(h(x) \neq y)$

Let $\mathrm{UB}(S, h) =$ upper bound on $e(h, D)$ assuming S IID from D.
Let $\mathrm{LB}(S, h) =$ lower bound on $e(h, D)$ assuming S IID from D.

$\mathrm{Disagree}(H, D) = \Pr_{x,y \sim D}(\exists h, h' \in H : h(x) \neq h'(x))$

$\mathrm{Done}(H, D) =$

$$(\min_{h \in H} \mathrm{UB}(S, h) - \min_{h \in H} \mathrm{LB}(S, h))\mathrm{Disagree}(H, D)$$

Agnostic_Active (error rate ϵ, classifiers H)

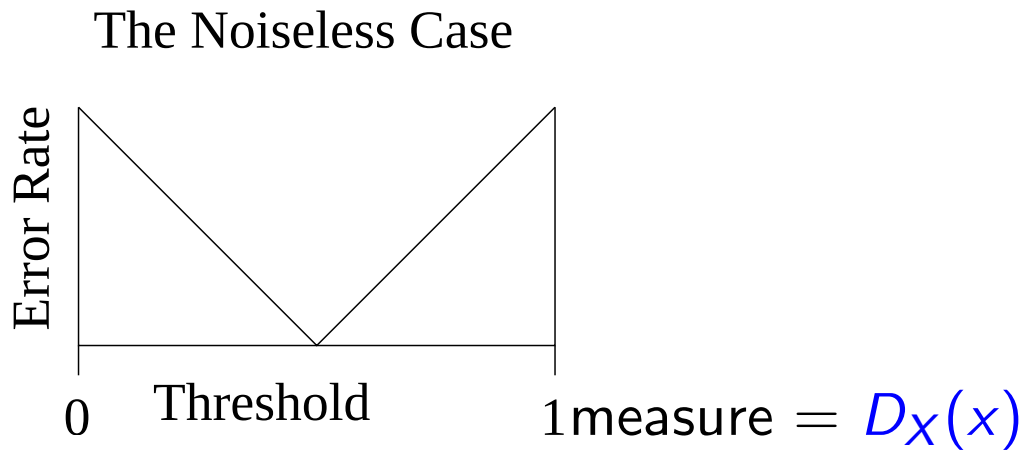
while $\mathrm{Done}(H, D) > \epsilon$:
  $S = \emptyset$, $H' = H$
  while $\mathrm{Disagree}(H', D) \geq \frac{1}{2}\mathrm{Disagree}(H, D)$:
    if $\mathrm{Done}(H', D) < \epsilon$: return $h \in H'$
    $S' =$ $2|S| + 1$ unlabeled x which H disagrees on
    $S = \{(x, \mathrm{Label}(x)) : x \in S'\}$
    $H' \leftarrow \{h \in H : \mathrm{LB}(S, h) \leq \min_{h' \in H} \mathrm{UB}(S, h')\}$
  $H \leftarrow H'$
return $h \in H$

Agnostic_Active: result

Theorem: There exists an algorithm Agnostic_Active that:

1. (Correctness) For all H, D, ϵ with probability 0.99 returns an ϵ-optimal c.
2. (Fall-Back) For all H, D the number of labeled samples required is $O(\mathrm{Batch})$.
3. (Structured Low noise) For all H, D with disagreement coefficient θ and $d = VC(H)$ with $\nu < \epsilon$, $\tilde{O}\left(\theta^2 d \ln^2 \frac{1}{\epsilon}\right)$ labeled examples suffice.
4. (Structured High noise) For all H, D with disagreement coefficient θ and $d = VC(H)$ with $\nu > \epsilon$, $\tilde{O}\left(\frac{\theta^2 \nu^2}{\epsilon^2} d\right)$ labeled examples suffice.

Proof of low noise/high speedup case



ν noise can not significantly alter this picture for ν small.

$\Rightarrow$ constant classifier fraction has error rate > 0.25.

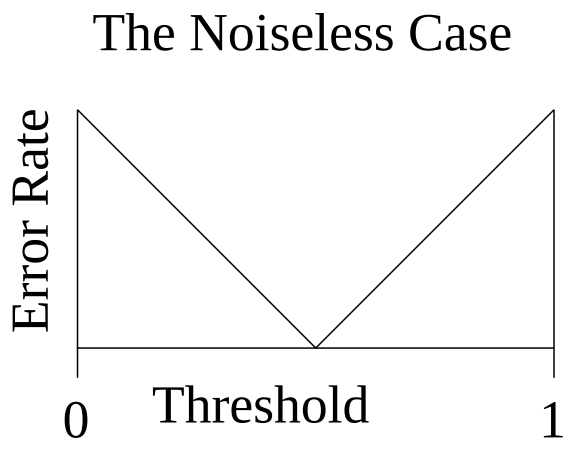
$\Rightarrow$ $O(\ln \frac{1}{\delta})$ labeled examples gives error rates of all thresholds up to tolerance $\frac{1}{8}$ with probability $1 - \delta$

General low noise/high speedup case

$\Rightarrow$ constant classifier fraction has error rate > 0.25.

$\Rightarrow$ $O(\theta^2(d + \ln \frac{1}{\delta}))$ labeled examples gives error rates of all classifiers up to tolerance $\frac{1}{8}$ with probability $1 - \delta$

Proof of low noise/high speedup case II



$\Rightarrow \frac{1}{2}$ fraction of "far" classifiers eliminatable via bound algebra.

$\Rightarrow$ Problem recurses. Spreading δ across recursions implies result.

Proof of high noise/low speedup case

low noise/high speedup theorem $\Rightarrow \mathrm{Disagree}(C, D) \simeq \nu$ after few examples.

$$\begin{aligned}&\mathrm{Done}(C, D)\\ &= [\min_{c \in C} \mathrm{UB}(S, c) - \min_{c \in C} \mathrm{LB}(S, c)]\mathrm{Disagree}(C, D)\\ &= [\min_{c \in C} \mathrm{UB}(S, c) - \min_{c \in C} \mathrm{LB}(S, c)]\nu\end{aligned}$$

$\Rightarrow$ Computing bounds to error rate $\frac{\epsilon}{\nu}$ makes $\mathrm{Done}(C, D) \simeq \epsilon$

$\tilde{O}\left(\frac{\theta^2 \nu^2}{\epsilon^2} d\right)$ samples suffice.

Gimme the algorithms

1. The A^2 algorithm
2. Limitations
3. IWAL

What's wrong with A^2?

What's wrong with A^2?

1. Unlabeled complexity You need infinite unlabeled data to measure $\mathrm{Disagree}(C, D)$ to infinite precision.
2. Computation You need to enumerate and check hypotheses—exponentially slower and exponentially more space than common learning algorithms.
3. Label Complexity Can't get logarithmic label complexity for $\epsilon < \nu$.
4. Label Complexity Throwing away examples from previous iterations can't be optimal, right?
5. Label Complexity Bounds are often way too loose.
6. Generality We care about more than 0/1 loss.

What's wrong with A^2?

1. Unlabeled complexity [DHM07]
2. Computation [DHM07] partially, [Beygelzimer, D, L, 2009], partially.
3. Label Complexity [Kaariainen 2006], Log is impossible for small ϵ.
4. Label Complexity [DHM07], use all labels for all decisions.
5. Label Complexity [BDL09] Importance weights partially address loose bounds.
6. Generality [BDL09] Importance weights address other losses.

An improved lower bound for Active Learning

Theorem: [BDL09] For all H with $VC(H) = d$, for all $\epsilon, \nu > 0$ with $\epsilon < \frac{\nu}{2} < \frac{1}{8}$, there exists D, such that all active learning algorithms require:

$$\Omega\left(\frac{d\nu^2}{\epsilon^2}\right)$$

samples to achieve to find an h satisfying $e(h, D) < \nu + \epsilon$.

An improved lower bound for Active Learning

Theorem: [BDL09] For all H with $VC(H) = d$, for all $\epsilon, \nu > 0$ with $\epsilon < \frac{\nu}{2} < \frac{1}{8}$, there exists D, such that all active learning algorithms require:

$$\Omega\left(\frac{d\nu^2}{\epsilon^2}\right)$$

samples to achieve to find an h satisfying $e(h, D) < \nu + \epsilon$.

Implication: For $\epsilon = \sqrt{\frac{d\nu}{T}}$ as in supervised learning, all label complexity bounds must have a dependence on:

$$\frac{d\nu^2}{\epsilon^2} = \frac{d\nu^2}{\frac{d\nu}{T}} = T\nu$$

An improved lower bound for Active Learning

Theorem: [BDL09] For all H with $VC(H) = d$, for all $\epsilon, \nu > 0$ with $\epsilon < \frac{\nu}{2} < \frac{1}{8}$, there exists D, such that all active learning algorithms require:

$$\Omega\left(\frac{d\nu^2}{\epsilon^2}\right)$$

samples to achieve to find an h satisfying $e(h, D) < \nu + \epsilon$.

Implication: For $\epsilon = \sqrt{\frac{d\nu}{T}}$ as in supervised learning, all label complexity bounds must have a dependence on:

$$\frac{d\nu^2}{\epsilon^2} = \frac{d\nu^2}{\frac{d\nu}{T}} = T\nu$$

Proof sketch:

1. $\mathrm{VC}(H) = d$ means there are d inputs x where the prediction can go either way.
2. Make $|P(y = 1|x) - P(y = 0|x)| = \frac{2\epsilon}{\nu+2\epsilon}$ on these points.
3. Apply coin flipping lower bound for determining whether heads or tails is most common.

Gimme the algorithms

1. The A^2 algorithm.
2. Limitations
3. IWAL

IWAL(subroutine Rejection-Threshold)

$S = \emptyset$

While (unlabeled examples remain)

1. Receive unlabeled example x.
2. Set $p =$ Rejection-Threshold(x, S).
3. If $U(0, 1) \leq p$, get label y, and add $(x, y, \frac{1}{p})$ to S.
4. Let $h =$ Learn(S).

IWAL: fallback result

Theorem: For all choices of Rejection-Threshold if for all x, S:

$$\text{Rejection-Threshold}(x, S) \geq p_{\min}$$

then IWAL is at most a constant factor worse than supervised learning.
Proof: Standard PAC/VC analysis, except with a martingale.
But is it better than supervised learning?

Loss-weighting(unlabeled example x, history S)

1. Let H_S =hypotheses that we might eventually choose, assuming the distribution is IID.
2. Return $\max_{f,g \in H_S, y} \ell(f(x), y) - \ell(g(x), y)$

Loss-weighting safety

(Not trivial, because sometimes 0 is returned)
Theorem: IWAL(Loss-weighting) competes with supervised learning.

But does it give us speedups?

The Slope Asymmetry

For two predictions z,z' what is the ratio of the maximum and minimum change in loss as a function of y?

$$C_\ell = \sup_{z,z'} \frac{\max_{y \in Y} |\ell(z,y) - \ell(z',y)|}{\min_{y \in Y} |\ell(z,y) - \ell(z',y)|}$$

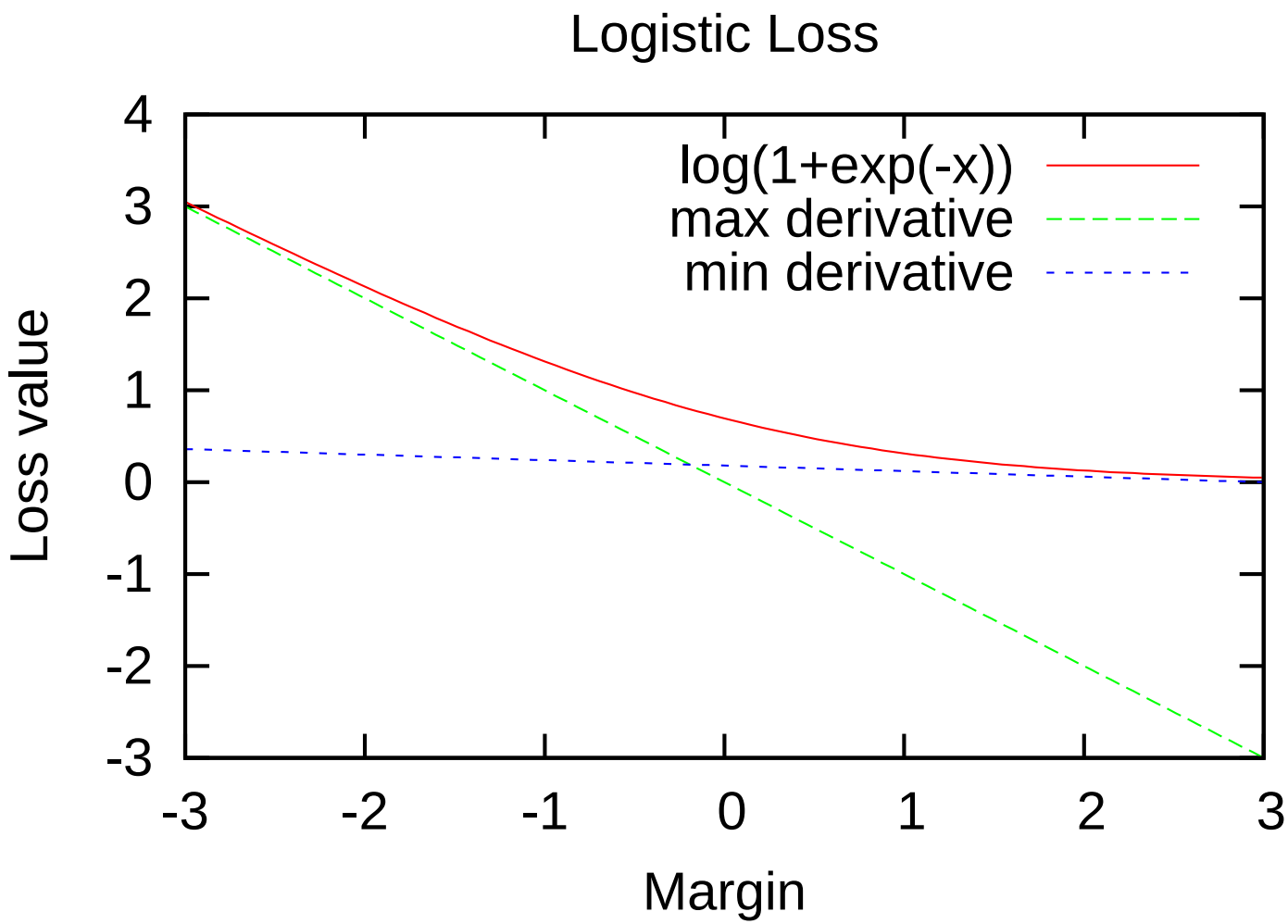
$=$ generalized maximum derivative ratio of loss function ℓ.

$= 1$ for 0/1 loss.

$= 1$ for hinge loss on $[-1, 1]$ (i.e. normalized margin SVM)

$\leq 1 + e^B$ for logistic in $[-B, B]$

∞ for squared loss.


Logistic Loss
log(1+exp(-x))
max derivative
min derivative
Loss value
Margin
4
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3

Generalized Disagreement Coefficient

Was:

$$\theta = \sup_r \frac{\mathbb{P}[\mathrm{DIS}(B(h^*, r))]}{r}$$
$$= \sup_r \frac{E_{x\sim D} I[\exists h, h' \in B(h^*, r) : h(x) \neq h'(x)]}{r}$$

Generalized:

$$\theta = \sup_r \frac{E_{x\sim D} \max_{h\in B(h^*,r)} \max_y |\ell(h(x), y) - \ell(h^*(x), y)|}{r}$$

Big disagreement coefficient $\Leftrightarrow$ near optimal hypotheses often disagree in loss.

Sufficient conditions for IWAL(Loss-weighting)

Let $\nu = \min_{h \in H} E_{x,y \sim D} \ell(h(x), y) =$ minimum possible loss rate.

Theorem: For all learning problems D, for all hypothesis sets H, the label complexity after T unlabeled samples is at most:

$$\theta C_l \left(\nu T + \sqrt{T \ln \frac{|H| T}{\delta}} \right)$$

(up to constants)

Experiments in the convex case

If:

1. Loss is convex
2. Representation is linear

Then computation is easy.(*)

(*) caveat: you can't quite track H_S perfectly, so must have minimum sample probability for safety.

Experiments in the nonconvex case

But, sometimes a linear predictor isn't the best predictor. What do we do?

Bootstrap(unlabeled example x, history S)

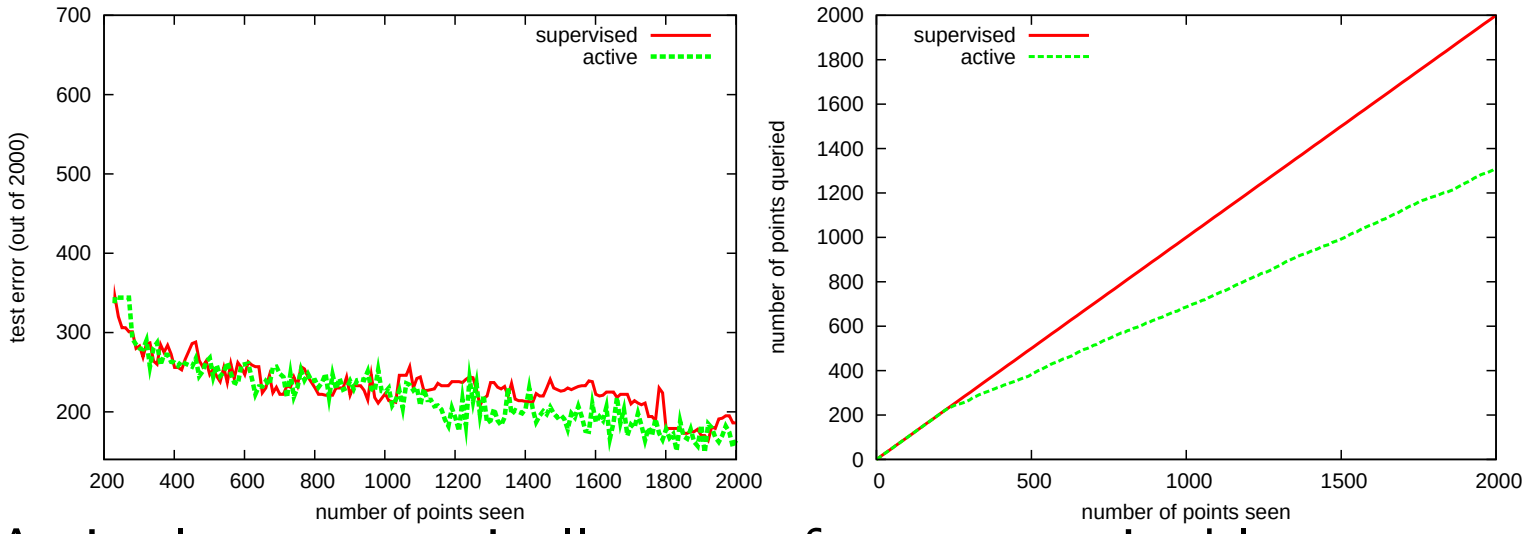
1. If $|S| < t$ return 1.
2. If $|S| = t$ train 10 hypotheses $H' = \{h_1, ..., h_{10}\}$ using a bootstrap sample.
3. If $|S| \geq t$ return $p_{\min} + (1 - p_{\min}) \max_{f,g \in H, y} \ell(f(x), y) - \ell(g(x), y)$

Label savings results

Logistic/interior point	Online Constrained	MNist	65%
J48	Batch Bootstrap	MNist	35%
J48	Batch Bootstrap	Adult	60%
J48	Batch Bootstrap	Pima	32%
J48	Batch Bootstrap	Yeast	19%
J48	Batch Bootstrap	Spambase	55%
J48	Batch Bootstrap	Waveform	16%
J48	Batch Bootstrap	Letter	25%

In all cases active learner's prediction performance $\simeq$ supervised prediction performance. (In fact, more often better than worse.)

An Example Plot: J48/Batch Bootstrap/MNist 3 vs 5



Active learner marginally outperforms supervised learner on the same number of examples while using only 2/3rds as many labels.

What the theory says

Active Learning applies anywhere supervised learning applies.
Current techniques help best when:

1. Discrete loss Hard choices with discrete losses are made.
2. Low noise The best predictor has small error.
3. Constrained H The set of predictors you care about is constrained.
4. Low Data You don't already have lots of labeled data.

Future work for all of us

1. Foundations Is active learning possible in a fully adversarial setting?
2. Application Is an active learning reduction to supervised possible without constraints?
3. Extension What about other settings for interactive learning? (structured? partial label? Differing oracles with differing expertise?)
4. Empirical Can we achieve good active learning performance with a consistent algorithm on a state-of-the-art problem?

Further discussion at http://hunch.net

Bibliography

1. Yotam Abramson and Yoav Freund, Active Learning for Visual Object Recognition, UCSD tech report.
2. Nina Balcan, Alina Beygelzimer, John Langford, Agnostic Active Learning. ICML 2006
3. Alina Beygelzimer, Sanjoy Dasgupta, and John Langford, Importance Weighted Active Learning, ICML 2009.
4. David Cohn, Les Atlas and Richard Ladner. Improving generalization with active learning, Machine Learning 15(2):201-221, 1994.
5. Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. ICML 2008.
6. Sanjoy Dasgupta, Coarse sample complexity bounds for active learning. NIPS 2005.

Bibliography II

1. Sanjoy Dasgupta, Daniel J. Hsu, and Claire Monteleoni. A general agnostic active learning algorithm. NIPS 2007.
2. Yoav Freund, H. Sebastian Seung, Eli Shamir, and Naftali Tishby, Selective Sampling Using the Query by Committee Algorithm, Machine Learning, 28, 133-168, 1997.
3. Hanneke, S. A Bound on the Label Complexity of Agnostic Active Learning. ICML 2007.
4. Manfred K. Warmuth, Gunnar Ratsch, Michael Mathieson, Jun Liao, and Christian Lemmon, Active Learning in the Drug Discovery Process, NIPS 2001.
5. Xiaojin Zhu, John Lafferty and Zoubin Ghahramani, Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions, ICML 2003 workshop