Recent advances in active learning

Sanjoy Dasgupta

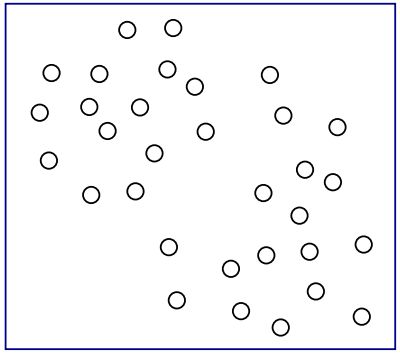
University of California, San Diego

Exploiting unlabeled data

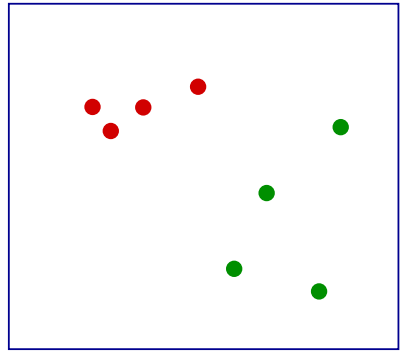
A lot of unlabeled data is plentiful and cheap, eg.

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

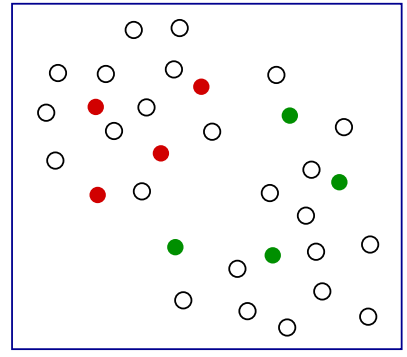
But labeling can be expensive.



Unlabeled points



Supervised learning



Semisupervised and active learning

Typical heuristics for active learning

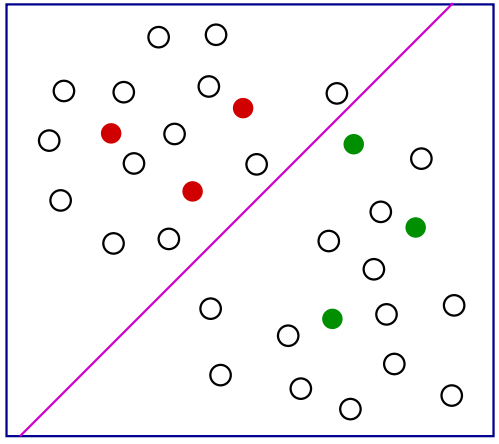
Start with a pool of unlabeled data

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,...)



How to analyze such schemes?

The statistical learning theory framework

Unknown, underlying distribution $\mathbb{P}$ on the (data, label) space.
Hypothesis class H of candidate classifiers.
Target: the $h^* \in H$ that has fewest errors on $\mathbb{P}$.

Get n samples from $\mathbb{P}$, choose $h_n \in H$ that does well on these.

We'd like: $h_n \to h^*$, as rapidly as possible.

Typical heuristics for active learning

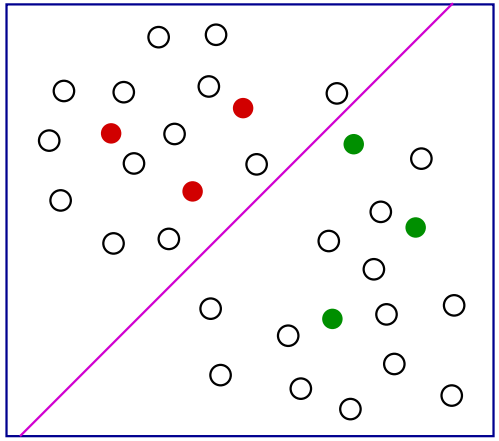
Start with a pool of unlabeled data

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,...)



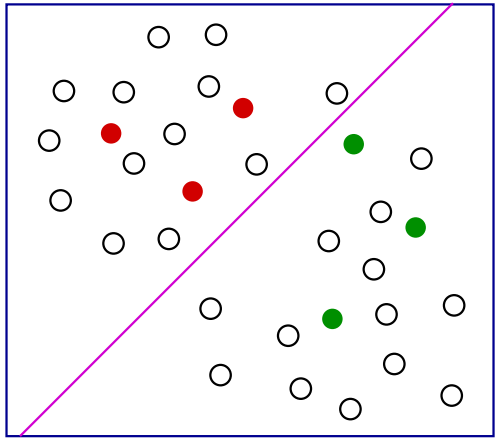
Typical heuristics for active learning

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

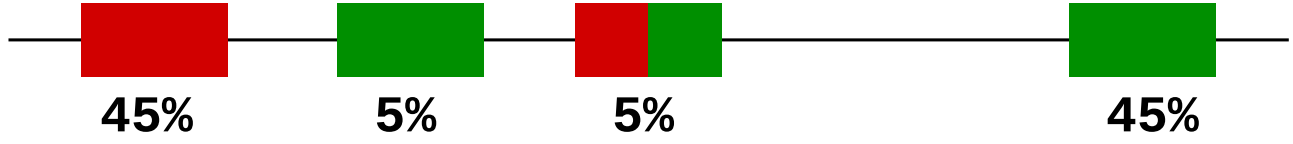
Start with a pool of unlabeled data

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



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-Velipasaoglu-Pedersen '03.

Sampling bias

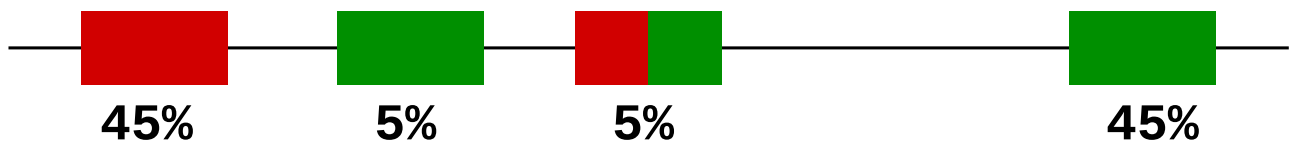
Start with a pool of unlabeled data

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



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-Velipasaoglu-Pedersen '03.

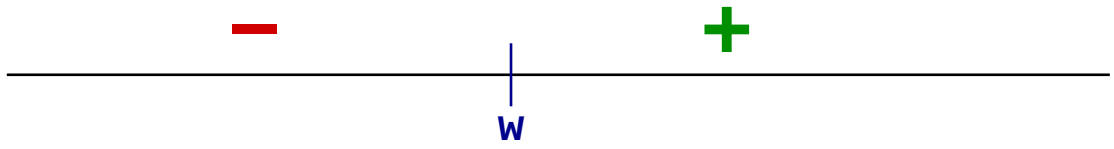
Henceforth, sine qua non: *statistical consistency*.

Can adaptive querying really help?

Threshold functions on the real line:

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

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



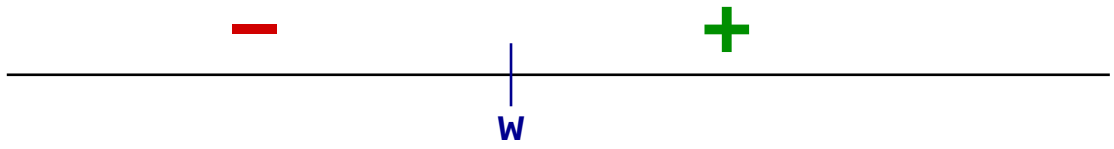
Supervised: for misclassification error $\leq \epsilon$, need $\approx 1/\epsilon$ labeled points.

Can adaptive querying really help?

Threshold functions on the real line:

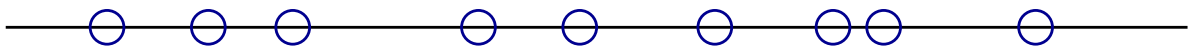
$H = \{h_w : w \in \mathbb{R}\}$

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



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

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



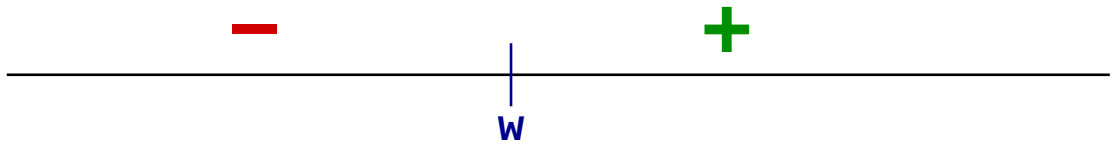
Binary search: need just $\log 1/\epsilon$ labels, from which the rest can be inferred. *Exponential improvement in label complexity*.

Can adaptive querying really help?

Threshold functions on the real line:

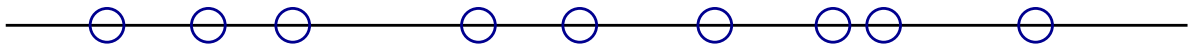
$H = \{h_w : w \in \mathbb{R}\}$

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



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 mellow learner (Cohn, Atlas, Ladner, 91)	A^2 algorithm (Balcan, Beygelzimer, Langford, 06) Disagreement coefficient (Hanneke, 07) Reduction to supervised (D, Hsu, Monteleoni, 07) Importance-weighted approach (Beygelzimer, D, Langford, 09)

Also: Friedman '09, Koltchinskii '10, Hsu-Kakade-Langford-Zhang '11.

Minimax analyses of some canonical cases: Castro-Nowak '07.

Issues guiding development of the theory:

- Computational tractability
- Are labels being used as efficiently as possible?

Three ways to manage sampling bias

1. Label everything.
2. Use importance weighting.
3. Explicitly manage sampling regions.

A generic mellow learner [Cohn-Atlas-Ladner '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

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

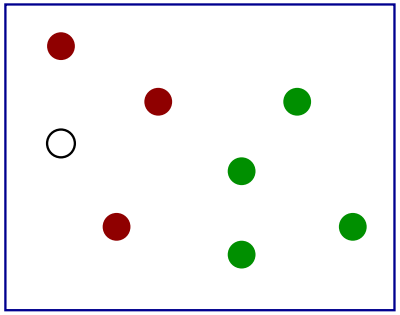
Receive unlabeled point x_t

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 [Cohn-Atlas-Ladner '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

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

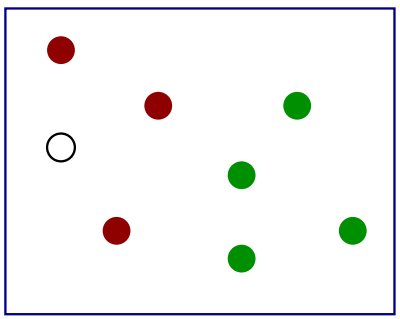
Receive unlabeled point x_t

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

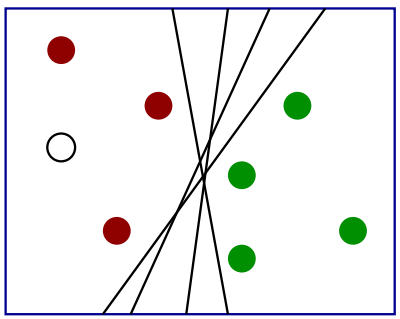
query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

$H_{t+1} = H_t$



Is a label needed?



$H_t =$ current candidate hypotheses

A generic mellow learner [Cohn-Atlas-Ladner '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

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

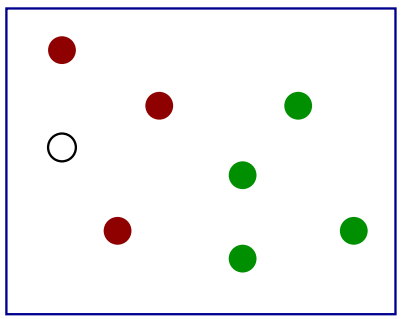
Receive unlabeled point x_t

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

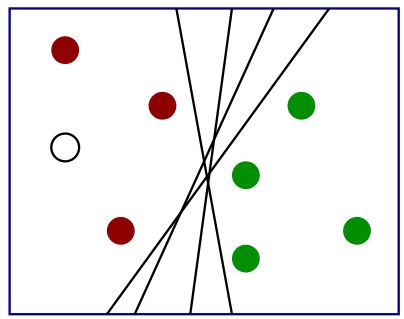
query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

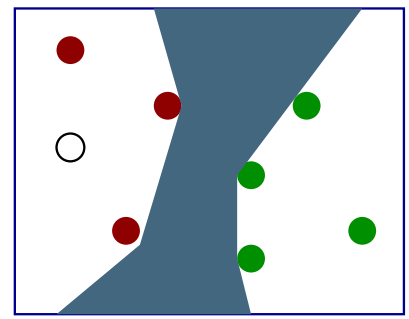
$H_{t+1} = H_t$



Is a label needed?



$H_t =$ current candidate hypotheses



Region of uncertainty

A generic mellow learner [Cohn-Atlas-Ladner '91]

For *separable* data that is streaming in.

$H_1 =$ hypothesis class

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

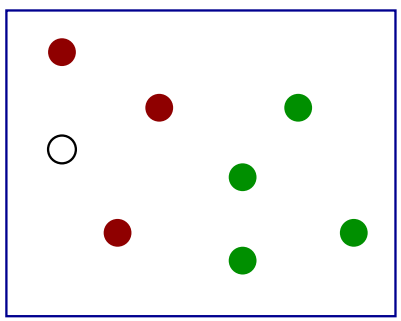
Receive unlabeled point x_t

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

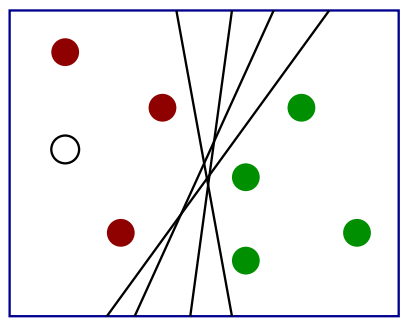
query label y_t and set $H_{t+1} = \{h \in H_t : h(x_t) = y_t\}$

else

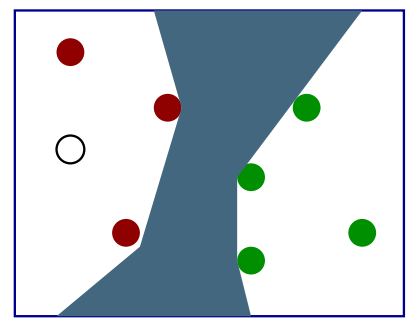
$H_{t+1} = H_t$



Is a label needed?



$H_t =$ current candidate hypotheses



Region of uncertainty

Issues: (1) intractable to maintain H_t; (2) nonseparable data; (3) how many labels are used?

(1) Maintaining H_t [D-Hsu-Monteleoni '07]

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
  $S = S \cup \{(x_t, y_t)\}$

(2) Dealing with nonseparable data [DHM '07]

Same general idea: query points you're at all unsure about.

(2) Dealing with nonseparable data [DHM '07]

Same general idea: query points you're at all unsure about.

- Set of candidate hypotheses H_t is now defined a bit differently

(2) Dealing with nonseparable data [DHM '07]

Same general idea: query points you're at all unsure about.

- Set of candidate hypotheses H_t is now defined a bit differently
- *Every data point x_t gets labeled.* Two cases:
  - There is disagreement in H_t about x_t's label. Query it.
  - Otherwise, assign it the label you're sure it has.

(2) Dealing with nonseparable data [DHM '07]

Same general idea: query points you're at all unsure about.

- Set of candidate hypotheses H_t is now defined a bit differently
- *Every data point x_t gets labeled.* Two cases:
  - There is disagreement in H_t about x_t's label. Query it.
  - Otherwise, assign it the label you're sure it has.
- At time t, we have t *labeled* points. Let $L_t(h)$ be the $0-1$ loss of hypothesis h on this data, and let h_t^* be the minimizer of this.

(2) Dealing with nonseparable data [DHM '07]

Same general idea: query points you're at all unsure about.

- Set of candidate hypotheses H_t is now defined a bit differently
- *Every data point x_t gets labeled.* Two cases:
  - There is disagreement in H_t about x_t's label. Query it.
  - Otherwise, assign it the label you're sure it has.
- At time t, we have t *labeled* points. Let $L_t(h)$ be the $0-1$ loss of hypothesis h on this data, and let h_t^* be the minimizer of this.
- $H_{t+1} := \{h \in H_t : L_t(h) \leq L_t(h_t^*) + \Delta_t\}$, for $\Delta_t \approx \sqrt{VC(H)/t}$.

(If instead we set $\Delta_t \equiv 0$, we'd get back CAL.)

(2) Dealing with nonseparable data [DHM '07]

Same general idea: query points you're at all unsure about.

- Set of candidate hypotheses H_t is now defined a bit differently
- *Every data point x_t gets labeled.* Two cases:
  - There is disagreement in H_t about x_t's label. Query it.
  - Otherwise, assign it the label you're sure it has.
- At time t, we have t *labeled* points. Let $L_t(h)$ be the $0-1$ loss of hypothesis h on this data, and let h_t^* be the minimizer of this.
- $H_{t+1} := \{h \in H_t : L_t(h) \leq L_t(h_t^*) + \Delta_t\}$, for $\Delta_t \approx \sqrt{VC(H)/t}$.

(If instead we set $\Delta_t \equiv 0$, we'd get back CAL.)

Δ_t comes out of a standard generalization bound. Guarantees:

- H_t always contains the optimal hypothesis; call it h^*.
- Every "inferred" label is consistent with h^*.
- The actual labels might differ from the inferred labels. But in the resulting distributional shift $\mathbb{P} \to \mathbb{P}'$, the *relative* ordering of hypotheses (by loss) is unchanged.

(3) Label complexity bounds [Hanneke '07, DHM '07]

The label complexity of mellow active learning can be captured by the *disagreement coefficient* θ and the VC dimension d.

- Separable case (CAL).
  To achieve misclassification rate ϵ with probability 0.9, suffices to have
  $$\theta d \log \frac{1}{\epsilon}$$
  labels. Usual supervised requirement: d/ϵ.
- Nonseparable case.
  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: $d/\epsilon + d\nu/\epsilon^2$.
- Lower bound.
  In the nonseparable case, a factor of $d\nu^2/\epsilon^2$ is inevitable.

Disagreement coefficient: examples [Hanneke '07, Friedman '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 $O(d^{3/2} \log 1/\epsilon)$.

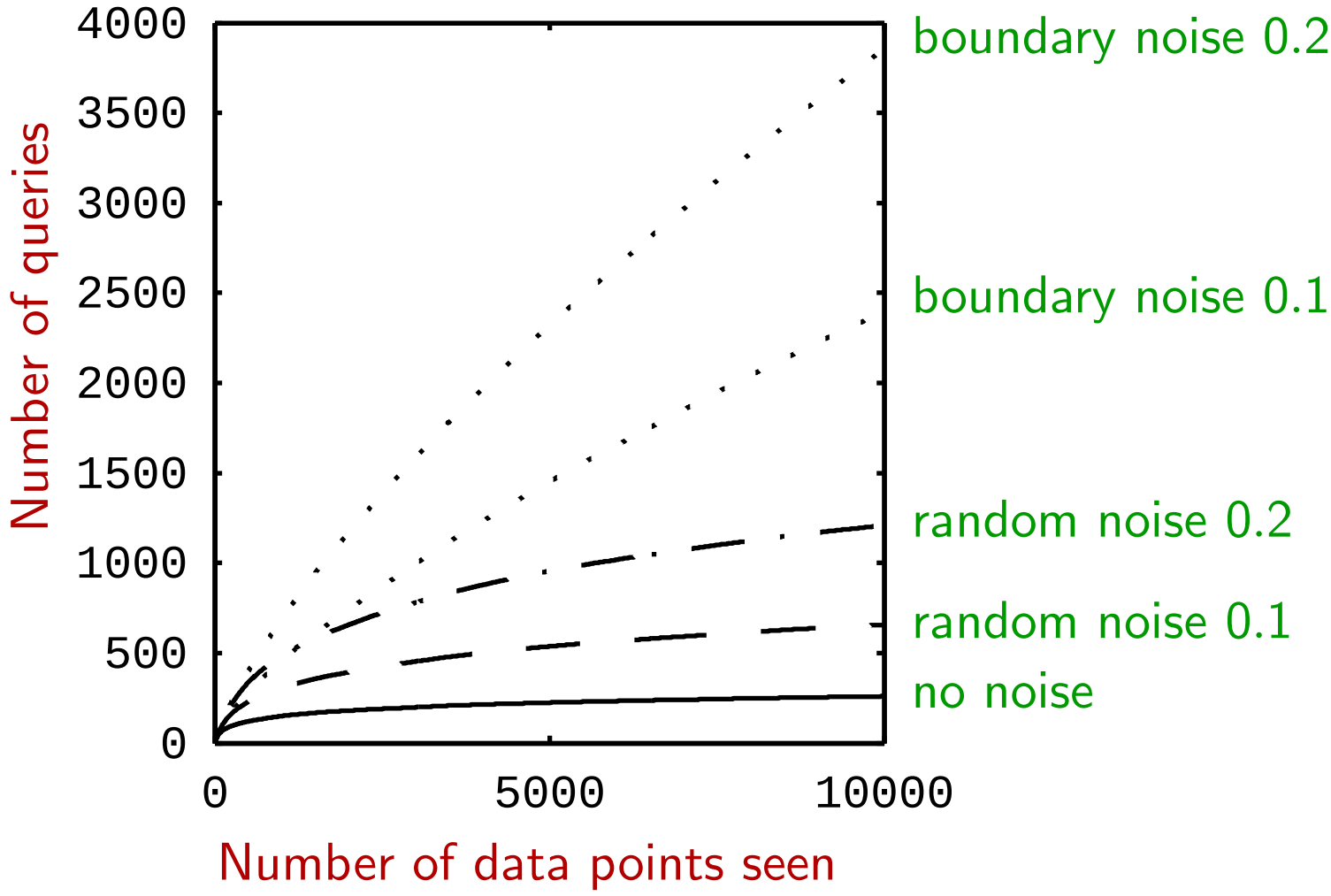
- Linear separators in $\mathbb{R}^d$, smooth data density bounded away from zero.

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

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

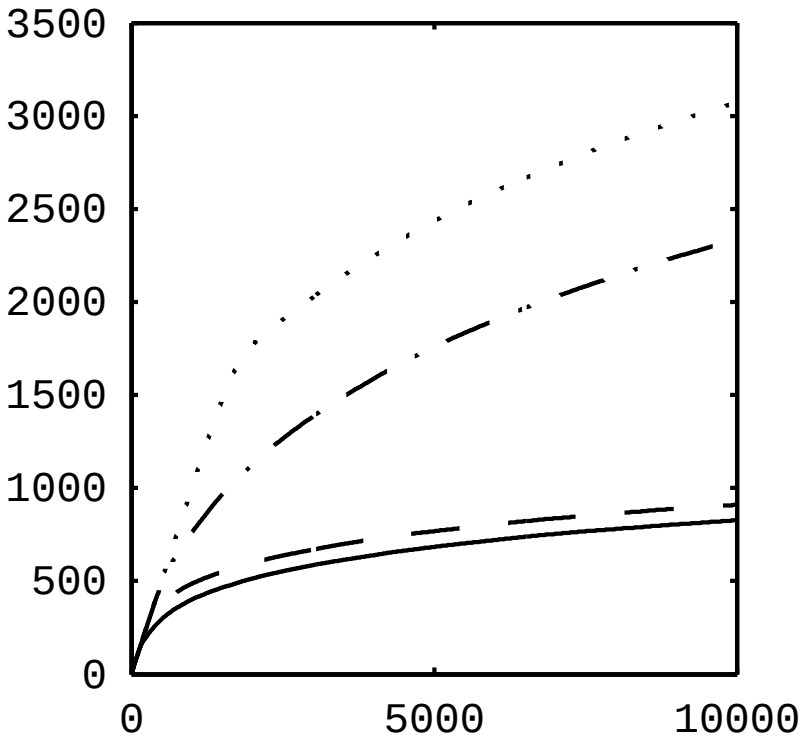
Example: thresholds on the line

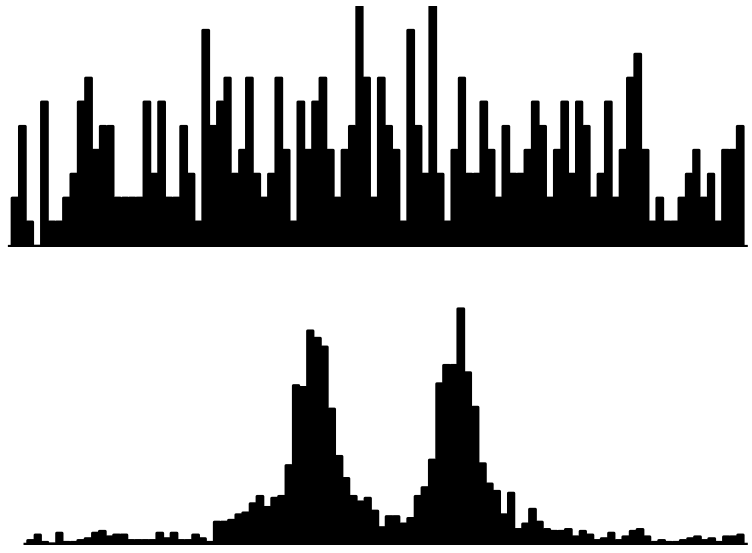
Data on $[0, 1]$. Target: threshold at 0.5.



Example: intervals on the line

Data on $[0, 1]$. Target: interval $[0.4, 0.6]$.





Overall distribution of queries
top: early on
bottom: later

Problems in scaling up to higher dimension

1. Loose bounds.
   These algorithms use generalization bounds to define the current version space.
2. Intractability of minimizing $0-1$ loss.
   Move to convex surrogate loss functions [BDL '09].

Three ways to manage sampling bias

1. Label everything.
2. Use importance weighting.
3. Explicitly manage sampling regions.

Importance weighting [Beygelzimer-D-Langford '09]

Standard classification setup with loss functions:

- Input space $\mathcal{X}$, label space $\mathcal{Y}$
- Hypotheses H map $\mathcal{X} \to \mathcal{Z}$
- Loss function $\ell : \mathcal{Z} \times \mathcal{Y} \to \mathbb{R}$
- Want $h \in H$ minimizing $L(h) = \mathbb{E}_{(X,Y)\sim\mathbb{P}}\ell(h(X), Y)$.

eg. $\mathcal{Y} = \{-1, 1\}$, $\mathcal{Z} = \mathbb{R}$, $\ell(z, y) = \ln(1 + e^{-yz})$.

Importance weighting [Beygelzimer-D-Langford '09]

Standard classification setup with loss functions:

- Input space $\mathcal{X}$, label space $\mathcal{Y}$
- Hypotheses H map $\mathcal{X} \to \mathcal{Z}$
- Loss function $\ell : \mathcal{Z} \times \mathcal{Y} \to \mathbb{R}$
- Want $h \in H$ minimizing $L(h) = \mathbb{E}_{(X,Y)\sim\mathbb{P}}\ell(h(X), Y)$.

eg. $\mathcal{Y} = \{-1, 1\}$, $\mathcal{Z} = \mathbb{R}$, $\ell(z, y) = \ln(1 + e^{-yz})$.

Importance-weighted active learning boilerplate:

1. Initialize $S = \emptyset$.
2. For $t = 1, 2, \ldots, T$:
   - Receive a new point x_t.
   - Choose a probability p_t.
   - With probability p_t: query label y_t and add $(x_t, y_t, 1/p_t)$ to S.
3. Return classifier $h \in H$ minimizing the weighted empirical loss

$$L_T(h) = \sum_{(x,y,w)\in S} w\,\ell(h(x), y).$$

Consistency of importance-weighted active learning

1. Initialize $S = \emptyset$.
2. For $t = 1, 2, \ldots, T$:
   - Receive a new point x_t.
   - Choose a probability p_t.
   - With probability p_t: query label y_t and add $(x_t, y_t, 1/p_t)$ to S.
3. Return the classifier $h \in H$ minimizing the weighted empirical loss $L_T(h) = \sum_{(x,y,w)\in S} w\,\ell(h(x), y)$.

Define $Q_t = 1(y_t \text{ is queried})$. Then $\mathbb{E}[Q_t|p_t] = p_t$, and

$$\mathbb{E}[L_T(h)] \;=\; \mathbb{E}\left[\sum_{t=1}^{T} \frac{Q_t}{p_t}\ell(h(x_t), y_t)\right] \;=\; \mathbb{E}_{(X,Y)\sim\mathbb{P}}\ell(h(X), Y) \;=\; L(h).$$

Consistency of importance-weighted active learning

1. Initialize $S = \emptyset$.
2. For $t = 1, 2, \ldots, T$:
   - Receive a new point x_t.
   - Choose a probability p_t.
   - With probability p_t: query label y_t and add $(x_t, y_t, 1/p_t)$ to S.
3. Return the classifier $h \in H$ minimizing the weighted empirical loss $L_T(h) = \sum_{(x,y,w)\in S} w\,\ell(h(x), y)$.

Define $Q_t = 1(y_t \text{ is queried})$. Then $\mathbb{E}[Q_t|p_t] = p_t$, and

$$\mathbb{E}[L_T(h)] \;=\; \mathbb{E}\left[\sum_{t=1}^{T} \frac{Q_t}{p_t}\ell(h(x_t), y_t)\right] \;=\; \mathbb{E}_{(X,Y)\sim\mathbb{P}}\ell(h(X), Y) \;=\; L(h).$$

If H is finite and all $p_t \geq p_{min}$, then with probability at least $1 - \delta$,

$$\max_{h\in H} |L_T(h) - L(h)| \;\leq\; \sqrt{\frac{2\ln|H|/\delta}{T p_{min}}}.$$

One way to pick the sampling probabilities

- L_t^* = minimum empirical loss at time t, achieved by $h_t \in H$
- H_t = hypotheses $h \in H$ such that for all $t' < t$,

$$L_{t'}(h) \leq L_{t'}^* + \Delta_{t'}$$

where $\Delta_t \sim \sqrt{(\log |H|)/t}$ is from a generalization bound.

One way to pick the sampling probabilities

- $L_t^* =$ minimum empirical loss at time t, achieved by $h_t \in H$
- $H_t =$ hypotheses $h \in H$ such that for all $t' < t$,

$$L_{t'}(h) \leq L_{t'}^* + \Delta_{t'}$$

where $\Delta_t \sim \sqrt{(\log |H|)/t}$ is from a generalization bound.

Upon seeing example x_t, set the probability of sampling its label to

$$p_t \propto \max_{f,g \in H_t} \max_{y \in \mathcal{Y}} \ell(f(x_t), y) - \ell(g(x_t), y).$$

Linear separators, convex loss: this is a convex optimization.

One way to pick the sampling probabilities

- $L_t^* =$ minimum empirical loss at time t, achieved by $h_t \in H$
- $H_t =$ hypotheses $h \in H$ such that for all $t' < t$,

$$L_{t'}(h) \leq L_{t'}^* + \Delta_{t'}$$

where $\Delta_t \sim \sqrt{(\log |H|)/t}$ is from a generalization bound.

Upon seeing example x_t, set the probability of sampling its label to

$$p_t \propto \max_{f,g \in H_t} \max_{y \in \mathcal{Y}} \ell(f(x_t), y) - \ell(g(x_t), y).$$

Linear separators, convex loss: this is a convex optimization.

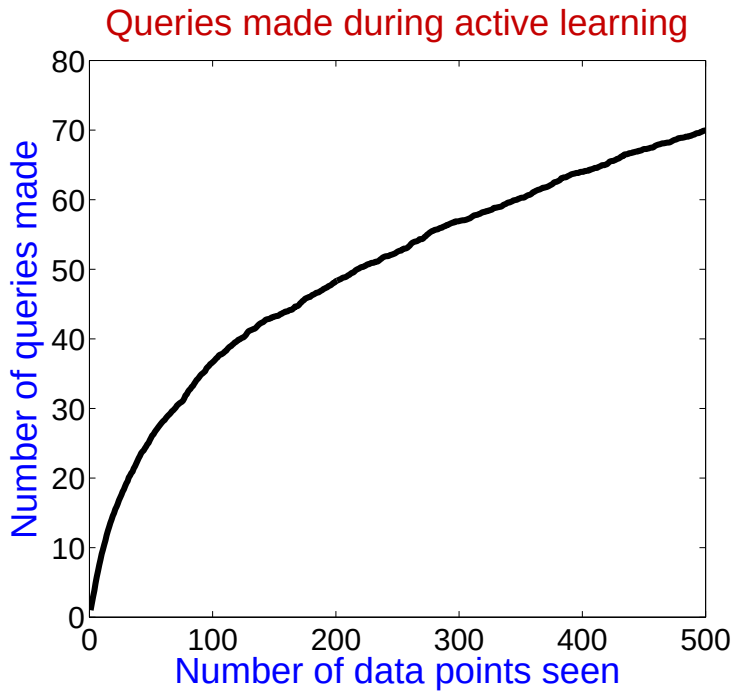
With probability at least $1 - \delta$, final hypothesis h_T satisfies

$$L(h_T) - L(h^*) \leq 2\Delta_{T-1}.$$

Label complexity from disagreement-region considerations.

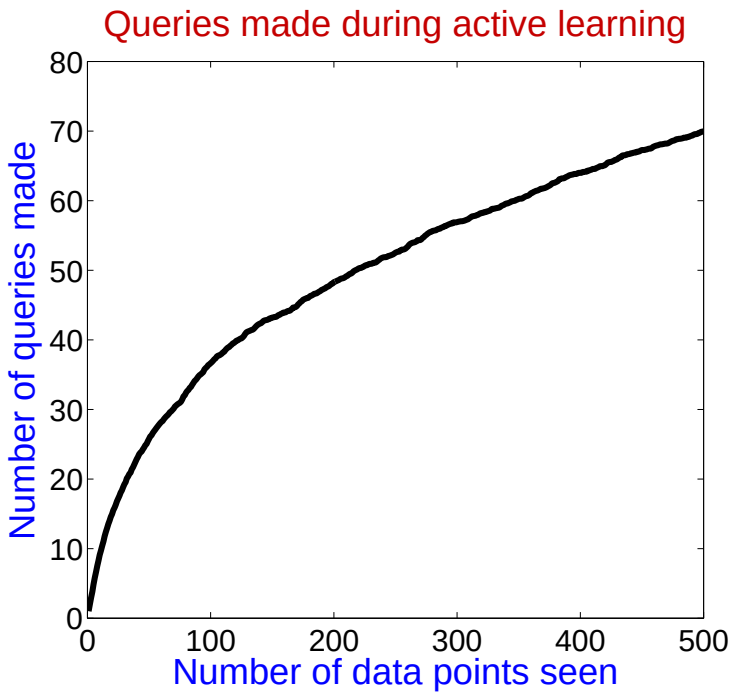
Example: linear separators in $\mathbb{R}^{10}$

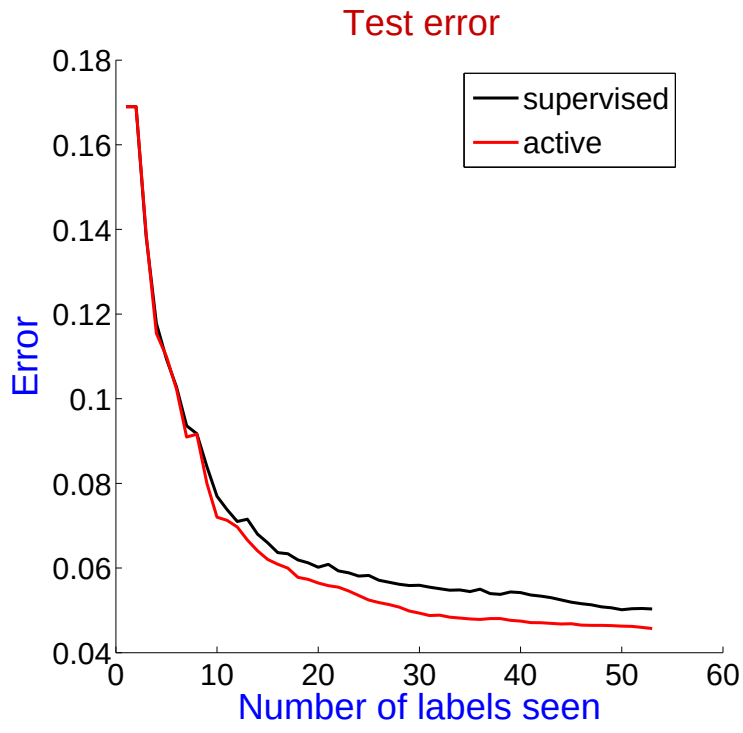
Two Gaussian classes, with 5% overlap.



Example: linear separators in $\mathbb{R}^{10}$

Two Gaussian classes, with 5% overlap.





Improving efficiency [Beygelzimer-Hsu-Langford-Zhang '10, Karampatziakis-L '11]

1. An alternate sampling scheme for linear classifiers.

   At time t, compute hypotheses h^*, h^-:

$$\begin{aligned} h^+ &= \arg\min L_t(h) \text{ subject to } h(x_t) = +1. \\ h^- &= \arg\min L_t(h) \text{ subject to } h(x_t) = -1. \end{aligned}$$

   Then set the sampling probability

$$p_t \approx |L_t(h^+) - L_t(h^-)| \cdot \frac{\log t}{t}.$$

   Label complexity bounds can be given for $0-1$ loss.

Improving efficiency [Beygelzimer-Hsu-Langford-Zhang '10, Karampatziakis-L '11]

1. An alternate sampling scheme for linear classifiers.

   At time t, compute hypotheses h^*, h^-:

   $$\begin{aligned} h^+ &= \arg\min L_t(h) \text{ subject to } h(x_t) = +1. \\ h^- &= \arg\min L_t(h) \text{ subject to } h(x_t) = -1. \end{aligned}$$

   Then set the sampling probability

   $$p_t \approx |L_t(h^+) - L_t(h^-)| \cdot \frac{\log t}{t}.$$

   Label complexity bounds can be given for $0-1$ loss.

2. Next, make this efficient.

   Approximate h^+, h^- by two separate gradient steps from h_t.

Big open problem

More aggressive querying strategies are needed...

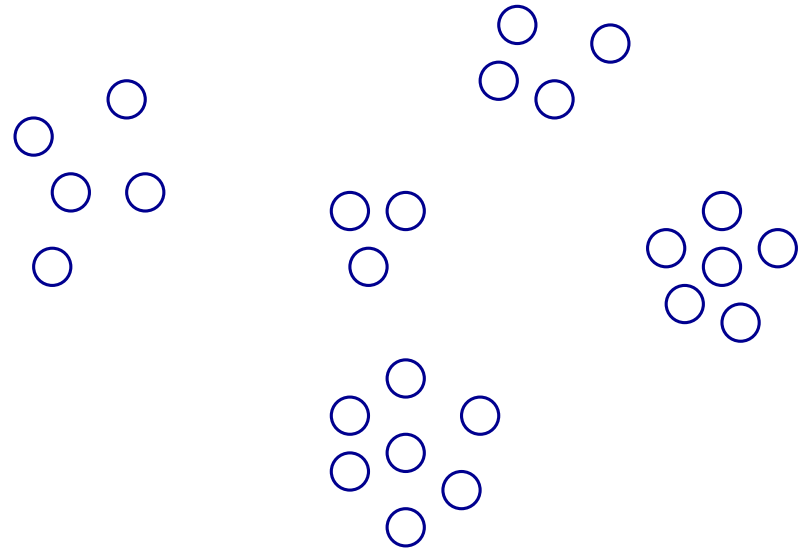
	Separable data	General (nonseparable) data
Aggressive	Query by committee (Freund, Seung, Shamir, Tishby, 97) Splitting index (D, 05)	???
Mellow	Generic mellow learner (Cohn, Atlas, Ladner, 91)	A^2 algorithm (Balcan, Beygelzimer, Langford, 06) Disagreement coefficient (Hanneke, 07) Reduction to supervised (D, Hsu, Monteleoni, 07) Importance-weighted approach (Beygelzimer, D, Langford, 09)

Three ways to manage sampling bias

1. Label everything.
2. Use importance weighting.
3. Explicitly manage sampling regions.

Exploiting cluster structure in data

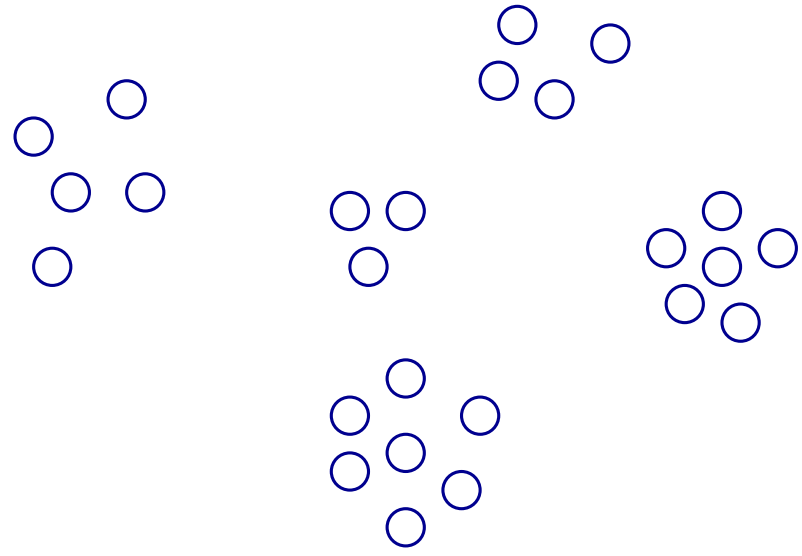
Suppose the unlabeled data looks like this.



Then perhaps we just need five labels.

Exploiting cluster structure in data

Suppose the unlabeled data looks like this.



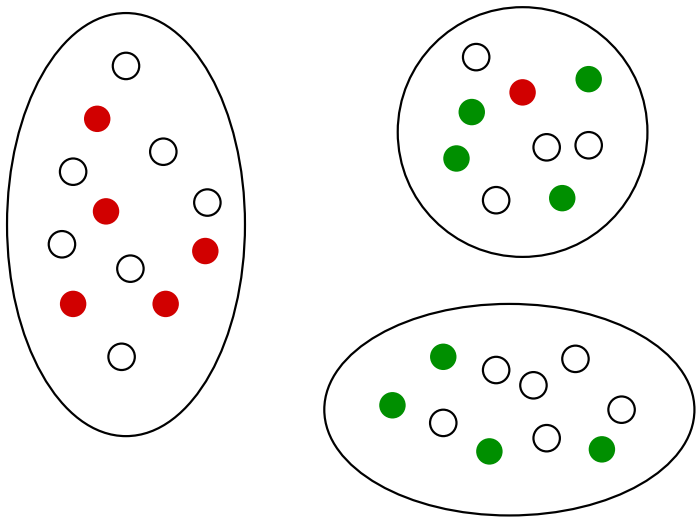
Then perhaps we just need five labels.

Challenges: In general, the cluster structure (i) is not so clearly defined and (ii) exists at many levels of granularity. And (iii) the clusters may not be pure in their labels.

Exploiting cluster structure in data [D-Hsu '08]

Basic primitive:

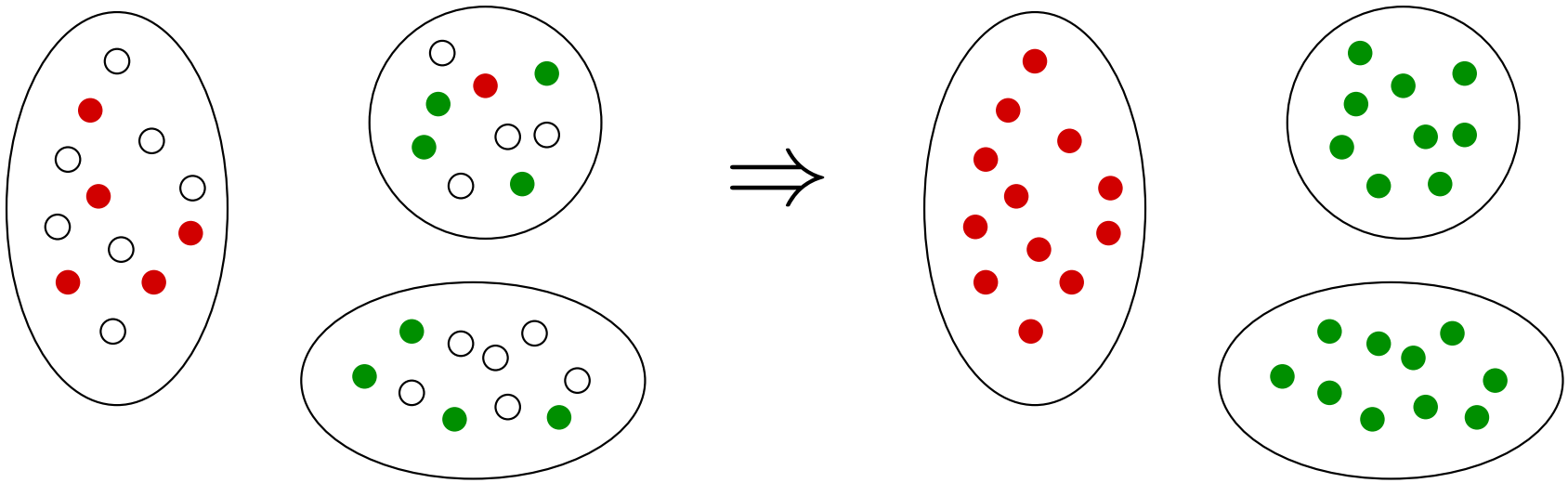
- Find a clustering of the data
- Sample a few *randomly-chosen* points in each cluster
- Assign each cluster its majority label
- Now use this fully labeled data set to build a classifier



Exploiting cluster structure in data [D-Hsu '08]

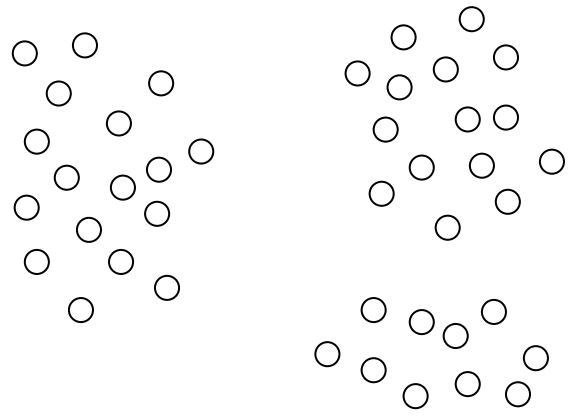
Basic primitive:

- 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



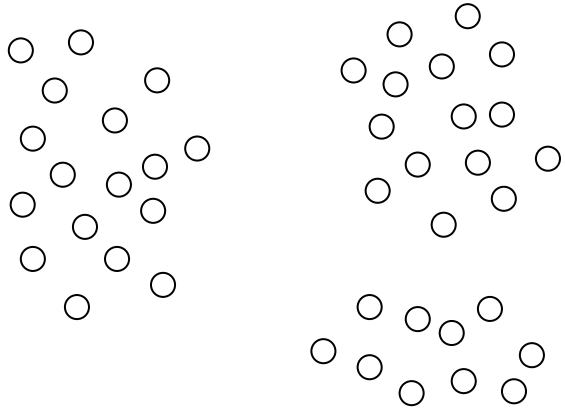
Finding the right granularity

Unlabeled data

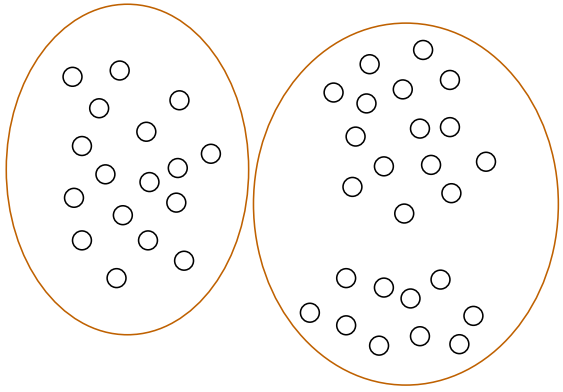


Finding the right granularity

Unlabeled data

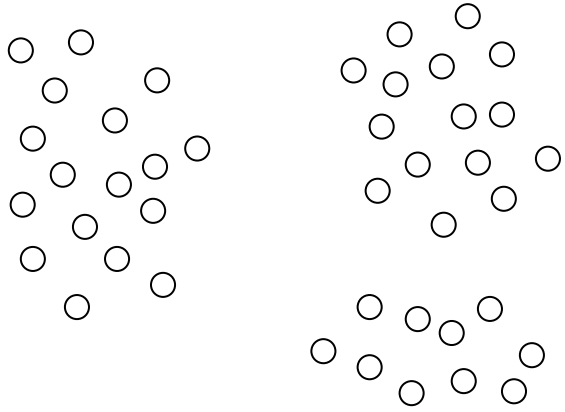


Find a clustering

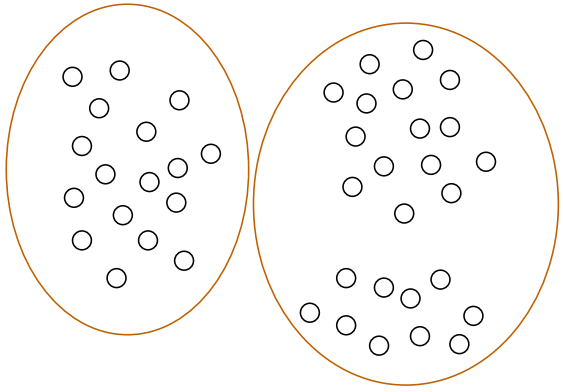


Finding the right granularity

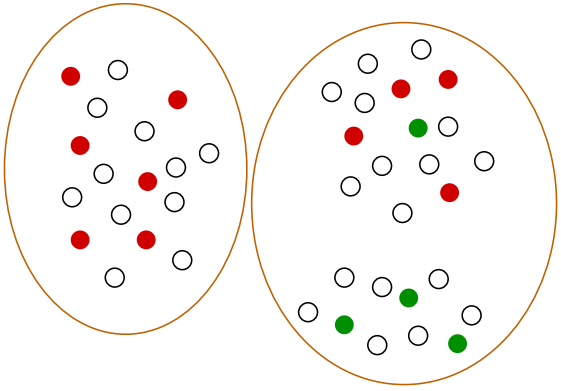
Unlabeled data



Find a clustering



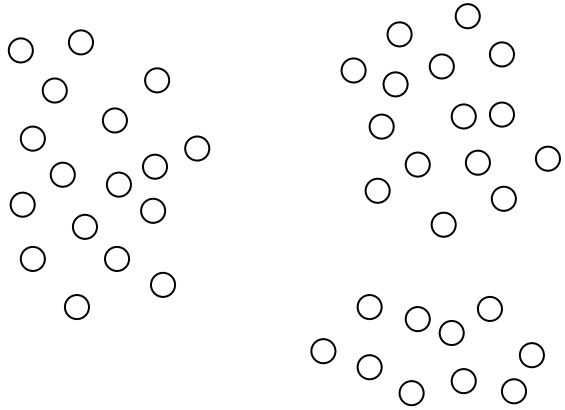
Ask for some labels



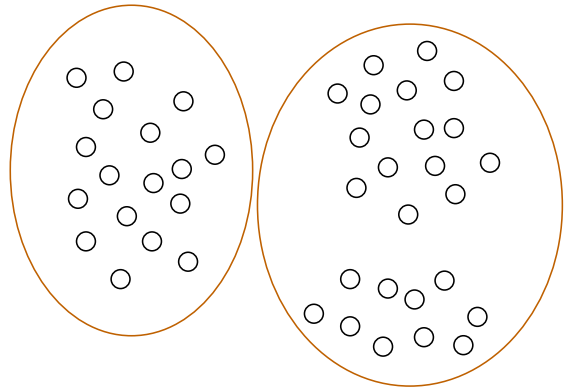
(random sampling within clusters)

Finding the right granularity

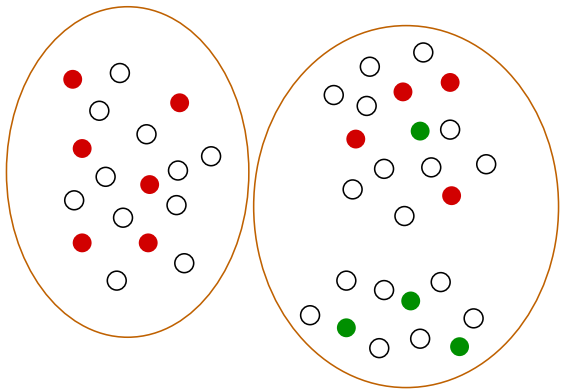
Unlabeled data



Find a clustering



Ask for some labels

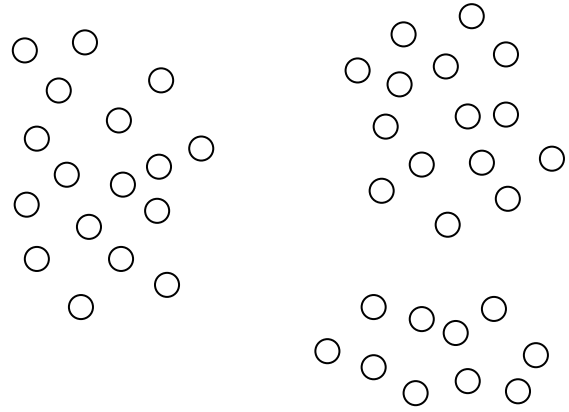


(random sampling within clusters)

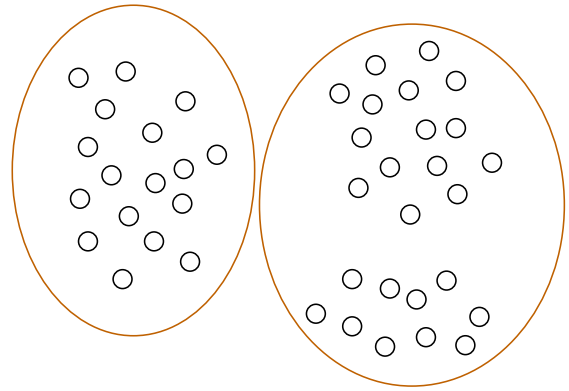
Now what?

Finding the right granularity

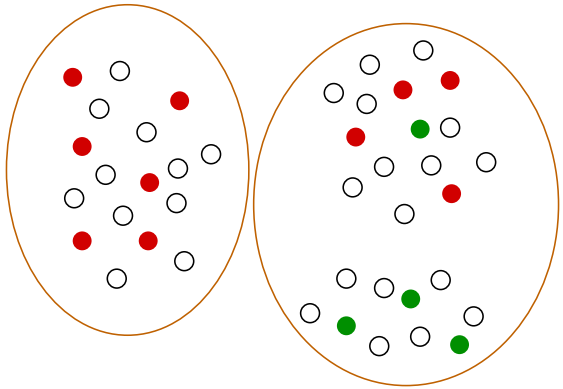
Unlabeled data



Find a clustering



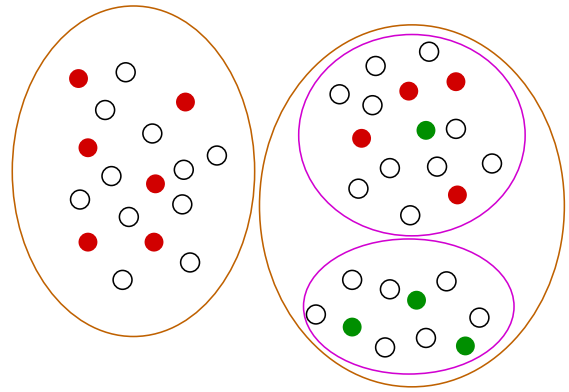
Ask for some labels



(random sampling within clusters)

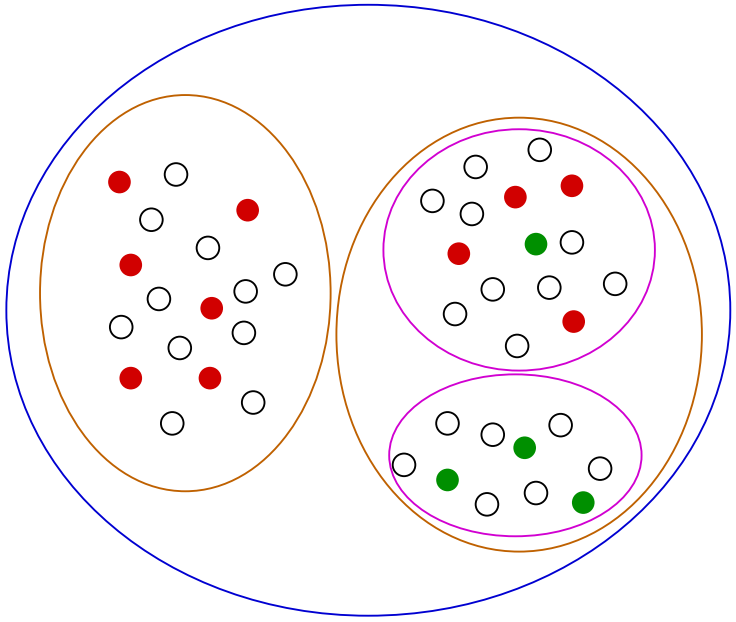
Now what?

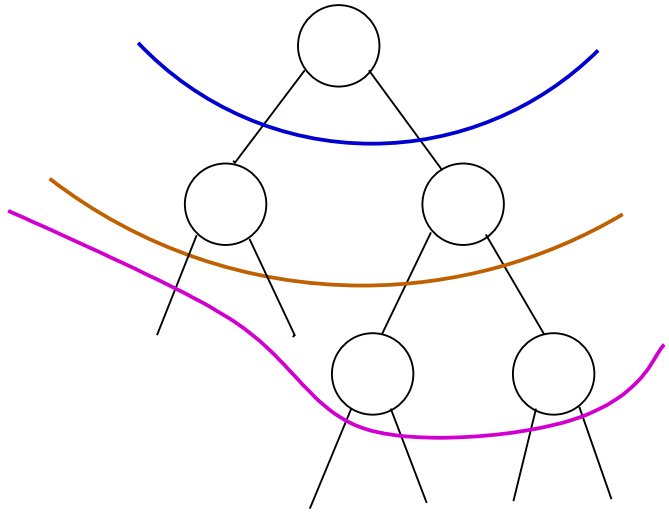
Refine the clustering



Queried points are also randomly distributed within the new clusters.

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'

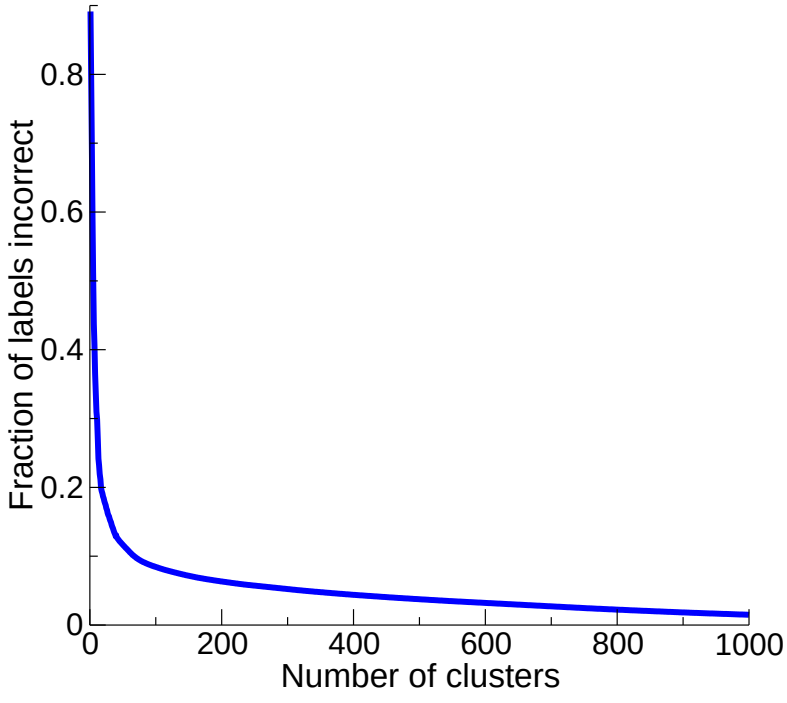
for each v in P: assign each leaf in T_v the label $L(v)$

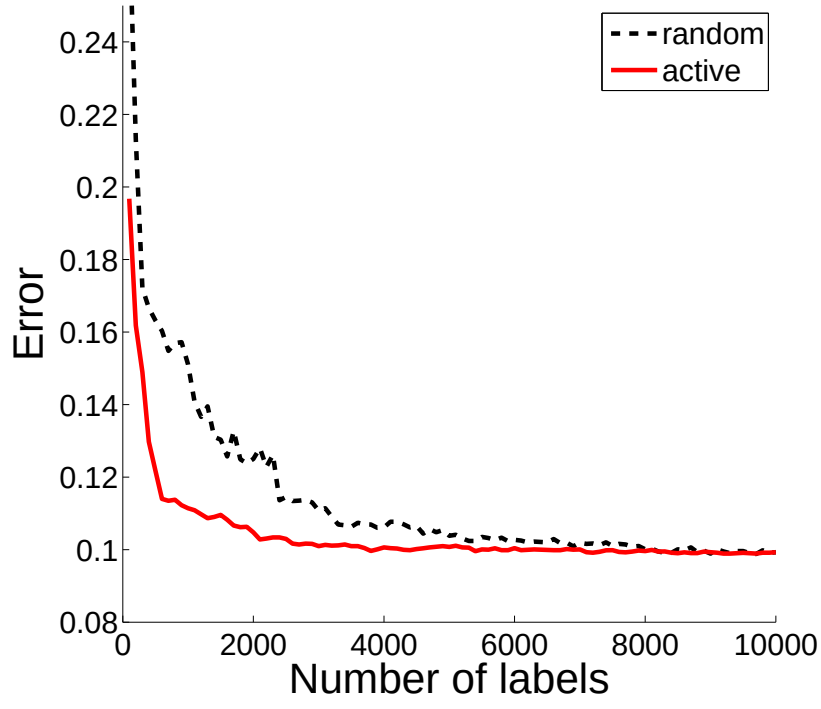
return the resulting fully labeled data set

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

Example: MNIST digits

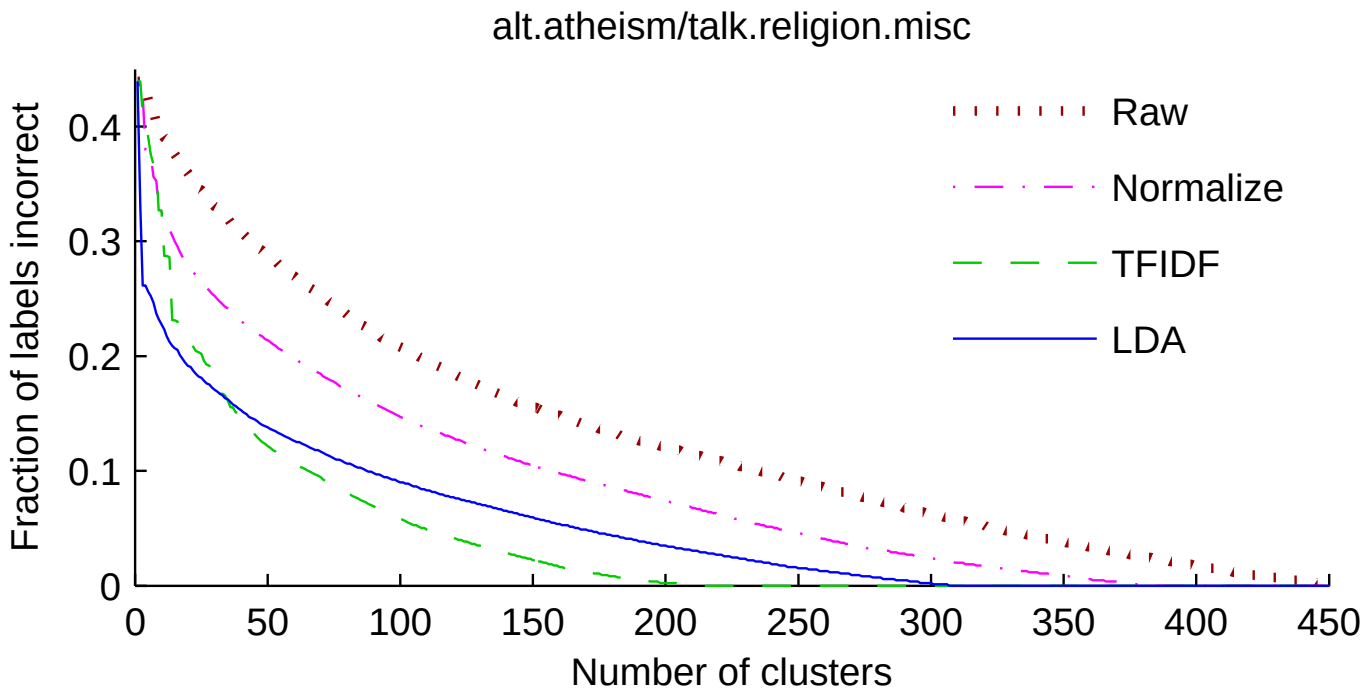
Hierarchy built using Ward's agglomerative clustering (k-means cost function) with Euclidean distance.





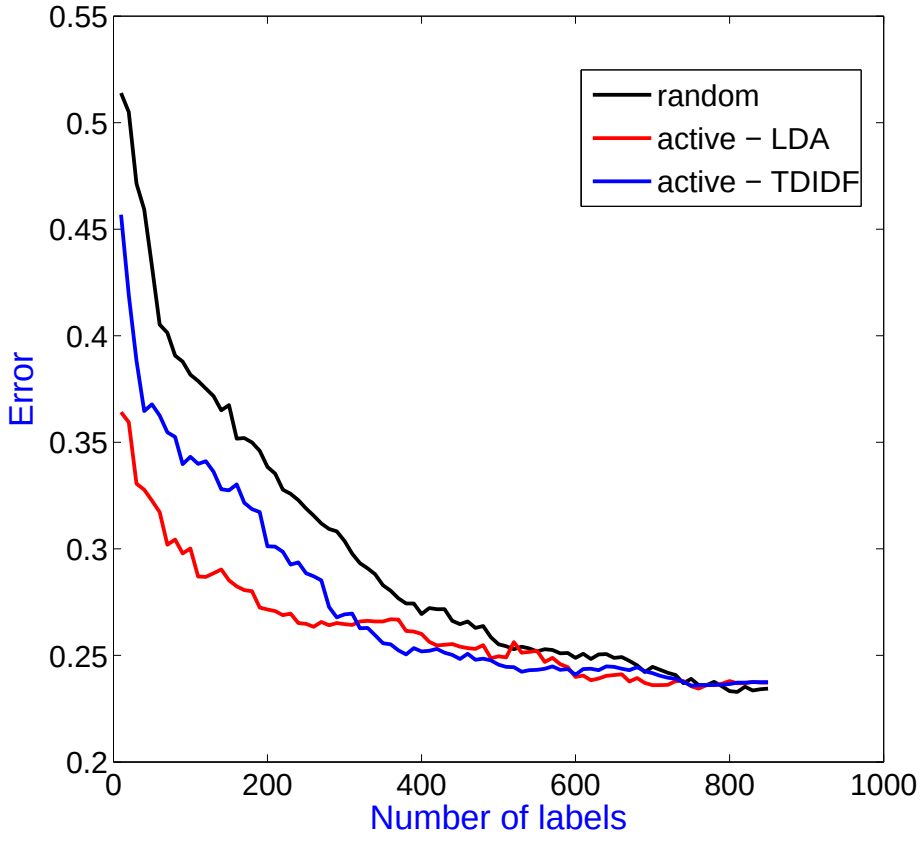
Example: 20 Newsgroups

Hierarchy built using Ward's agglomerative clustering with Euclidean distance (all but LDA) or KL divergence (LDA).



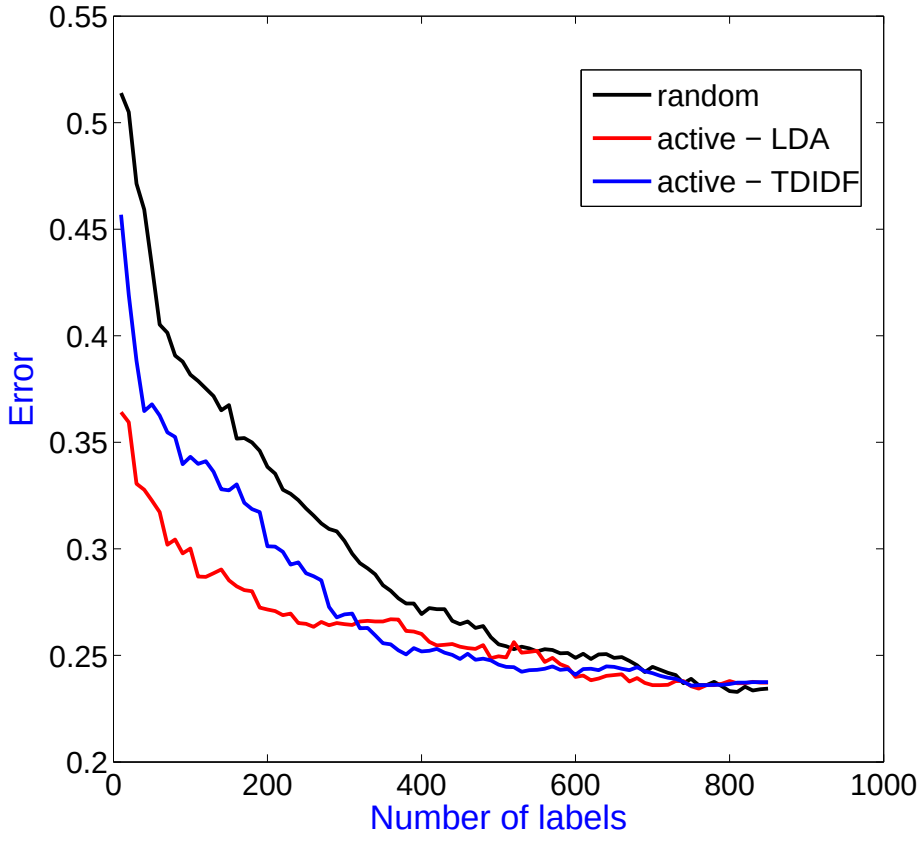
Example: 20 Newsgroups

alt.atheism vs talk.religion.misc



Example: 20 Newsgroups

alt.atheism vs talk.religion.misc



Open problem: Hierarchical sampling is a promising aggressive strategy – how can its label complexity be characterized? When is it most effective?

Thanks

To my co-authors Alina Beygelzimer, Daniel Hsu, John Langford, and Claire Monteleoni.

And to the National Science Foundation for support under grants IIS-0347646, IIS-0713540, IIS-0812598, and CPS-0932403.