Learning Theory Continued…

Machine Learning – CSE446

Carlos Guestrin

University of Washington

May 13, 2013

A simple setting…

- Classification
  - N data points iid
  - **Finite** number of possible hypothesis (e.g., dec. trees of depth d)
- A learner finds a hypothesis *h* that is **consistent** with training data
  - Gets zero error in training – $error_{train}(h) = 0$
- What is the probability that *h* has more than ε true error?
  - $error_{true}(h) \geq \varepsilon$ ← large for $\varepsilon > 0$ given

Generalization error in finite hypothesis spaces [Haussler '88]

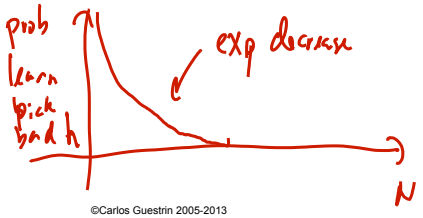
- ***Theorem***: Hypothesis space *H* finite, dataset *D* with *N* i.i.d. samples, $0 < \varepsilon < 1$: for any learned hypothesis *h* that is consistent on the training data:

$$P(error_{true}(h) > \epsilon) \leq |H|e^{-N\epsilon}$$

$\leq |H|(1-\varepsilon)^N \leq |H|(e^{-\varepsilon})^N = |H|e^{-N\varepsilon}$

for $0 \leq \varepsilon \leq 1$

$1-\varepsilon \leq e^{-\varepsilon}$



Limitations of Haussler '88 bound

$P(error_{true}(h) > \epsilon) \leq |H|e^{-N\epsilon}$

- Consistent classifier

↑ $error_{train}(h) = 0$

highly unrealist, and bad WRT overfitting

- Size of hypothesis space

$|n|H|$, bad is H is continuous (infinite)

or H is very very large

What if our classifier does not have zero error on the training data?

- A learner with zero training errors may make mistakes in test set
- What about a learner with $error_{train}(h)$ in training set?

if $error_{train}(h) > 0$

what about $error_{true}(h)$?

in Logistic Regression, there are infinitely many h, parameterized by w

Generalization bound for |H| hypothesis

- ***Theorem***: Hypothesis space *H* finite, dataset *D* with *N* i.i.d. samples, $0 < \varepsilon < 1$: for any learned hypothesis *h*:

does?

$$P(error_{true}(h_i) - error_{train}(h_i) > \epsilon) \leq e^{-2N\epsilon^2} \leq \delta$$

hold $\forall h_i$

$P(error_{true}(h) - error_{train}(h) > \varepsilon) \leq |H| e^{-2N\varepsilon^2}$

$\varepsilon \geq \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2N}}$

with prob. at least $1-\delta$

$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2N}}$

PAC bound and Bias-Variance tradeoff

$$P\left(error_{true}(h) - error_{train}(h) > \epsilon\right) \leq e^{-2N\epsilon^2}$$

or, after moving some terms around, with probability at least 1-δ:

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln|H| + \ln\frac{1}{\delta}}{2N}}$$

Trees of depth 1,000,000	"Complex" hypothesis space	Small
Trees of depth 4	"Simple" hypothesis space	larger "bias"

|H| large => large

Smaller, because |H| smaller "variance"

- **Important: PAC bound holds for all *h*, but doesn't guarantee that algorithm finds best *h*!!!**

What about the size of the hypothesis space?

$$N \geq \frac{\ln|H| + \ln\frac{1}{\delta}}{2\epsilon^2}$$

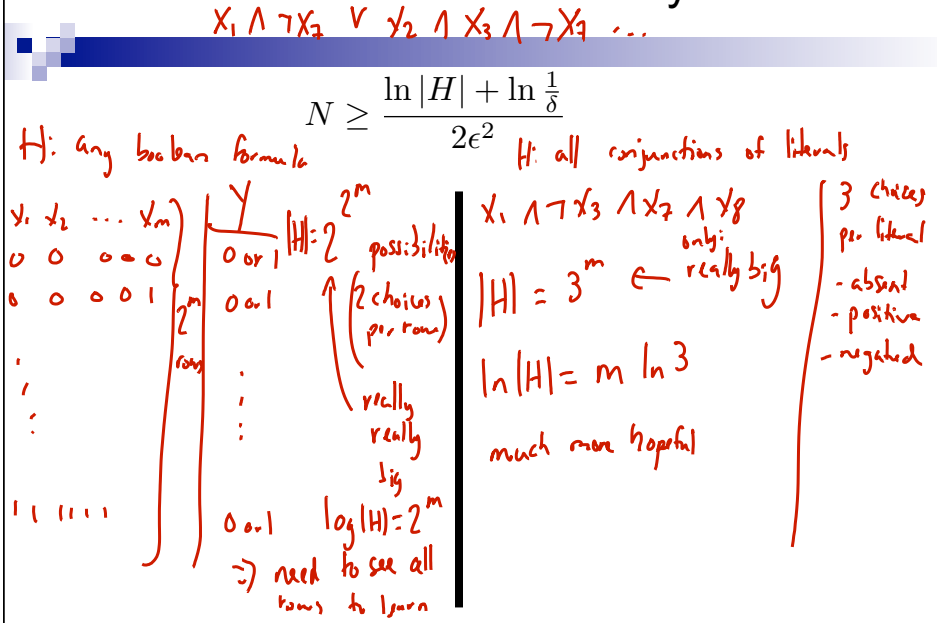
- How large is the hypothesis space?

How big is |H|?

really big Versus really really big

Boolean formulas with *m* binary features

$$N \geq \frac{\ln |H| + \ln \frac{1}{\delta}}{2\epsilon^2}$$



Number of decision trees of depth k

$$N \geq \frac{\ln |H| + \ln \frac{1}{\delta}}{2\epsilon^2}$$

Recursive solution

Given *m* attributes

H_k = Number of decision trees of depth k

H_0 =2

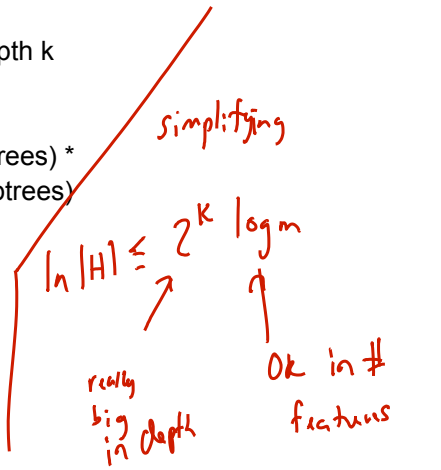
H_{k+1} = (#choices of root attribute) *
(# possible left subtrees) *
(# possible right subtrees)
= m * H_k * H_k

Write $L_k = \log_2 H_k$

$L_0 = 1$

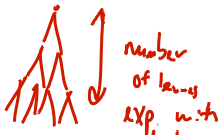
$L_{k+1} = \log_2 m + 2L_k$

So $L_k = (2^k-1)(1+\log_2 m) +1$



PAC bound for decision trees of depth k

$$N \geq \frac{2^k \log m + \ln \frac{1}{\delta}}{\epsilon^2}$$



- Bad!!!
  - Number of points is exponential in depth!

- But, for *N* data points, decision tree can't get too big…

no reason to expand to more than N leaves

Number of leaves never more than number data points

Number of Decision Trees with k Leaves

- Number of decision trees of depth k is really really big:
  - ln |H| is about $2^k \log m$

← extra flexibility U unnecessary in the theory

- Decision trees with up to k leaves:
  - |H| is about $m^k k^{2k}$ ← only really big
    - A very loose bound

$\ln |H| \leq k \ln m + 2k \ln k$ ←

much better!! U

PAC bound for decision trees with k leaves – Bias-Variance revisited

$\ln |H_{\text{DTs k leaves}}| \leq 2k(\ln m + \ln k)$

my bound $error_{true}(h) \leq 1$

$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2N}}$

want to be small

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{2k(\ln m + \ln k) + \ln \frac{1}{\delta}}{2N}}$$

$2N \leftarrow$ # datapoints

real number of leaves DT	"bias"	"variance"
$K \approx N$	goes to zero	LARGE greater than 1
$K << N$	Potentially larger	Small

What did we learn from decision trees?

- Bias-Variance tradeoff formalized

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{2k(\ln m + \ln k) + \ln \frac{1}{\delta}}{2N}}$$

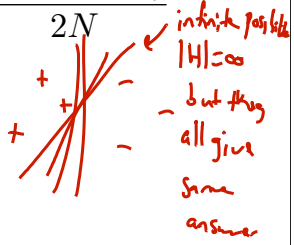
- Moral of the story:

  Complexity of learning not measured in terms of size hypothesis space, but in maximum *number of points* that allows consistent classification
  - Complexity N – no bias, lots of variance
  - Lower than N – some bias, less variance

↑ flexibility of the class

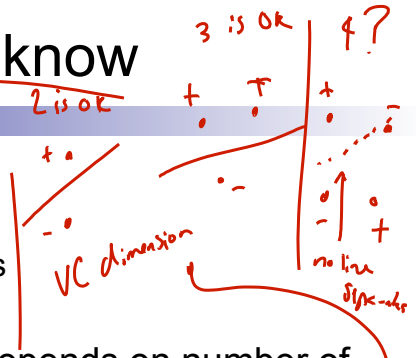
What about continuous hypothesis spaces?

$$error_{true}(h) \leq error_{train}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2N}}$$



- Continuous hypothesis space:
  - |H| = ∞
  - Infinite variance???
- **As with decision trees, only care about the maximum number of points that can be classified exactly!**
  - **Called VC dimension… see readings for details**

What you need to know



- Finite hypothesis space
  - Derive results
  - Counting number of hypothesis
  - Mistakes on Training data
- Complexity of the classifier depends on number of points that can be classified exactly
  - Finite case – decision trees
  - Infinite case – VC dimension
- Bias-Variance tradeoff in learning theory
- Remember: will your algorithm find best classifier?

Clustering
K-means

Machine Learning – CSE446

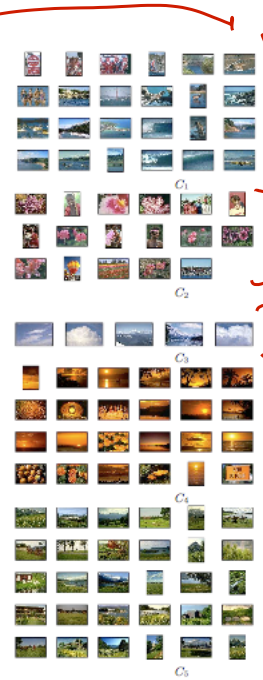
Carlos Guestrin

University of Washington

May 13, 2013

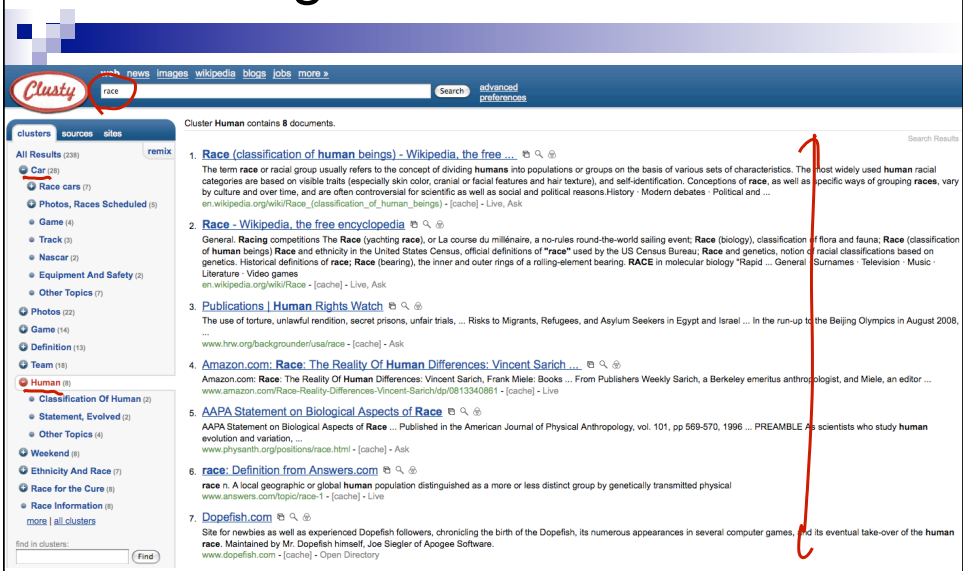
Clustering images

Set of Images

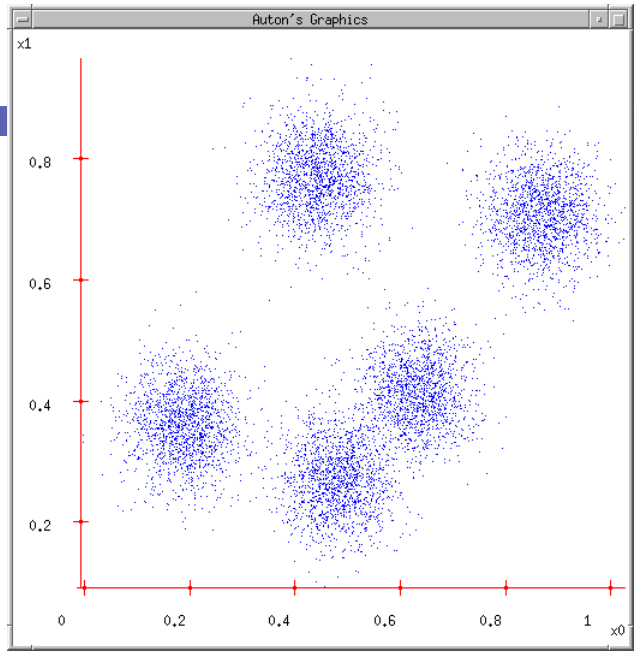


[Goldberger et al.]

Clustering web search results

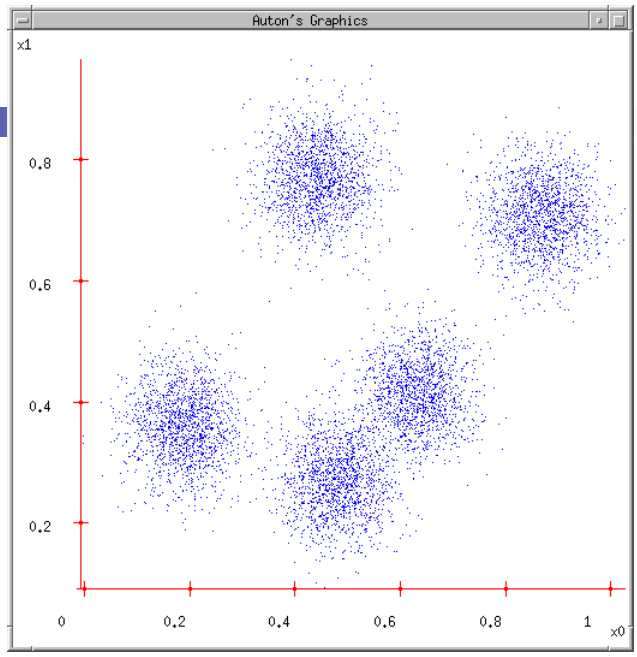


Some Data



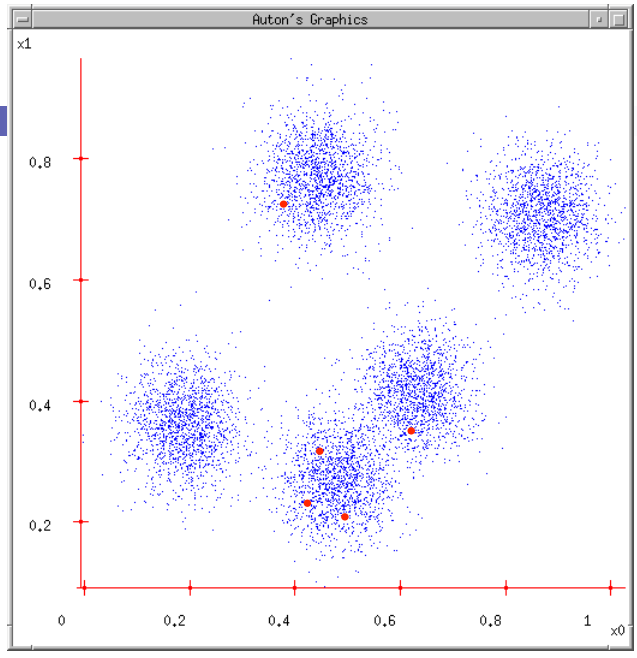
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



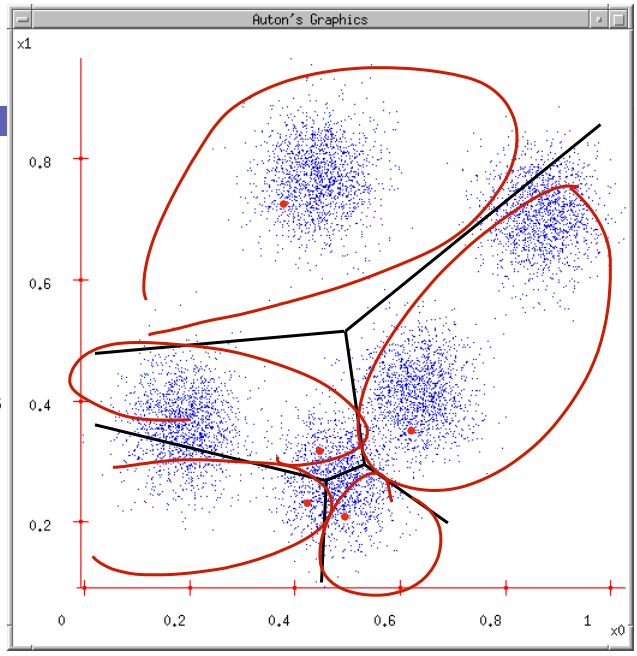
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



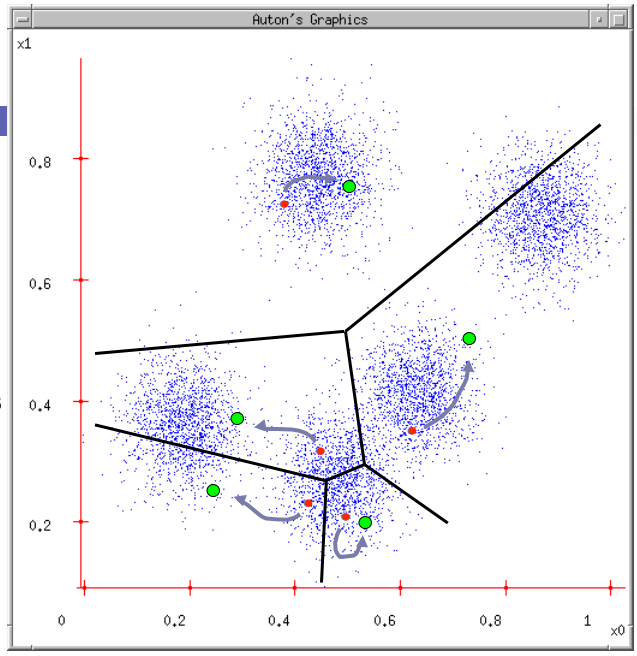
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



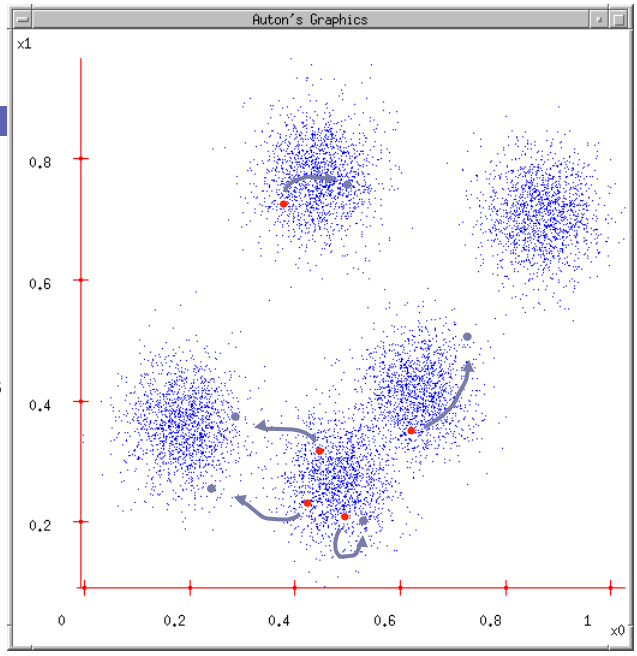
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



K-means

- Randomly initialize *k* centers or smartly
  - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

converge when nothing moves (no point changes its cluster)

- **Classify**: Assign each point $j \in \{1, \ldots n\}$ to nearest center:
  - $C^{(t)}(j) \leftarrow \arg\min_i ||\mu_i - x_j||^2$

μ_i is cluster center; x_j data point

- **Recenter**: μ_i becomes centroid of its point:
  - $\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} ||\mu - x_j||^2 \leftarrow \mu_i = \frac{\sum_{j:C(j)=i} x_j}{\text{num of points assigned to cluster } i}$
  - Equivalent to $\mu_i \leftarrow$ average of its points!

What is K-means optimizing?

- Potential function F(μ,C) of centers μ and point allocations C:
  - $F(\mu, C) = \sum_{j=1}^{N} ||\mu_{C(j)} - x_j||^2$
- Optimal K-means:
  - $\min_{\mu} \min_{C} F(\mu, C)$

Does K-means converge??? Part 1

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix μ, optimize C

Does K-means converge??? Part 2

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

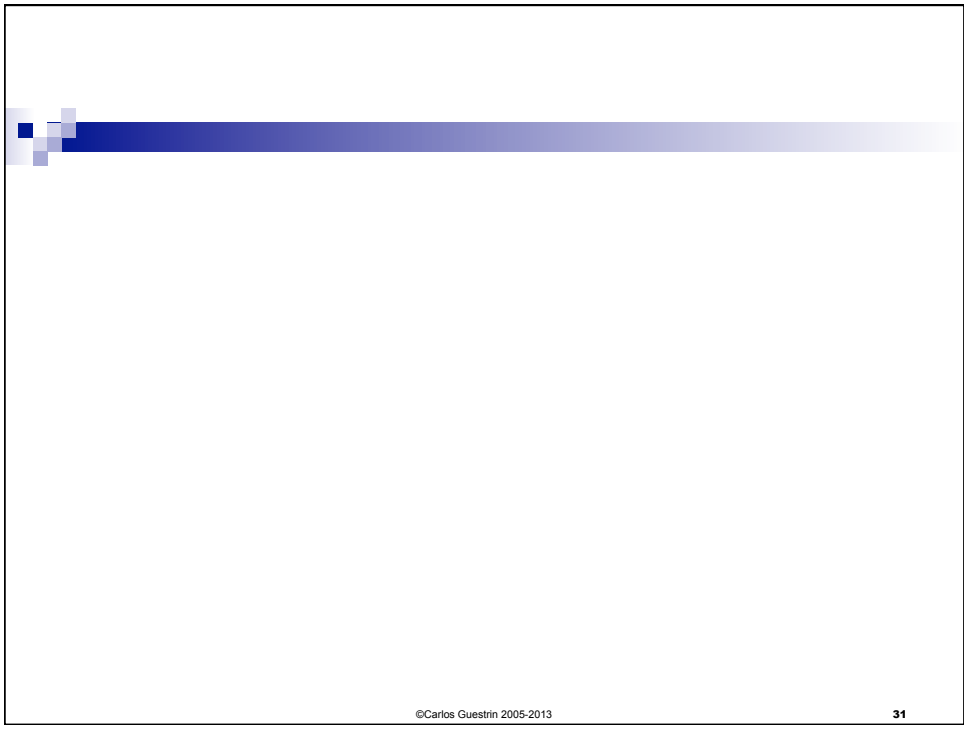
- Fix C, optimize $\boldsymbol{\mu}$

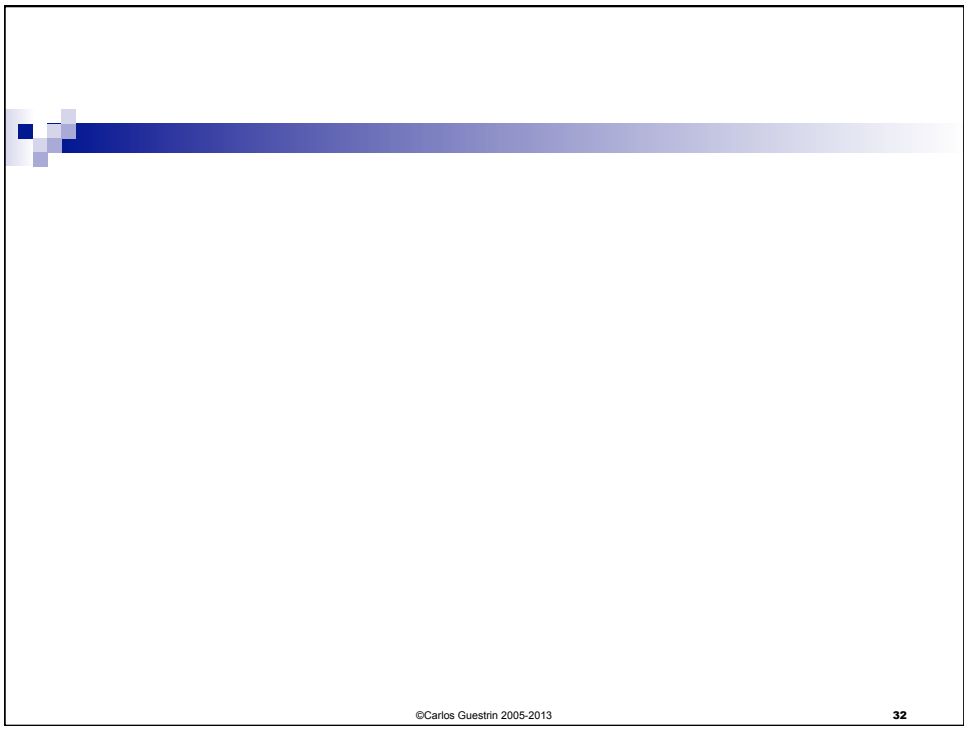
Coordinate descent algorithms

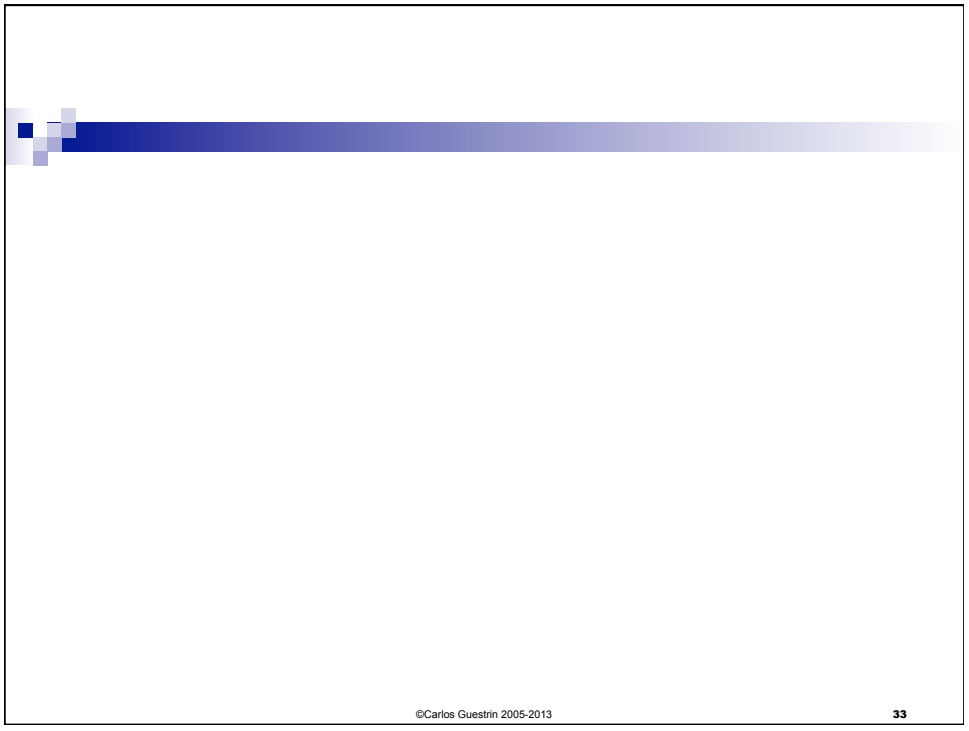
$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Want: $\min_a \min_b F(a,b)$
- Coordinate descent:
  - fix a, minimize b
  - fix b, minimize a
  - repeat
- Converges!!!
  - if F is bounded
  - to a (often good) local optimum
    - as we saw in applet (play with it!)
      - (For LASSO it converged to the optimum)

- K-means is a coordinate descent algorithm!







How many points can a linear boundary classify exactly? (1-D)

How many points can a linear boundary classify exactly? (2-D)

How many points can a linear boundary classify exactly? (d-D)

PAC bound using VC dimension

- **Number of training points that can be classified exactly is VC dimension!!!**
  - **Measures relevant size of hypothesis space, as with decision trees with k leaves**

$$\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{VC(H)\left(\ln \frac{2m}{VC(H)} + 1\right) + \ln \frac{4}{\delta}}{m}}$$

Shattering a set of points

Definition: a **dichotomy** of a set S is a partition of S into two disjoint subsets.

Definition: a set of instances S is **shattered** by hypothesis space H if and only if for every dichotomy of S there exists some hypothesis in H consistent with this dichotomy.

VC dimension

Definition: The **Vapnik-Chervonenkis dimension**, $VC(H)$, of hypothesis space H defined over instance space X is the size of the largest finite subset of X shattered by H. If arbitrarily large finite sets of X can be shattered by H, then $VC(H) \equiv \infty$.

PAC bound using VC dimension

- **Number of training points that can be classified exactly is VC dimension!!!**
  - **Measures relevant size of hypothesis space, as with decision trees with k leaves**
  - **Bound for infinite dimension hypothesis spaces:**

$$\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{VC(H)\left(\ln \frac{2m}{VC(H)} + 1\right) + \ln \frac{4}{\delta}}{m}}$$

Examples of VC dimension

$$\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{VC(H)\left(\ln\frac{2m}{VC(H)}+1\right)+\ln\frac{4}{\delta}}{m}}$$

- Linear classifiers:
  - VC(H) = d+1, for *d* features plus constant term *b*
- Neural networks
  - VC(H) = #parameters
  - Local minima means NNs will probably not find best parameters
- 1-Nearest neighbor?

Another VC dim. example - What can we shatter?

- What's the VC dim. of decision stumps in 2d?

Another VC dim. example - What can't we shatter?

- What's the VC dim. of decision stumps in 2d?

What you need to know

- Finite hypothesis space
  - Derive results
  - Counting number of hypothesis
  - Mistakes on Training data
- Complexity of the classifier depends on number of points that can be classified exactly
  - Finite case – decision trees
  - Infinite case – VC dimension
- Bias-Variance tradeoff in learning theory
- Remember: will your algorithm find best classifier?