A simple setting…

- Classification
  - N data points \( \text{iid} \)
  - 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 – \( \text{error}_{\text{train}}(h) = 0 \)

- What is the probability that \( h \) has more than \( \varepsilon \) true error?
  - \( \text{error}_{\text{true}}(h) \geq \varepsilon \) for \( \varepsilon > 0 \)
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:

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

Limitations of Haussler ‘88 bound

- Consistent classifier
  - $\text{Error}_{\text{train}}(h) = 0$
  - Highly unlearnable, and bad w.r.t. overfitting

- Size of hypothesis space
  - $|h|$, 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 $\text{error}_{\text{train}}(h)$ in training set?

In Logistic Regression, there are infinitely many $h$, parametrized 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 $h$:

$$P(\text{error}_{\text{true}}(h) - \text{error}_{\text{train}}(h) > \varepsilon) \leq e^{-2N\varepsilon^2}$$

$$P(\text{error}_{\text{true}}(h) - \text{error}_{\text{min}}(h) > \varepsilon) \leq |H| e^{-2N\varepsilon^2}$$

$$\varepsilon \geq \sqrt{\frac{\ln |H| + \frac{h}{2}}{2N}}$$
PAC bound and Bias-Variance tradeoff

\[ P \left( \text{error}_{\text{true}}(h) - \text{error}_{\text{train}}(h) > \epsilon \right) \leq e^{-2N\epsilon^2} \]

or, after moving some terms around, with probability at least 1-\(\delta\):

\[ \text{error}_{\text{true}}(h) \leq \text{error}_{\text{train}}(h) + \sqrt{\frac{\ln |H| + \ln \frac{1}{\delta}}{2N}} \]

- 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?
Boolean formulas with $m$ binary features

\[ N \geq \frac{\ln|H| + \ln \frac{1}{\delta}}{2e^2} \]

Number of decision trees of depth $k$

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)
$= m \times H_k \times H_k$

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...

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 $2^k \log m$

- Decision trees with up to $k$ leaves:
  - $|H|$ is about $m^k k^{2k}$
    - A very loose bound
PAC bound for decision trees with \( k \) leaves – Bias-Variance revisited

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

\[
\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{2k(\ln m + \ln k) + \ln \frac{1}{\delta}}{2N}}
\]

What did we learn from decision trees?

- Bias-Variance tradeoff formalized

\[
\text{error}_{true}(h) \leq \text{error}_{train}(h) + \sqrt{\frac{2k(\ln m + \ln k) + \ln \frac{1}{\delta}}{2N}}
\]

- Moral of the story:
  Complexity of learning not measured in terms of size hypothesis space, but in maximum \textit{number of points} that allows consistent classification

  - Complexity \( N \) – no bias, lots of variance
  - Lower than \( N \) – some bias, less variance
What about continuous hypothesis spaces?

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

- Continuous hypothesis space:
  - \(|H| = \infty\)
  - 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

©Carlos Guestrin 2005-2013

Clustering images

Set of Images

©Carlos Guestrin 2005-2013

[Goldberger et al.]
Clustering web search results

Some Data

Clustering human data on 6 documents.

©Carlos Guestrin 2005-2013
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)

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
  - \(\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}\)

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

- **Recenter**: \(\mu_i\) becomes centroid of its point:
  - \(\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} \|\mu - x_j\|^2\)
  - Equivalent to \(\mu_i \leftarrow \text{average of its points!}\)
What is K-means optimizing?

- Potential function $F(\mu, C)$ of centers $\mu$ 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 $\mu$, 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 \( \mu \)

Coordinate descent algorithms

- 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!