TDT4173 Machine Learning

Decision Trees, Hypothesis Testing, and Learning Theory

Norwegian University of Science and Technology

Helge Langseth
IT-VEST 310
helgel@idi.ntnu.no

NTNU

Outline

1. Wrap-up from last time
2. Decision trees
   - Basics
   - Entropy and ID3
   - Bias
   - Overfitting
3. Evaluating hypothesis
   - Sample error, true error
   - Estimators
   - Confidence intervals for observed hypothesis error
   - Comparing hypothesis
   - Comparing learners
4. Computational Learning Theory
   - Background
   - Bounding the error
   - PAC learning

First Assignment

- First assignment is out
- Should be delivered by September 6th at 20:00.
- **Question time:**
  Wednesdays 1215 – 1400 in Lars Bungums office (IT-Vest Room 359).
- **Remember:**
  If you for some reason do not pass the assignment it will take 3.33 points from the top of your evaluation (out of the max. 100 points).

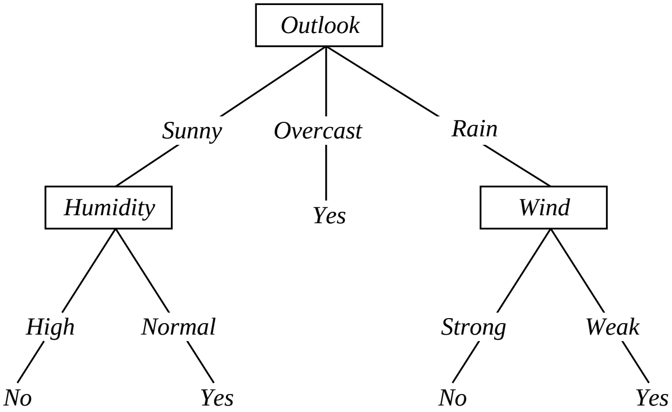
Summary-points from last lesson

1. Hypothesis space:
   - Concept learning as search through H
   - General-to-specific ordering over H
2. Version spaces:
   - Version space candidate elimination algorithm
   - S and G boundaries characterize learner's uncertainty
3. Inductive bias:
   - Inductive leaps possible only if learner is biased
   - Inductive learners can be modelled by equivalent deductive systems

Training Examples for *EnjoySport*

Day	Outlook	Temperature	Humidity	Wind	EnjoySport
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision Tree for *EnjoySport*



Decision Trees

Decision tree representation:

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification

When to Consider Decision Trees

- Instances describable by attribute–value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data

Examples:

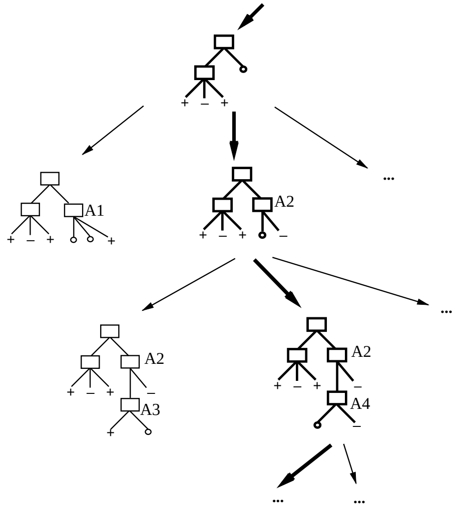
- Equipment or medical diagnosis
- Credit risk analysis
- Classifying email as spam or ham

Top-Down Induction of Decision Trees

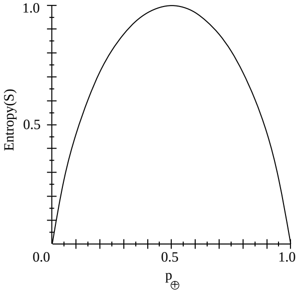
Main loop:

1. $A \leftarrow$ the **best** decision attribute for next `node`
2. Assign A as decision attribute for `node`
3. For each value of A, create new descendant of `node`
4. Sort training examples to leaf nodes
5. `If` training examples perfectly classified, Then `STOP`, `else` iterate over new leaf nodes

Hypothesis Space Search



Entropy



- S is a sample of training examples
- $p_\oplus$ is the proportion of positive examples in S
- $p_\ominus$ is the proportion of negative examples in S
- Entropy measures the impurity of S

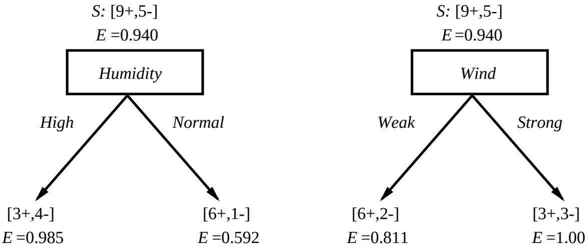
$$\text{Entropy}(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

Information Gain

$\text{Gain}(S, A)$: Expected reduction in entropy due to sorting on A

$$\text{Gain}(S, A) \equiv \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

Which attribute is the best classifier?



Gain (S, Humidity)
= .940 - (7/14).985 - (7/14).592
= .151

Gain (S, Wind)
= .940 - (8/14).811 - (6/14)1.0
= .048

Hypothesis Space Search by ID3

- Hypothesis space is complete, so target function surely in there. . .
- Outputs a single hypothesis
- No back tracking: Local minima. . .
- Statistically-based search choices, so robust to noisy data. . .

Inductive Bias in ID3

Note: Hypothesis space is complete, so H is the power set of instances X.

Does this imply that ID3 is an unbiased learner (lacking both inductive bias as well as preference bias)?

Inductive Bias in ID3

Note: Hypothesis space is complete, so H is the power set of instances X.

Does this imply that ID3 is an unbiased learner (lacking both inductive bias as well as preference bias)?

Not really...

- Preference for short trees, and for those with high information gain attributes near the root
- Bias is a *preference* for some hypotheses, rather than a *restriction* of hypothesis space H
- Occam's razor: prefer the shortest hypothesis that fits the data

Why Occam's Razor?

Why prefer short hypotheses?

Argument in favor:

- Fewer short hyps. than long hyps.
  - a short hyp that fits data unlikely to be coincidence
  - a long hyp that fits data might be coincidence

Argument opposed:

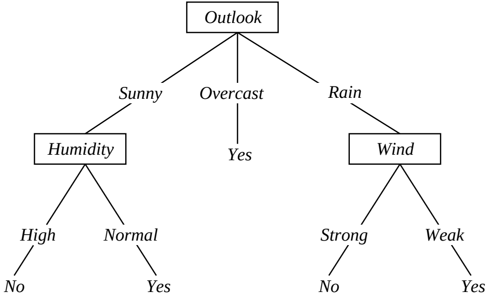
- What's so special about small sets based on *size* of hypothesis??
- There are many ways to define small sets of hypothesis, e.g., all trees with a prime number of nodes that use attributes beginning with "Z"

Overfitting in Decision Trees

Consider adding noisy training example #15:

Sky	Temp	Humid	Wind	Water	Outlook	EnjoySport
Sunny	Hot	Normal	Weak	Warm	Sunny	No

Consider:
What is the effect on the tree we learned earlier?



Overfitting

Consider error of hypothesis h over

- Training data: $\text{error}_t(h)$
- Entire distribution $\mathcal{D}$ of data: $\text{error}_{\mathcal{D}}(h)$

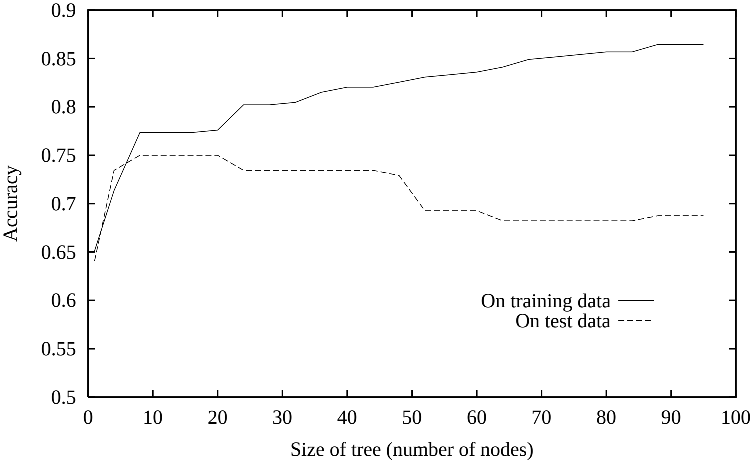
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

$$\text{error}_t(h) < \text{error}_t(h')$$

and

$$\text{error}_{\mathcal{D}}(h) > \text{error}_{\mathcal{D}}(h')$$

Overfitting in Decision Tree Learning



Avoiding Overfitting

How can we avoid overfitting?

- stop growing when data split not statistically significant
- grow full tree, then post-prune

How to select "best" tree:

- Measure performance over training data
- Measure performance over separate validation data set
- MDL: Minimize size(tree) + size(misclassifications(tree))

Reduced-Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

Produces smallest version of most accurate subtree

Nice applet on the web

If you want to learn (more) about decision trees, try this applet:

http://www.cs.ualberta.ca/~aixplore/learning/DecisionTrees/

Two Definitions of Error

The **true error** of hypothesis h with respect to target function f and distribution $\mathcal{D}$ is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$.

$$\text{error}_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[f(x) \neq h(x)]$$

The **sample error** of h with respect to target function f and data sample S is the proportion of examples h misclassifies

$$\text{error}_S(h) \equiv \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$$

Where $\delta(f(x) \neq h(x))$ is 1 if $f(x) \neq h(x)$, and 0 otherwise.

How well does $\text{error}_S(h)$ estimate $\text{error}_{\mathcal{D}}(h)$?

Problems Estimating Error

1. **Bias:** If S is training set, $\text{error}_S(h)$ is optimistically biased

$$\text{bias} \equiv E[\text{error}_S(h)] - \text{error}_{\mathcal{D}}(h)$$

For unbiased estimate, h and S must be chosen independently.

$\rightarrow$ Assume S is a separate *validation set* (for now).

2. **Variance:** Even with unbiased S, $\text{error}_S(h)$ may still *vary* from $\text{error}_{\mathcal{D}}(h)$

Example

Hypothesis h misclassifies 12 of the 40 examples in S

$$\text{error}_S(h) = \frac{12}{40} = .30$$

What is $\text{error}_{\mathcal{D}}(h)$?

Example

Hypothesis h misclassifies 12 of the 40 examples in S

$$\text{error}_S(h) = \frac{12}{40} = .30$$

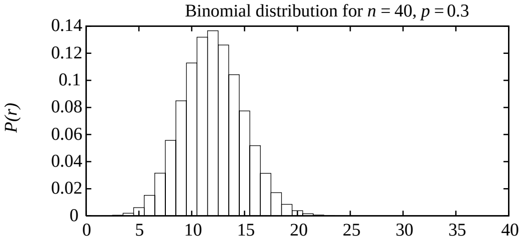
What is $\text{error}_{\mathcal{D}}(h)$?

And if true error is $\text{error}_{\mathcal{D}}(h) = 0.30$, what can we say about the number of misclassifies from 40 examples?

$\text{error}_S(h)$ is a random variable

Rerun the experiment with different randomly drawn S (of size n)

Probability of observing r misclassified examples:



$$P(r) = \frac{n!}{r!(n-r)!} \, \text{error}_{\mathcal{D}}(h)^r (1 - \text{error}_{\mathcal{D}}(h))^{n-r}$$

Estimators

Experiment:

1. choose sample S of size n according to distribution $\mathcal{D}$
2. measure $\text{error}_S(h)$

We know:

- $\text{error}_S(h)$ is a random variable (i.e., result of an experiment)
- $\text{error}_S(h)$ is an unbiased *estimator* for $\text{error}_{\mathcal{D}}(h)$

But:

Given observed $\text{error}_S(h)$ what can we conclude about $\text{error}_{\mathcal{D}}(h)$?

$\rightarrow$ Theory from statistics will give us the answer ...

Central Limit Theorem

Consider a set of independent, identically distributed random variables $Y_1 \dots Y_n$, all governed by an arbitrary probability distribution with mean μ and finite variance σ^2. Define the sample mean,

$$\bar{Y} \equiv \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Central Limit Theorem

As $n \to \infty$, the distribution governing $\bar{Y}$ approaches a Normal distribution, with mean μ and variance $\frac{\sigma^2}{n}$.

Confidence Intervals

If S contains n examples that are drawn independently of h and each other, and $n \geq 30$ then

With approximately 95% probability, $\text{error}_{\mathcal{D}}(h)$ lies in interval

$$\text{error}_S(h) \pm 1.96\sqrt{\frac{\text{error}_S(h)(1-\text{error}_S(h))}{n}}$$

Confidence Intervals

If S contains n examples that are drawn independently of h and each other, and $n \geq 30$ then

With approximately $N\%$ probability, $\text{error}_{\mathcal{D}}(h)$ lies in interval

$$\text{error}_S(h) \pm z_N \sqrt{\frac{\text{error}_S(h)(1 - \text{error}_S(h))}{n}}$$

where

$N\%$:	50%	68%	80%	90%	95%	98%	99%
z_N:	0.67	1.00	1.28	1.64	1.96	2.33	2.58

Calculating Confidence Intervals

1. Pick parameter p to estimate:
   - $\text{error}_{\mathcal{D}}(h)$
2. Choose an estimator:
   - $\text{error}_S(h)$
3. Determine probability distribution that governs estimator:
   - $\text{error}_S(h)$ governed by Binomial distribution, approximated by Normal when $n \geq 30$
4. Find interval (L, U) such that $N\%$ of probability mass falls in the interval:
   - Use table of z_N values

Difference Between Hypotheses

Test h_1 on sample S_1, test h_2 on S_2

1. Pick parameter to estimate: $d \equiv \text{error}_{\mathcal{D}}(h_1) - \text{error}_{\mathcal{D}}(h_2)$
2. Choose an estimator: $\hat{d} \equiv \text{error}_{S_1}(h_1) - \text{error}_{S_2}(h_2)$
3. Determine probability distribution that governs estimator

$$\sigma_{\hat{d}} \approx \sqrt{\frac{\text{error}_{S_1}(h_1)(1 - \text{error}_{S_1}(h_1))}{n_1} + \frac{\text{error}_{S_2}(h_2)(1 - \text{error}_{S_2}(h_2))}{n_2}}$$

4. Find interval (L, U) such that $N\%$ of probability mass falls in the interval

$$\hat{d} \pm z_N \cdot \sigma_{\hat{d}}$$

Comparing learning algorithms L_A and L_B

Moving from comparing **hypothesis** to comparing **learners** we now like to estimate the expected difference in true error between hypotheses output by learners L_A and L_B, when trained using randomly selected training sets S drawn according to distribution $\mathcal{D}$:

$$E_{S \subset \mathcal{D}}[\text{error}_{\mathcal{D}}(L_A(S)) - \text{error}_{\mathcal{D}}(L_B(S))]$$

$L(S)$ is the hypothesis output by learner L using training set S

Given limited data D_0, what is a good estimator?

- Partition D_0 into training set S_0 and test set T_0, and measure

$$\text{error}_{T_0}(L_A(S_0)) - \text{error}_{T_0}(L_B(S_0))$$

- **Other ideas?**

Comparing learning algorithms L_A and L_B

Moving from comparing **hypothesis** to comparing **learners** we now like to estimate the expected difference in true error between hypotheses output by learners L_A and L_B, when trained using randomly selected training sets S drawn according to distribution $\mathcal{D}$:

$$E_{S \subset \mathcal{D}}[\text{error}_{\mathcal{D}}(L_A(S)) - \text{error}_{\mathcal{D}}(L_B(S))]$$

$L(S)$ is the hypothesis output by learner L using training set S

Given limited data D_0, what is a good estimator?

- Partition D_0 into training set S_0 and test set T_0, and measure

$$\text{error}_{T_0}(L_A(S_0)) - \text{error}_{T_0}(L_B(S_0))$$

- Partition data, repeat this for each part, and average!

Comparing learning algorithms L_A and L_B (2)

1. Partition data D_0 into k disjoint test sets $T_1, T_2, \ldots, T_k$ of equal size, where this size is at least 30.
2. For i from 1 to k, do

   use T_i *for the test set, and the remaining data for training set* S_i
   - $S_i \leftarrow \{D_0 - T_i\}$
   - $h_A \leftarrow L_A(S_i)$
   - $h_B \leftarrow L_B(S_i)$
   - $\delta_i \leftarrow \text{error}_{T_i}(h_A) - \text{error}_{T_i}(h_B)$
3. Return the value $\bar{\delta} \equiv \frac{1}{k}\sum_{i=1}^{k} \delta_i$

$N\%$ confidence interval estimate for d:

$$\bar{\delta} \pm t_{N,k-1}\sqrt{\frac{1}{k(k-1)}\sum_{i=1}^{k}(\delta_i - \bar{\delta})^2}$$

Note! δ_i and $\bar{\delta}$ are approximately Normally distributed.

Computational Learning Theory

Top-level question:
What general laws constrain inductive learning?

We seek theory to relate:

- Probability of successful learning
- Number of training examples
- Complexity of hypothesis space
- Accuracy to which target concept is approximated
- The manner in which training examples are presented

Prototypical Concept Learning Task

Given:

- Instances $x \in \mathcal{X}$: Possible days, each described by the attributes *Sky, AirTemp, Humidity, Wind, Water, Forecast*
- Target function c: $\texttt{EnjoySport} : \mathcal{X} \rightarrow \{0, 1\}$
- Hypotheses H: Conjunctions of literals. E.g.

$$\langle ?, \text{Cold}, \text{High}, ?, ?, ? \rangle.$$

- Training examples $\boldsymbol{D}$: Positive and negative noise free examples of the target function

$$\langle x_1, c(x_1) \rangle, \ldots, \langle x_m, c(x_m) \rangle$$

Determine:

- A hypothesis h in H such that $h(x) = c(x)$ for all $x \in \boldsymbol{D}$.
- A hypothesis h in H such that $h(x) = c(x)$ for all $x \in \mathcal{X}$.

Sample Complexity

How many training examples are sufficient to learn the target concept?

1. If learner proposes instances, as queries to teacher
   - Learner proposes instance x, teacher provides $c(x)$
2. If teacher (who knows c) provides training examples
   - Teacher provides sequence of examples of form $\langle x, c(x) \rangle$
3. If some random process (e.g., nature) proposes instances
   - Instance x generated randomly, teacher provides $c(x)$

Sample Complexity: Case 1

Learner proposes instance x, teacher provides $c(x)$
(assume c is in learner's hypothesis space H)

Optimal query strategy: pretend to play 20 questions

- Pick instance x such that half of hypotheses in `VersionSpace` classify x positive, half classify x negative
- When this is possible, need $\lceil \log_2 |H| \rceil$ queries to learn c
- When not possible, we need more queries

Sample Complexity: Case 2

Teacher (who knows c) provides training examples
(assume c is in learner's hypothesis space H)

Optimal teaching strategy: Depends on H used by learner

Consider the case $H =$ conjunctions of up to n boolean literals and their negations

- e.g., $(\text{AirTemp} = \text{Warm}) \wedge (\text{Wind} = \text{Strong})$, where $\text{AirTemp}, \text{Wind}, \ldots$ each have 2 possible values.
- If there are n possible boolean attributes in H, it will suffice with $n + 1$ examples. **Why?**

Sample Complexity: Case 3

Given:

- Set of instances $\mathcal{X}$
- Set of hypotheses H
- Set of possible target concepts $\mathcal{C}$
- Training instances generated by a fixed, unknown probability distribution $\mathcal{D}$ over $\mathcal{X}$

Sample Complexity: Case 3

Given:

- Set of instances $\mathcal{X}$
- Set of hypotheses H
- Set of possible target concepts $\mathcal{C}$
- Training instances generated by a fixed, unknown probability distribution $\mathcal{D}$ over $\mathcal{X}$

Learner observes a sequence $\boldsymbol{D}$ of training examples of form $\langle x, c(x) \rangle$, for some target concept $c \in \mathcal{C}$:

- instances x are drawn from distribution $\mathcal{D}$
- teacher provides target value $c(x)$ for each

Learner must output a hypothesis h estimating c:

- h is evaluated by its performance on subsequent instances drawn according to $\mathcal{D}$

Note: randomly drawn instances, noise-free classifications

Two Notions of Error

Remember the definition of the **True error** of hypothesis h with respect to c:

- How often $h(x) \neq c(x)$ over future random instances

Now, also focus on **Training error** of hypothesis h with respect to target concept c

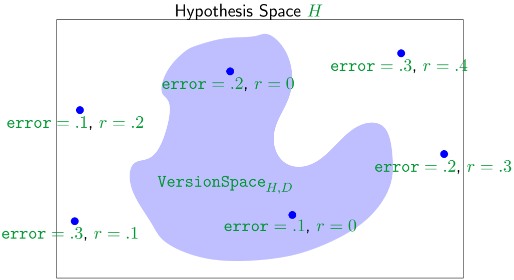
- How often $h(x) \neq c(x)$ over training instances

Focus for the rest of the lesson

Earlier today we considered the **sample error** on a **validation set** because we wanted to avoid bias.

From now on we try to bound the true error of h given that the **training error** of h on the **training set** is zero (i.e., $h \in \texttt{VersionSpace}_{H,D}$)

Exhausting the Version Space



(r: training error, error: true error)

Definition (ϵ-exhausted)

The version space $\texttt{VersionSpace}_{H,D}$ is said to be ϵ-exhausted with respect to c and $\mathcal{D}$, if every hypothesis h in $\texttt{VersionSpace}_{H,D}$ has error less than ϵ with respect to c and $\mathcal{D}$.

$$(\forall h \in \texttt{VersionSpace}_{H,D})\ \text{error}_{\mathcal{D}}(h) < \epsilon$$

How many examples will ϵ-exhaust the VS?

Theorem (Haussler, 1988)

If the hypothesis space H is finite, and $\boldsymbol{D}$ is a sequence of $m \geq 1$ independent random examples of some target concept c, then for any $0 \leq \epsilon \leq 1$, the probability that the version space with respect to H and $\boldsymbol{D}$ is not ϵ-exhausted (with respect to c) is less than

$$|H|e^{-\epsilon m}$$

$\rightarrow$ This bounds the probability that any consistent learner will output a hypothesis h with $\text{error}(h) \geq \epsilon$:

If we want this probability to be below δ, $|H|e^{-\epsilon m} \leq \delta$, then

$$m \geq \frac{1}{\epsilon}(\ln|H| + \ln(1/\delta))$$

Example: Learning Conjunctions of Boolean Literals

How many examples are sufficient to assure with probability at least $(1-\delta)$ that every h in $\texttt{VersionSpace}_{H,D}$ satisfies $\text{error}_{\mathcal{D}}(h) \leq \epsilon$?

Use the theorem:

$$m \geq \frac{1}{\epsilon}(\ln |H| + \ln(1/\delta))$$

Suppose H contains conjunctions of constraints on up to n boolean attributes (i.e., n boolean literals).

Then $|H| = 3^n$, and

$$m \geq \frac{1}{\epsilon}(\ln 3^n + \ln(1/\delta)) = \frac{1}{\epsilon}(n \ln 3 + \ln(1/\delta))$$

How About EnjoySport?

$$m \geq \frac{1}{\epsilon}(\ln |H| + \ln(1/\delta))$$

If H is as given in `EnjoySport` then $|H| = 973$, and

$$m \geq \frac{1}{\epsilon}(\ln 973 + \ln(1/\delta))$$

If want to assure that with probability 95%, `VersionSpace` contains only hypotheses with $\text{error}_{\mathcal{D}}(h) \leq .1$, then it is sufficient to have m examples, where

$$\begin{aligned} m &\geq \frac{1}{.1}(\ln 973 + \ln(1/.05)) \\ &= 10(6.88 + 3.00) \\ &= 98.8 \end{aligned}$$

PAC Learning

Consider a class $\mathcal{C}$ of possible target concepts defined over a set of instances $\mathcal{X}$ of length n, and a learner L using hypothesis space H.

Definition (PAC-learnable)

$\mathcal{C}$ is PAC (Probably Approximately Correct)-learnable by L using H if for all

- $c \in \mathcal{C}$,
- distributions $\mathcal{D}$ over $\mathcal{X}$,
- ϵ such that $0 < \epsilon < 1/2$, and
- δ such that $0 < \delta < 1/2$,

the learner L will – with probability at least $(1 - \delta)$ – output a hypothesis $h \in H$ such that $\text{error}_{\mathcal{D}}(h) \leq \epsilon$, in time that is polynomial in $1/\epsilon$, $1/\delta$, n and $\text{size}(c)$.

Example: PAC Learning of Conjunction of Boolean Literals

Is there a learner L which makes the "Conjunction of Boolean Literals" - problem PAC learnable?

- Number of examples required by consistent learner:

$$m = \frac{1}{\epsilon}(n \ln 3 + \ln(1/\delta))$$

- `Find-S` is consistent, and the number of operations required per training example for `Find-S` is $O(n)$.
- Learning is linear in $1/\epsilon$, logarithmic in $1/\delta$, linear in n and constant in size(c).

So... Yes, it is PAC learnable, e.g. using `Find-S`!