CS 2750 Machine Learning
Lecture 22

Concept learning

Milos Hauskrecht
milos@cs.pitt.edu

Concept Learning

Outline:

- Learning boolean functions
- Most general and most specific consistent hypothesis.
- Mitchell's version space algorithm
- Probably approximately correct (PAC) learning.
- Sample complexity for PAC.
- Vapnik-Chervonenkis (VC) dimension.
- Improved sample complexity bounds.

Learning concepts

Assume objects (examples) described in terms of attributes:

Sky	Air-Temp	Humidity	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Warm	Same	yes
Rainy	Cold	Normal	Strong	Warm	Change	no

Concept = a set of objects

- **Concept learning:**
  Given a sample of labeled objects we want to learn a boolean mapping from objects to T/F identifying an underlying concept
  - E.g. EnjoySport concept
- Concept (hypothesis) space H
  - Restriction on the boolean description of concepts

Learning concepts

- Object (instance) space X
- Concept (hypothesis) spaces H

$$H \neq X \qquad !!!!$$

Assume *n* binary attributes (e.g. true/false, warm/cold)

- **Instance space X:**

  2^n different objects

- **Concept space H:**

  2^{2^n} possible concepts

  = all possible subsets of objects

Learning concepts

- **Problem:** Concept space too large
- **Solution:** restricted hypothesis space H
- Example: **conjunctive concepts**

$$(Sky = Sunny) \wedge (Weather = Cold)$$

3^n possible concepts **Why?**

- Other restricted spaces:

3-CNF (or k-CNF) $(a_1 \vee a_3 \vee a_7) \wedge (...)$

3-DNF (or k-DNF) $(a_1 \wedge a_5 \wedge a_9) \vee (...)$

Learning concepts

- After seeing k examples the hypothesis space (even if restricted) can have many consistent concept hypotheses
- **Consistent hypothesis:** a concept *c* that evaluates to T on all positive examples and to F on all negatives.
- What to learn?
  - **General to specific learning.** Start from all true and refine with the maximal (consistent) generalization.
  - **Specific to general learning.** Start from all false and refine with the most restrictive specialization.
  - **Version space learning**. Keep all consistent hypothesis around – the combination of the above two cases.

Specific to general learning (for conjunctive concepts)

Assume two hypotheses: **arbitrary**

$$h1 = (Sunny, ?, ?\ Strong, ?, ?)$$

$$h2 = (Sunny, ?, ?, ?, ?, ?)$$

Then we say that:

h2 is more general than *h1*,
h1 is a special case (specialization of) h2

Specific to general learning:

- start from the all-false hypothesis $h0 = (-,-,-,-,-,-)$
- by scanning samples, gradually refine the hypothesis (make it more general) whenever it does not satisfy the new sample seen (keep the most restrictive specialization of positives)

Specific to general learning. Example

Conjunctive concepts, target is a conjunctive concept

$h = (-,-,-,-,-,-)$ All false

(Sunny,Warm, Normal, Strong, Warm, Same) T ←

$$h = (Sunny, Warm, Normal, Strong, Warm, Same)$$

(Rainy, Cold, Normal, Strong, Warm, Change) F

$$h = (Sunny, Warm, Normal, Strong, Warm, Same)$$

(Sunny,Warm, **High**, Strong, Warm, Same) T ←

$$h = (Sunny, Warm, ?, Strong, Warm, Same)$$

(Sunny,Warm, High, Strong, **Cool**, Same) T ←

$$h = (Sunny, Warm, ?, Strong, ?, Same)$$

General to specific learning

- Dual problem to the specific to general learning
- Start from the all true hypothesis $h0 = (?, ?, ?, ?, ?, ?)$
- Refine the concept description such that all samples are consistent (keep maximal possible generalization)

$h = (?, ?, ?, ?, ?, ?)$

(Sunny,Warm, Normal, Strong, Warm, Same) T

$h = (?, ?, ?, ?, ?, ?)$

(Sunny,Warm, High, Strong, Warm, Same) T

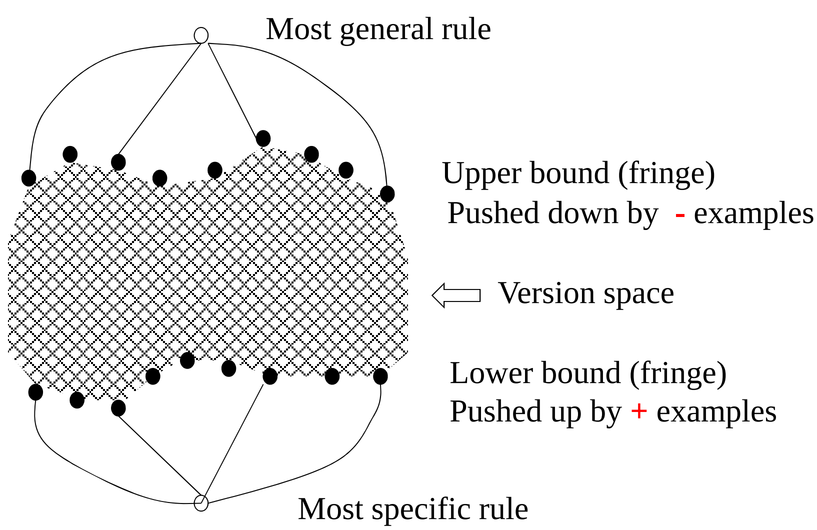
$h = (?, ?, ?, ?, ?, ?)$

(**Rainy**, **Cold**, Normal, Strong, Warm, **Change**) F

$h = (Sunny\ , ?, ?, ?, ?, ?), (?, Warm\ ?, ?, ?, ?),$
$(?, ?, ?, ?, ?, Same\ )$

Mitchell's version space algorithm

- **Keeps the space of consistent hypotheses**



Mitchell's version space algorithm

- Keeps and refines the fringes of the version space
- Converges to the target concept whenever the target is a member of the hypotheses space H
- Assumption:
  - No noise in the data samples (the same example has always the same label)
- The hope is that the fringe is always small

Is this correct ?

Exponential fringe set – example

Conjunctive concepts, upper fringe (general to specific)

Samples: $(true, true, true, true, ..., true)$ T

$$\frac{1}{2}n \left\{ \begin{array}{ll} (false, false, true, true, ..., true) & F \\ (true, true, false, false, ..., true) & F \\ \dots & \\ (true, true, true, ..., false, false) & F \end{array} \right.$$

Maximal generalizations – different hypotheses we need to remember

$$2^{\frac{n}{2}} \left\{ \begin{array}{l} (true, ?, true, ?, ..., true, ?) \\ (?, true, true, ?, ..., true, ?) \\ (true, ?, ?, true, ..., true, ?) \\ \dots \\ (?, true, ?, true, ..., ?, true) \end{array} \right.$$

Learning concepts

- Version space algorithm may require large number of samples to converge to the target concept
  - In the worst case we must see all concepts before converging to it.
  - The samples may come from different distributions – it may take a very long time to see all examples
- The fringe can go exponential in the number of attributes
- Alternative solution: Select a hypothesis that is consistent after some number of (+, -) samples is seen by our algorithm
- Can we tell how far are we from the solution?

  Yes !!! PAC framework develops the criteria for measuring the accuracy of our choice in probabilistic terms

Valiant's framework

- Probability distribution from which samples are drawn
- There is an error permitted in assigning the labels to examples
  - The concept learned does not have to be perfect but it should not be very far from the target concept

c_T - target concept
c - learned concept
x - next sample from the distribution

$$Error\,(c_T, c) = P(x \in c \wedge x \notin c_T) + P(x \notin c \wedge x \in c_T)$$

ε - accuracy parameter

We would like to have concept such that $Error\,(c_T, c) \leq \varepsilon$

PAC learning

- To get the error to be smaller than the accuracy parameter in all cases may be hard:
  - Some examples may be very rare and to see them may require large number of samples
- Instead we choose:

$$P(Error\ (c_T, c) \leq \varepsilon) = 1 - \delta$$

where δ is a confidence factor

- **Probably approximately correct (PAC)** learning
  With probability $1-\delta$ a concept with an error not more than ε is found

Sample complexity of PAC learning

- How many samples we need to see to satisfy PAC criterion?

Assume:

we saw *m* independent samples drawn from the distribution, and

h is a hypothesis that is consistent with all *m* examples and its error is larger than epsilon $Error\ (c_T, h) > \varepsilon$

$$P(\text{a sample is consistent with a given } h) \leq (1-\varepsilon)$$

$$P(m \text{ samples are consistent with a given } h) \leq (1-\varepsilon)^m$$

There are at most $|H|$ hypotheses in the space

$$P(\text{any bad hypothesis survives } m \text{ samples}) \leq |H|(1-\varepsilon)^m$$

Sample complexity of PAC learning

$$P(\text{any bad hypothesis survives } m \text{ samples}) \leq |H|(1-\varepsilon)^m$$
$$\leq |H| e^{-\varepsilon m}$$

In the PAC framework we want to bound this probability with the confidence factor δ

$$|H| e^{-\varepsilon m} \leq \delta$$

Expressing for m

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

After m samples satisfying the above inequality any consistent hypothesis satisfies the PAC criterion

Efficient PAC learnability

- The concept is efficiently PAC learnable if the time it takes to output the concept is polynomial in $n, 1/\varepsilon, 1/\delta$

Two aspects:

- **Sample complexity** – a number of examples needed to learn the concept satisfying PAC criterion
  - A prerequisite to efficient PAC learnability
- **Time complexity** – the time it takes to find the concept
  - Even if the sample complexity is OK, the learning procedure may not be efficient (e.g. exponential fringe)

Efficient PAC learnability

- Sample complexities depends on the hypothesis space we use
- **Conjunctive concepts** 3^n possible concepts

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

efficient

- **All possible concepts** (unbiased hypothesis space)

$$m \geq \frac{(\ln(1/\delta) + \ln 2^{2^n})}{\varepsilon} = \frac{(\ln(1/\delta) + 2^n \ln 2)}{\varepsilon}$$

inefficient

Efficient PAC learnability

- Polynomial sample complexity is necessary but not sufficient
- Algorithm should work in polynomial time
- Assume: **learning conjunctive concepts**
  - Specific to general learning. It is sufficient to keep one hypothesis around. The most specific description of all positive examples. Can be done in poly time.
  - General to specific learning. We need to keep the complete upper fringe which can be exponential. Cannot be done in poly time.
- Other concept (hypothesis) spaces with poly sample complexity:
  - k-DNF – cannot be PAC learned in poly time.
  - k-CNF – polynomial time solution

Learning 3-CNF

- Sample complexity for the k-CNF and k-DNF
- k-DNF – cannot be learned efficiently
- k-CNF – can be learned efficiently. How?

  Assume 3-CNF $(a_1 \vee a_3 \vee a_7) \wedge (a_2 \vee \neg a_4 \vee a_5) \wedge \ldots$

Only a polynomial number of clauses with at most 3 variables !!

$$2n + 2n2(n-1) + 2n2(n-1)2(n-2) = O(n^3)$$

Algorithm (specific to general learning):

- Start with the conjunction of all possible clauses (always false)
- On positive example any clause that is not true is deleted
- On negative examples do nothing

Interesting Any k-DNF can be converted into k-CNF

Quantifying inductive bias

- During learning only small fraction of samples seen
- We need to generalize to unseen examples
- Choice of the hypotheses space restrict our learning options – biases our learning
- Other biases: preference towards simpler hypothesis, smaller degrees of freedom

Questions:

How to measure the bias?

To what extent our biases affect our learning capabilities?

Can we learn even if the hypotheses space is infinite?

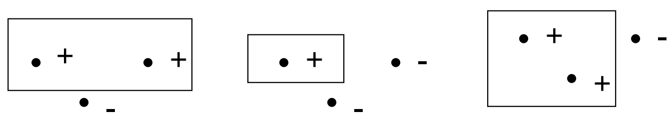
$$m \geq \frac{(\ln(1/\delta) + \ln|H|)}{\varepsilon}$$

CS 2750 Machine Learning

Vapnik-Chervonenkis dimension

- Measures the biases of the concept space
- Allows us to:
  - Obtain better sample complexity bound
  - Can be extended to attributes with infinite value spaces.
- **VC idea**: do not measure the size of the space, but the number of distinct instances that can be completely discriminated using *H*

Example: H is a set of space of rectangles

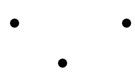


Discrimination of labelings of 3 points with rectangles

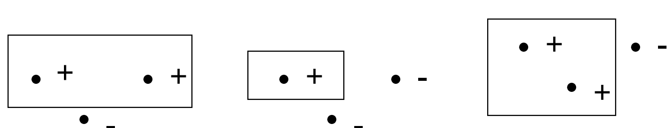
Shattering of a set of instances

- A set of instances $S \subseteq X$
- *H* shatters *S* if for every dichotomy (combination of labels) there is a hypothesis *h* consistent with the dichotomy

Example: *H* is a set of space of rectangles



A set of 3 instances (most flexible choice)



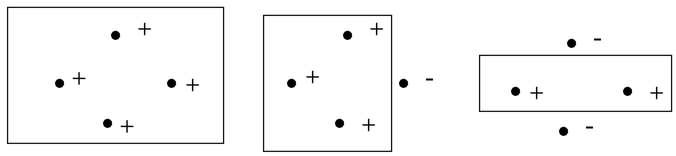
Dichotomy 1 Dichotomy 2 Dichotomy k

2^3 different dichotomies, hypothesis for each of them

Vapnik-Chervonenkis dimension

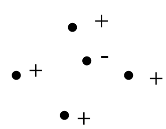
- VC dimension of a hypothesis space *H* is the size of the largest subset of instances that is shattered by *H*.
- Example: rectangles (VC at least 3)

Try 4:



Can be shattered (for the most flexible 4), VC dimension at least 4

Try 5:



No set of 5 points that can be shattered, thus VC dimension is 4

VC dimension and sample complexity

- One can derive the sample complexity bound for PAC learning using VC dimension instead of hypothesis space size
  (we won't do it here)

$$m \ge \frac{(4\ln(2/\delta) + 8\,VC\dim(H)\ln(13/\varepsilon))}{\varepsilon}$$

Adding noise

- We have a target concept but there is a chance of mislabeling the examples seen
- Can we PAC-learn also in this case?
- Blumer (1986). If *h* is a hypothesis that agrees with at least

$$m = \frac{1}{\varepsilon}\ln(\frac{n}{\delta})$$

samples drawn from the distribution then

$$P(error\ (h, c_T) \geq \varepsilon) \leq \delta$$

Mitchell gives the sample complexity bound for the choice of the hypothesis with the best training error

Summary

- Learning boolean functions
- Most general and most specific consistent hypothesis.
- Mitchell's version space algorithm
- Probably approximately correct (PAC) learning.
- Sample complexity for PAC.
- Vapnik-Chervonenkis (VC) dimension.
- Improved sample complexity bounds.
- Adding noise.