Machine Learning

CSE 142

Xin (Eric) Wang

Friday, October 15, 2021

Today

- Concept learning, Ch. 4
- *Decision trees, Ch.5

Notes

- HW1 due next Monday 11:59pm PT
- https://citris.sites.ucsc.edu/tech-for-social-good/

MISSION

The **CITRIS and the Banatao Institute** Tech for Social Good Program supports student-led learning and technology development for healthy, sustainable, connected, and equitable livelihoods in the United States and abroad. We have partnered with the Institute for Social Transformation to bring the program to the UC Santa Cruz campus.

To achieve its mission, the CITRIS Tech for Social Good Program provides funding support to undergraduate, graduate, and postdoctoral students, groups, teams or organizations developing hardware, software, events or programs and supporting innovation and entrepreneurship for social good at UCSC.

If you are interested in developing a technology for social good, please visit the **Technology Development Track** for more information.

If you are interested in applying for funding to support a student-led event or programming that promotes technology for social good, please visit the

Student-Led Events Track for more information.

Information and matchmaking sessions for students – plus staff and faculty working with students they feel would be interested – will be held via Zoom on:

- Monday, October 18th, 2021 from 12:15 – 1:00 pm – RSVP here: https://ucsc.zoom.us/meeting/register/tJIpdeCrqjspGN3GEyHrs8IKfQCyiX9Vdbhj
- Monday, October 25th, 2021 from 4:15 – 5:00 pm – RSVP here: https://ucsc.zoom.us/j/99625959469?pwd=YlNXdlRQaWFpQmFJb3ZzcXlacUVOZz09

See examples of funded projects from the Tech for Social Good Programs at UC Santa Cruz at UC Berkeley and UC Davis.

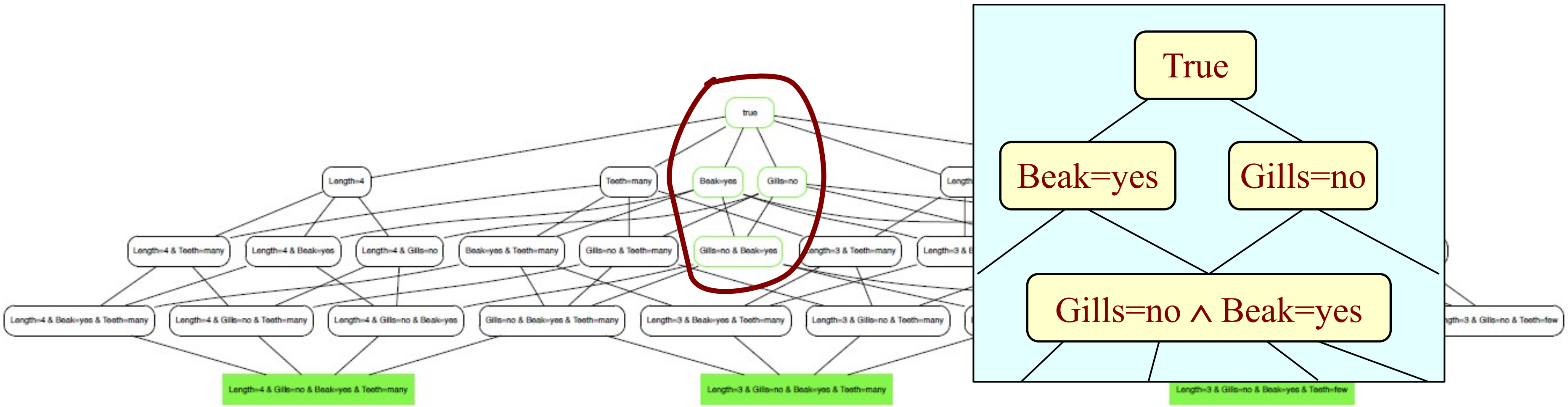
Pruning the hypothesis space from training data

- Given that hypothesis space and our training examples:
  - Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many
  - Length = 4 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many
  - Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = few

- Let's rule out all the hypotheses (concepts) that don't include at least one of the instances in our example
  - I.e., delete nodes that don't fit with at least one training example
  - This leaves us with just 32 conjunctive concepts (out of the original 108)

Pruning the hypothesis space from training data

- But if we require hypotheses to cover <u>all</u> three examples, we're left with only four concepts



- Let's choose the least general of these as our result – the concept defined by our training data

  Gills = no $\wedge$ Beak = yes

Least general generalization (LGG)

- We want to generalize beyond our specific training data, but not too much – the most general hypothesis is to accept everything
- Thus we'd like the *least general* generalization
  - General enough to include all of our training data, but no more general than that

Least general generalization (LGG) procedure

Algorithm LGG-Set(D) – find least general generalisation of a set of instances.

Input : data D.
Output : logical expression H.

$x \leftarrow$ first instance from D;
$H \leftarrow x$;
while instances left **do**
  $x \leftarrow$ next instance from D;
  $H \leftarrow$ LGG(H, x) ;   // e.g., LGG-Conj (Alg. 4.2) or LGG-Conj-ID (Alg. 4.3)
end
return H

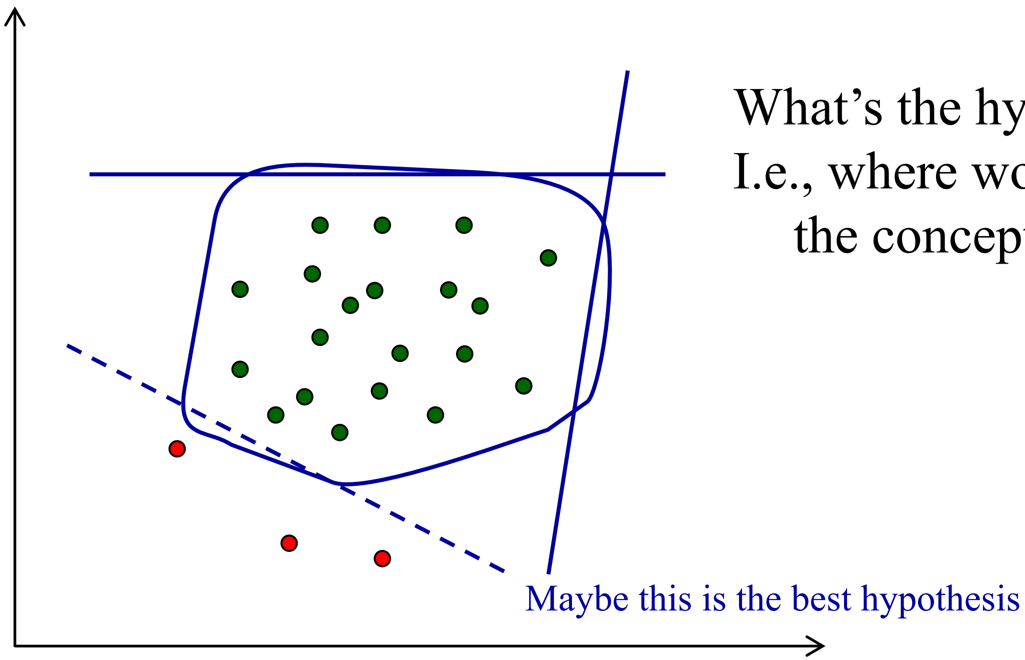
Algorithm LGG-Conj(x, y) – find least general conjunctive generalisation of two conjunctions.

Input : conjunctions x, y.
Output : conjunction z.

$z \leftarrow$ conjunction of all literals common to x and y;
return z

Negative examples

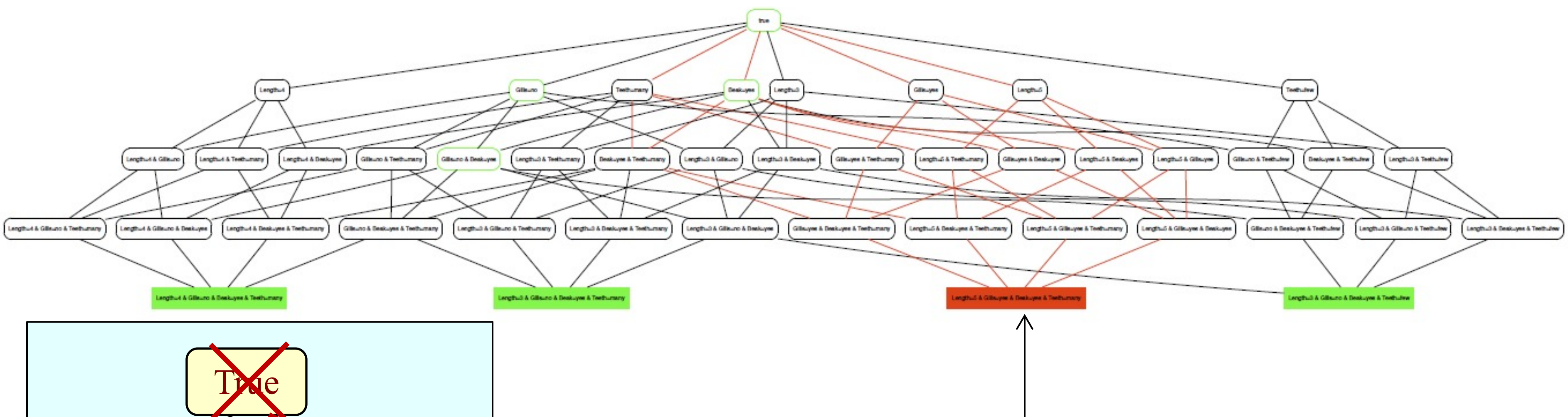
- So far, we've only looked at CHS learning from positive training examples (examples of the concept)
- Negative examples are very useful in learning concepts (for people and machines!)
  - Negative examples help to prune the hypothesis space
    - E.g., in learning *horse*, it would be helpful to show zebras and donkeys

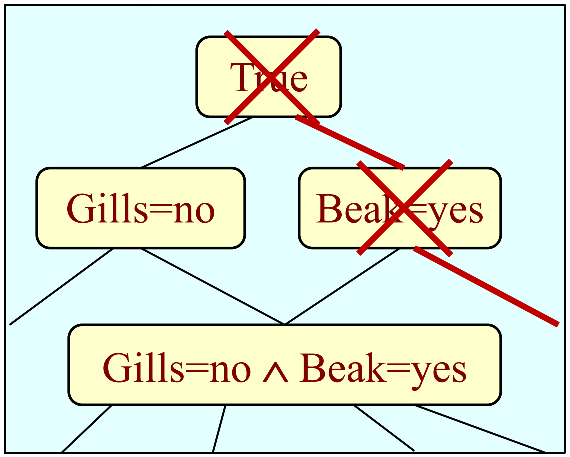


What's the hypothesis?
I.e., where would we draw the concept boundary?

Negative examples

- We'd like to refine our hypothesis (guide our search through hypothesis space) using both positive and negative examples
  - Look for hypotheses that cover positive examples
  - Rule out hypotheses that include negative examples

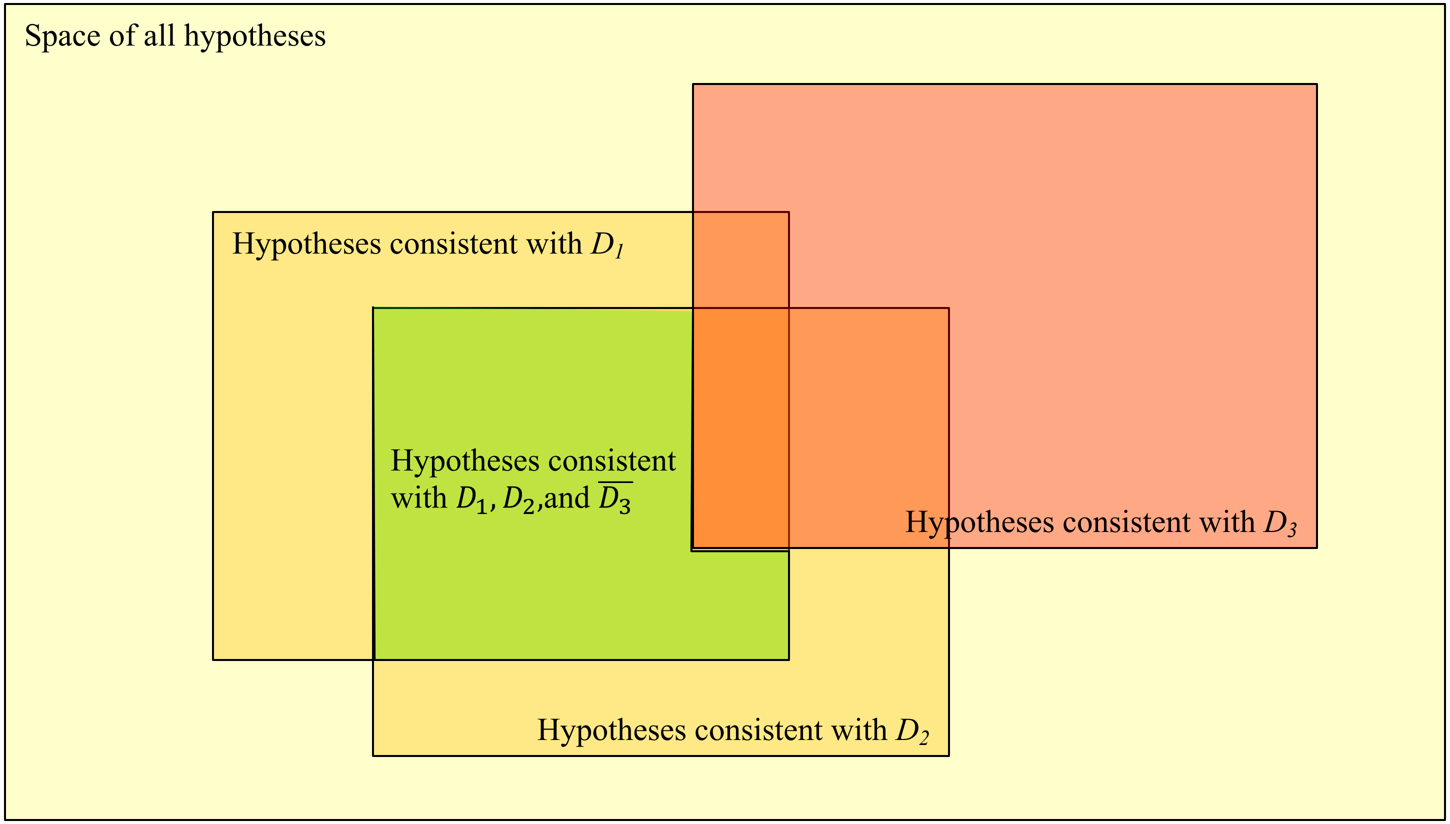




Negative sample: Length=5 ∧ Gills=yes ∧ Beak=yes ∧ Teeth=many

Pruning the hypothesis space

A Venn diagram to show how both positive and negative examples prune the space of consistent hypotheses:



Adding internal disjunction to CHS

- We can make our representation somewhat richer by allowing internal disjunctions (ORs within a feature)
- So instead of positive examples of length=4 and length=5 causing a conflict and thus resulting in our concept containing length=X, we can use length=[4,5] in our concept
  - This means length=4 $\vee$ length=5
- F attribute values → $2^F - 1$ combinations w/internal disjunctions
- Allowing internal disjunction increases the size of our hypothesis space
  - Rather than $|H| = (3+1)(2+1)(2+1)(2+1) = 108$, we'll have $|H| = (2^3-1)(2^2-1)(2^2-1)(2^2-1) = 189$ hypotheses

Adding internal disjunction to CHS

- E.g., for Quarter, $F = 4$ so there are $2^4 - 1 = 15$ combinations:

  Quarter=Fall, Quarter=Winter, Quarter=Spring, Quarter=Summer,

  Quarter=[Fall, Winter], Quarter=[Fall, Spring], Quarter=[Fall, Summer], Quarter=[Winter, Spring], Quarter=[Winter, Summer], Quarter=[Spring, Summer],

  Quarter=[Fall, Winter, Spring], Quarter=[Fall, Winter, Summer], Quarter=[Fall, Spring, Summer], Quarter=[Winter, Spring, Summer],

  ➡ Quarter=[Fall, Spring, Summer, Winter]

- When all values of a feature are included, the feature becomes X (*don't care*)

Equivalent
- Quarter=[Fall, Spring, Summer, Winter]
- Quarter=Fall $\vee$ Quarter=Winter $\vee$ Quarter=Spring $\vee$ Quarter=Summer
- Quarter=X

Quiz: Adding internal disjunction to our example

We (again) take a specific-to-general approach in coming up with a hypothesis:

Instances: Hypotheses:

(1) Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many

(2) Length = 4 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many

(3) Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = few

Adding internal disjunction to our example

We (again) take a specific-to-general approach in coming up with a hypothesis:

Instances: Hypotheses:

(1) Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many

Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many

(2) Length = 4 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many

Length = [3,4] $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = many

(3) Length = 3 $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = few

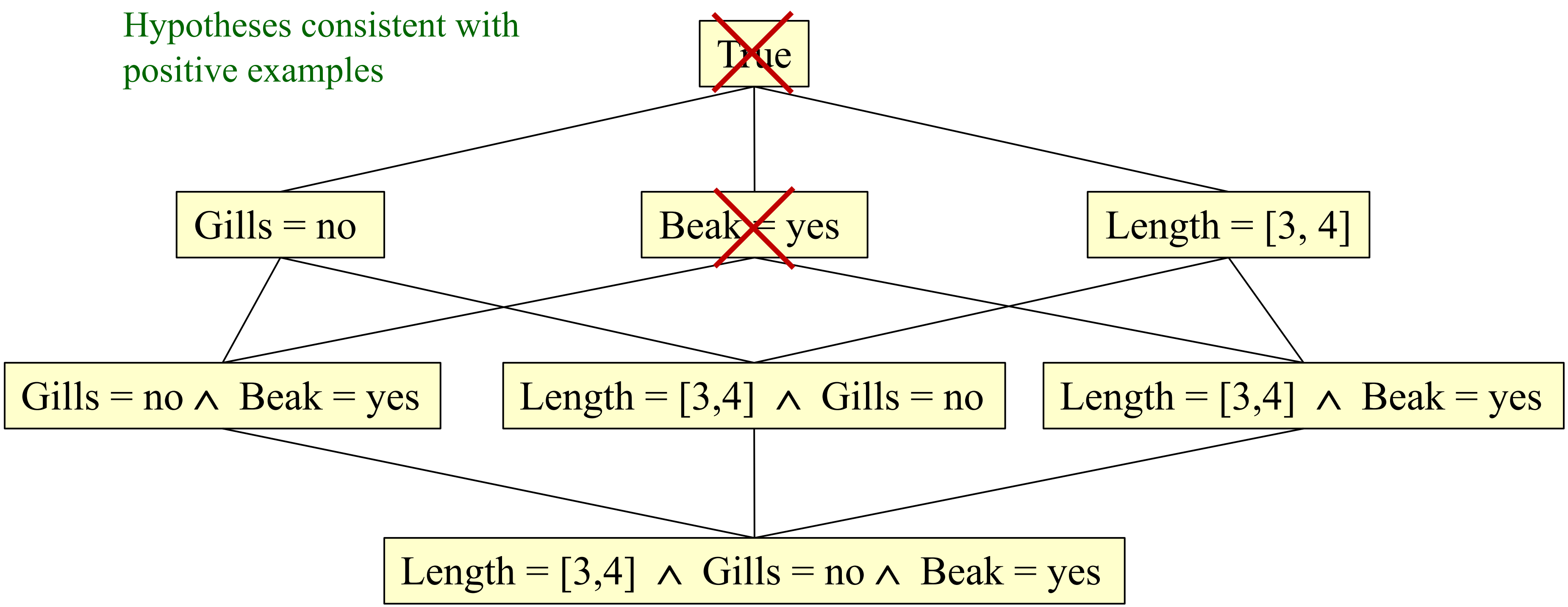
Length = [3,4] $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = X

(4) [Negative] Length = 5 $\wedge$ Gills = yes $\wedge$ Beak = yes $\wedge$ Teeth = many

Length = [3,4] $\wedge$ Gills = no $\wedge$ Beak = yes $\wedge$ Teeth = X

The negative example isn't covered by the hypothesis (it's not in the set of instances defined by the hypothesis), so there's no conflict.

Internal disjunction – adding negative example



Add negative example: Length = 5 ∧ Gills = yes ∧ Beak = yes ∧ Teeth = many

Which hypotheses are no longer consistent with the data?

CHS learning – notes

- There are two different issues here:
  1. The number of hypotheses consistent with the data
     - For a given training data example, which is a leaf node at the lowest level in the graph, all connected nodes above it (including the data point itself) are hypotheses consistent with that data point
     - Adding a new, different data point prunes the nodes that represent hypotheses (in)consistent with the data
       - For positive and negative data points
  2. The generality of various hypotheses
     - Higher/lower nodes represent more/less general hypotheses

- Our algorithm of finding a hypothesis from training data (starting with the first data point as the initial hypothesis and generalizing with more data points) prunes hypotheses at each step, choosing the least general consistent hypothesis as the best current hypothesis

CHS learning – notes

- Adding a positive training example prunes the space of consistent hypotheses by eliminating hypothesis that do not have a link to the new example
  - I.e., candidate hypotheses must be connected to all positive examples
  - The less similar it is to the previous training examples, the more it will prune
- Adding a negative training example prunes the space of consistent hypotheses by eliminating hypotheses that do have a link to the new example
  - I.e., candidate hypotheses must not be connected to any negative examples
  - The more similar it is to the previous training examples, the more it will prune
- A negative example will always prune the top node (True)
- The method we've described won't work well with noisy data
  - This idea can be extended to more realistic data by modifying the "all or nothing" nature of positive/negative training data

Hypothesis languages for concept learning

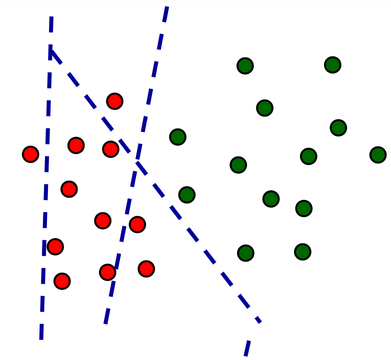
- We can make richer hypothesis representation languages:
  - Hypothesis space
  - Conjunctive hypothesis space
  - Conjunctive hypothesis space with internal disjunctions
  - Conjunctions of Horn clauses
  - Clauses in first-order logic
  - Etc.
- The richer the representation and thus the more expressive the hypothesis language, the more difficult the learning problem
  - Learnability – how hard it is to learn a concept?
- Let's look at some important learning concepts and terms:

"Cover" – classify as positive

Complete and consistent concepts

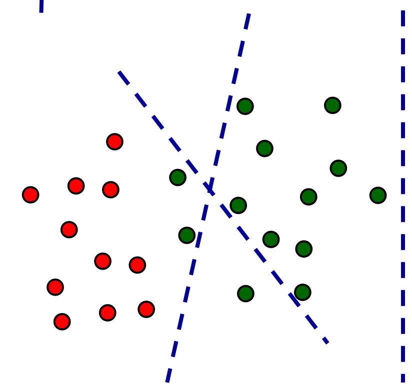
A concept is **complete** if it <u>covers</u> all positive examples:

- FNR = 0



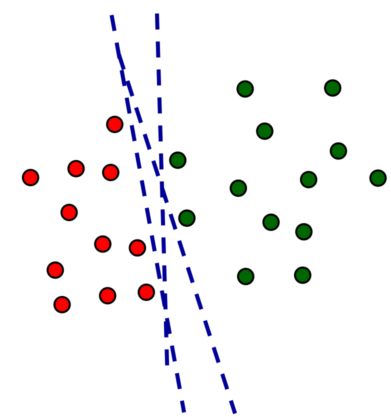
A concept is **consistent** if it covers none of the negative examples:

- FPR = 0



The **version space** is the set of all concepts that are both complete and consistent:

- FPR = FNR = 0



PAC learning

- Let's consider an important learning model: Probably Approximately Correct (PAC) learning
- If a concept is PAC-learnable, then there exists a learning algorithm that gets it mostly right, most of the time
- Terms:
  - Hypothesis: h, hypothesis space: H
  - Distribution of the (true) data: D
  - Error rate of h for data distribution D: err_D
  - Allowable error rate: ε
  - Allowable failure rate: δ

"most of the time"

- PAC learning outputs, with probability at least $1-\delta$, a hypothesis h such that $err_D < \varepsilon$

"mostly right"

Low generalization error

PAC learning

We may get back to this later, but wanted to mention it now since it's briefly discussed in Chapter 4….

- Even with noise-free data and a complete and consistent hypothesis ***h*** (i.e., no errors on the training data), the training data may not have been perfectly representative of the instance space, and the hypothesis might have a "large" error ($err_D > \varepsilon$) over the instance space.
  - This should happen infrequently, with probability less than δ
- It turns out that we can guarantee this by choosing a large enough training set, $m = |D|$
  - With various assumptions, this is:

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

- This leads to the concept of VC dimension, which is a key theoretical concept in machine learning

Next

- Decision tree (Chapter 5.1)
- Ranking and probability estimation trees (Chapter 5.2, 5.4)
- Linear models (Chapter 7)
- HW1 due by next Monday 11:59pm PT