How good is a learning algorithm?

Machine Learning

THE UNIVERSITY OF UTAH

Quantifying Performance

- How can we rigorously quantify the performance of our learning algorithm?

- **One approach**: Compute how many examples should the learning algorithm see before we can say that our learned hypothesis is *good* (or *good enough*)

This number will depend on the learning protocol

Example: Learning Conjunctions

There is a hidden (monotone) conjunction for the learner (you) to learn

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

There are 100 Boolean variables.
But you don't know that only these ***five*** are relevant

How many examples are needed to learn it? How does learning proceed?

Example: Learning Conjunctions

There is a hidden (monotone) conjunction for the learner (you) to learn

$$f = x_2 \land x_3 \land x_4 \land x_5 \land x_{100}$$

There are 100 Boolean variables.
But you don't know that only these ***five*** are relevant

How many examples are needed to learn it? How does learning proceed?

Active learning

– ***Protocol I***: The learner proposes instances as queries to the teacher

Example: Learning Conjunctions

There is a hidden (monotone) conjunction for the learner (you) to learn

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

There are 100 Boolean variables.
But you don't know that only these ***five*** are relevant

How many examples are needed to learn it? How does learning proceed?

Active learning – ***Protocol I***: The learner proposes instances as queries to the teacher

Teaching – ***Protocol II***: The teacher (who knows f) provides training examples

Example: Learning Conjunctions

There is a hidden (monotone) conjunction for the learner (you) to learn

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

There are 100 Boolean variables.
But you don't know that only these ***five*** are relevant

How many examples are needed to learn it? How does learning proceed?

Active learning
- ***Protocol I***: The learner proposes instances as queries to the teacher

Teaching
- ***Protocol II***: The teacher (who knows f) provides training examples

- ***Protocol III***: Some random source (e.g. Nature) provides training examples; the Teacher (Nature) provides the labels (f(x))

Example: Learning Conjunctions

There is a hidden (monotone) conjunction for the learner (you) to learn

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

There are 100 Boolean variables.
But you don't know that only these ***five*** are relevant

How many examples are needed to learn it? How does learning proceed?

- Active learning – ***Protocol I***: The learner proposes instances as queries to the teacher
- Teaching – ***Protocol II***: The teacher (who knows f) provides training examples
- ***Protocol III***: Some random source (e.g. Nature) provides training examples; the Teacher (Nature) provides the labels (f(x))

Let us compare these protocols.

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

If the learner can choose the queries wisely, then each query will force the teacher to reveal new information about the hidden function

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- Learner asks: What is the label for (1,1,1…,1,0)?

  Teacher's answer: $f(x) = 0$

  Learner's conclusion: Yes, x_{100} is in f

Note:
0 = `False`
1 = `True`

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- Learner asks: What is the label for (1,1,1…,1,0)?

  Teacher's answer: $f(x) = 0$

  Learner's conclusion: Yes, x_{100} is in f

Why? To learn whether x_{100} is in the true function, the learner can look at the label assigned to this specific instance

1. If the label is false, setting x_{100} to 0 makes the output 0. That is, x_{100} is part of the conjunction

2. If the label is true, setting x_{100} to 0 does not make the output 0. That is, x_{100} is not needed in the conjunction

$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- What is the label for (1,1,1...,1,0)? $f(x) = 0$ (conclusion: x_{100} is in f)
- What is the label for (1,1,...1,0,1)? $f(x) = 1$ (conclusion: x_{99} is not in f)

$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- What is the label for (1,1,1…,1,0)? $f(x) = 0$ (conclusion: x_{100} is in f)
- What is the label for (1,1,…1,0,1)? $f(x) = 1$ (conclusion: x_{99} is not in f)
- …

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- What is the label for (1,1,1…,1,0)? $f(x) = 0$ (conclusion: x_{100} is in f)
- What is the label for (1,1,…1,0,1)? $f(x) = 1$ (conclusion: x_{99} is not in f)
- …
- What is the label for (1,0,…1,1,1)? $f(x) = 0$ (conclusion: x_2 is in f)

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- What is the label for (1,1,1…,1,0)? $f(x) = 0$ (conclusion: x_{100} is in f)
- What is the label for (1,1,…1,0,1)? $f(x) = 1$ (conclusion: x_{99} is not in f)
- …
- What is the label for (1,0,…1,1,1)? $f(x) = 0$ (conclusion: x_2 is in f)
- What is the label for (0,1,…1,1,1)? $f(x) = 1$ (conclusion: x_1 is not in f)

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- What is the label for (1,1,1…,1,0)? $f(x) = 0$ (conclusion: x_{100} is in f)
- What is the label for (1,1,…1,0,1)? $f(x) = 1$ (conclusion: x_{99} is not in f)
- …
- What is the label for (1,0,…1,1,1)? $f(x) = 0$ (conclusion: x_2 is in f)
- What is the label for (0,1,…1,1,1)? $f(x) = 1$ (conclusion: x_1 is not in f)

A straight forward algorithm requires $n = 100$ queries, and will produce the hidden conjunction (exactly)

$$h = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol I: The learner proposes instances as queries to the teacher

Since we know we are after a monotone conjunction:

- What is the label for (1,1,1…,1,0)? $f(x) = 0$ (conclusion: x_{100} is in f)
- What is the label for (1,1,…1,0,1)? $f(x) = 1$ (conclusion: x_{99} is not in f)
- …
- What is the label for (1,0,…1,1,1)? $f(x) = 0$ (conclusion: x_2 is in f)
- What is the label for (0,1,…1,1,1)? $f(x) = 1$ (conclusion: x_1 is not in f)

A straight forward algorithm requires $n = 100$ queries, and will produce the hidden conjunction (exactly)

$$h = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

What happens here if the conjunction is not known to be monotone?

If we know of a positive example, a similar algorithm works.

Exercise: Verify this

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol II: The teacher (who knows f) provides training examples

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol II: The teacher (who knows f) provides training examples

- First: Teacher gives a superset of the good variables
  - $f(0,1,1,1,1,0,\ldots,0,1) = 1$

$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

Learning Conjunctions

Protocol II: The teacher (who knows f) provides training examples

- First: Teacher gives a superset of the good variables
  - $f(0,1,1,1,1,0,\ldots,0,1) = 1$

- Next: Teacher proves that each of these variables are required
  - $f(0,0,1,1,1,0,\ldots,0,1) = 0$ Conclusion: need x_2
  - $f(0,1,0,1,1,0,\ldots,0,1) = 0$ Conclusion: need x_3
  - $f(0,1,1,0,1,0,\ldots,0,1) = 0$ Conclusion: need x_4
  - $f(0,1,1,1,0,0,\ldots,0,1) = 0$ Conclusion: need x_5
  - $f(0,1,1,1,1,0,\ldots,0,0) = 0$ Conclusion: need x_{100}

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol II: The teacher (who knows f) provides training examples

- First: Teacher gives a superset of the good variables
  - $f(0,1,1,1,1,0, \ldots, 0,1) = 1$

These variables are sufficient

- Next: Teacher proves that each of these variables are required
  - $f(0,0,1,1,1,0, \ldots, 0,1) = 0$ Conclusion: need x_2
  - $f(0,1,0,1,1,0, \ldots, 0,1) = 0$ Conclusion: need x_3
  - $f(0,1,1,0,1,0, \ldots, 0,1) = 0$ Conclusion: need x_4
  - $f(0,1,1,1,0,0, \ldots, 0,1) = 0$ Conclusion: need x_5
  - $f(0,1,1,1,1,0, \ldots, 0,0) = 0$ Conclusion: need x_{100}

All the variables are necessary

$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

Learning Conjunctions

Protocol II: The teacher (who knows f) provides training examples

- First: Teacher gives a superset of the good variables
  - $f(0,1,1,1,1,0,\ldots,0,1) = 1$

- Next: Teacher proves that each of these variables are required
  - $f(0,0,1,1,1,0,\ldots,0,1) = 0$ Conclusion: need x_2
  - $f(0,1,0,1,1,0,\ldots,0,1) = 0$ Conclusion: need x_3
  - …
  - $f(0,1,1,1,1,0,\ldots,0,0) = 0$ Conclusion: need x_{100}

A straight forward algorithm requires k = 6 examples to produce the hidden conjunction (exactly)

$h = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

Learning Conjunctions

Protocol II: The teacher (who knows f) provides training examples

- First: Teacher gives a superset of the good variables
  - $f(0,1,1,1,1,0,\ldots,0,1) = 1$
- Next: Teacher proves that each of these variables are required
  - $f(0,0,1,1,1,0,\ldots,0,1) = 0$ Conclusion: need x_2
  - $f(0,1,0,1,1,0,\ldots,0,1) = 0$ Conclusion: need x_3
  - …
  - $f(0,1,1,1,1,0,\ldots,0,0) = 0$ Conclusion: need x_{100}

Modeling teaching can be very difficult, unfortunately

A straight forward algorithm requires k = 6 examples to produce the hidden conjunction (exactly)

$h = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol III: Some random source (nature) provides training examples

Teacher (Nature) provides the labels (f(x))

- <(1,1,1,1,1,1,…,1,1), 1>
- <(1,1,1,0,0,0,…,0,0), 0>
- <(1,1,1,1,1,0,...0,1,1), 1>
- <(1,0,1,1,1,0,...0,1,1), 0>
- <(1,1,1,1,1,0,...0,0,1), 1>
- <(1,0,1,0,0,0,...0,1,1), 0>
- <(1,1,1,1,1,1,…,0,1), 1>
- <(0,1,0,1,0,0,...0,1,1), 0>

Notation: <example, label>

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol III: Some random source (nature) provides training examples

Teacher (Nature) provides the labels (f(x))

- <(1,1,1,1,1,1,…,1,1), 1>
- <(1,1,1,0,0,0,…,0,0), 0>
- <(1,1,1,1,1,0,...0,1,1), 1>
- <(1,0,1,1,1,0,...0,1,1), 0>
- <(1,1,1,1,1,0,...0,0,1), 1>
- <(1,0,1,0,0,0,...0,1,1), 0>
- <(1,1,1,1,1,1,…,0,1), 1>
- <(0,1,0,1,0,0,...0,1,1), 0>

Look for the variables that are present in *every* positive example.

All other variables can be eliminated

Why?

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol III: Some random source (nature) provides training examples

Teacher (Nature) provides the labels (f(x))

- <(1,1,1,1,1,1,…,1,1), 1>
- <(1,1,1,0,0,0,…,0,0), 0>
- <(1,1,1,1,1,0,...0,1,1), 1>
- <(1,0,1,1,1,0,...0,1,1), 0>
- <(1,1,1,1,1,0,...0,0,1), 1>
- <(1,0,1,0,0,0,...0,1,1), 0>
- <(1,1,1,1,1,1,…,0,1), 1>
- <(0,1,0,1,0,0,...0,1,1), 0>

For a reasonable learning algorithm (by *elimination*), the final hypothesis will be

$$h = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol III: Some random source (nature) provides training examples

Teacher (Nature) provides the labels (f(x))

- <(1,1,1,1,1,1,...,1,1), 1>
- <(1,1,1,0,0,0,...,0,0), 0>
- <(1,1,1,1,1,0,...0,1,1), 1>
- <(1,0,1,1,1,0,...0,1,1), 0>
- <(1,1,1,1,1,0,...0,0,1), 1>
- <(1,0,1,0,0,0,...0,1,1), 0>
- <(1,1,1,1,1,1,...,0,1), 1>
- <(0,1,0,1,0,0,...0,1,1), 0>

For a reasonable learning algorithm (by *elimination*), the final hypothesis will be

$$h = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Whenever the output is 1, x_1 is present

$$f = x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Learning Conjunctions

Protocol III: Some random source (nature) provides training examples

Teacher (Nature) provides the labels (f(x))

- <(1,1,1,1,1,1,...,1,1), 1>
- <(1,1,1,0,0,0,...,0,0), 0>
- <(1,1,1,1,1,0,...0,1,1), 1>
- <(1,0,1,1,1,0,...0,1,1), 0>
- <(1,1,1,1,1,0,...0,0,1), 1>
- <(1,0,1,0,0,0,...0,1,1), 0>
- <(1,1,1,1,1,1,...,0,1), 1>
- <(0,1,0,1,0,0,...0,1,1), 0>

For a reasonable learning algorithm (by *elimination*), the final hypothesis will be

$$h = x_1 \wedge x_2 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_{100}$$

Whenever the output is 1, x_1 is present

With the given data, we only learned an *approximation* to the true concept.
Is it good enough?

Two Directions for How good is our learning algorithm?

- Can analyze the probabilistic intuition
  - Never saw x_1=0 in positive examples, maybe we'll never see it
  - And if we do, it will be with small probability, so the concepts we learn may be *pretty good*
    - *Pretty good:* In terms of performance on future data
  - ***PAC framework***

Two Directions for How good is our learning algorithm?

- Can analyze the probabilistic intuition
  - Never saw x_1=0 in positive examples, maybe we'll never see it
  - And if we do, it will be with small probability, so the concepts we learn may be *pretty good*
    - *Pretty good:* In terms of performance on future data
  - ***PAC framework***

- ***Mistake Driven*** Learning algorithms
  - Update your hypothesis only when you make mistakes
  - Define *good* in terms of how many mistakes you make before you stop