CS4710: Artificial Intelligence
Intro to Machine Learning

Intelligent beings (e.g., humans) are excellent at learning. How can we try to program systems to learn interesting new things?

Topics

- Overview of Learning
  - Types of learning, etc.
- Supervised Learning
  - Induction Learning
  - Decision Trees
  - Regression / Gradient Descent
  - Logistic Regression

Introduction

Learning

- What are some different ways in which human beings learn?

- We will focus on these:
  - Rote Learning
  - Learning by example / recognizing patterns
  - Learning by observing similarities
  - Learning by practice

Rote Learning

- Memorizing what is learned
  - Difficult to transfer knowledge to different domains or different problems
  - Easy (ish) to recall the information
  - Hard to apply the information
- Computers are GREAT at this
  - String knowledge = “America declared independence in 1776”
  - The computer is learning!!!!
  - …but it can’t DO anything with that knowledge

Learning by Doing

- Learn to do something by practicing and retrying until you get better
  - Painting, math, sports, etc.

- How can computers do this?
- This is called **Reinforcement Learning** and is a well-studied area

Learning by Doing

- Reinforcement Learning Models usually contain (at a minimum):
  - A concept of reward
  - A set of actions an agent can take
  - A set of ways to observe the environment
  - Sound familiar? Can be modeled with Markov Decision Processes

Reinforcement Learning: Example

- Suppose we want an agent that plays Mario
- A concept of reward
  - Going to the right = reward (maximize this)
- A set of actions an agent can take
  - Buttons on the controller (jump, etc.)
- A set of ways to observe the environment
  - Depends. What are the options here?

Learning by Example

- If I show you a bunch of examples of a new species of animal, you'll know how to recognize that animal
- I didn't explicitly teach you anything, so how did you do it?
- This is called **Supervised Learning** because a supervisor (teacher, etc.) is telling you the answer to many examples
- Then, hopefully you can perform the task independently afterward

Learning by Example

- This is called **Supervised Learning** because a supervisor (teacher, etc.) is telling you the answer to many examples

- For supervised learning, we will need:
  - Many examples with the correct answer!
  - A set of characteristics to look for in each example
    - E.g., looking at cardinals, we notice the male cardinals are red and the females are green'ish
    - We call these **features**

Learning by Observing Similarities

- What if we are looking at pictures of animals, but aren't given the ground truth of what animal is in which picture?
- We can still group similar photos together
  - E.g., these are all birds, these are all green, etc.
- This is called clustering, and is a form of **unsupervised learning**.

Supervised Learning

Supervised Learning

- Sometimes also called **classification** if you are placing examples into discrete classes
- We are given a bunch of data points, each one with the ground truth classification that we care about
- **Goal**: Build a model that allows us to accurately predict new data points not in the original set

Example: Predicting Grades

Student	A last year?	Male?	Works Hard?	Drinks?	A this year?
Richard	Yes	Yes	No	Yes	No
Allen	Yes	Yes	Yes	No	Yes
Alison	No	No	Yes	No	No
Jeff	No	Yes	No	Yes	No
Gail	Yes	No	Yes	Yes	Yes
Simon	No	Yes	Yes	Yes	No

Supervised Learning

- The data we are given (previous slide as example) is called a **training set**
- After training your model, you can run it on new data called the **test set** to judge the accuracy of your model
- …we'll see more of this later in a moment

Example: Predicting Grades

Student	A last year?	Male?	Works Hard?	Drinks?	A this year?
Richard	Yes	Yes	No	Yes	No
Allen	Yes	Yes	Yes	No	Yes
Alison	No	No	Yes	No	No
Jeff	No	Yes	No	Yes	No
Gail	Yes	No	Yes	Yes	Yes
Simon	No	Yes	Yes	Yes	No

- Some possible learned rules?
- You will get an A this year if:
  - A Last Year && Work Hard
  - (Male && Don't Drink) || (Female && Drink)
- Will these hold up if given a new test set? Probably not…

Version Space Learning

Version Space Learning

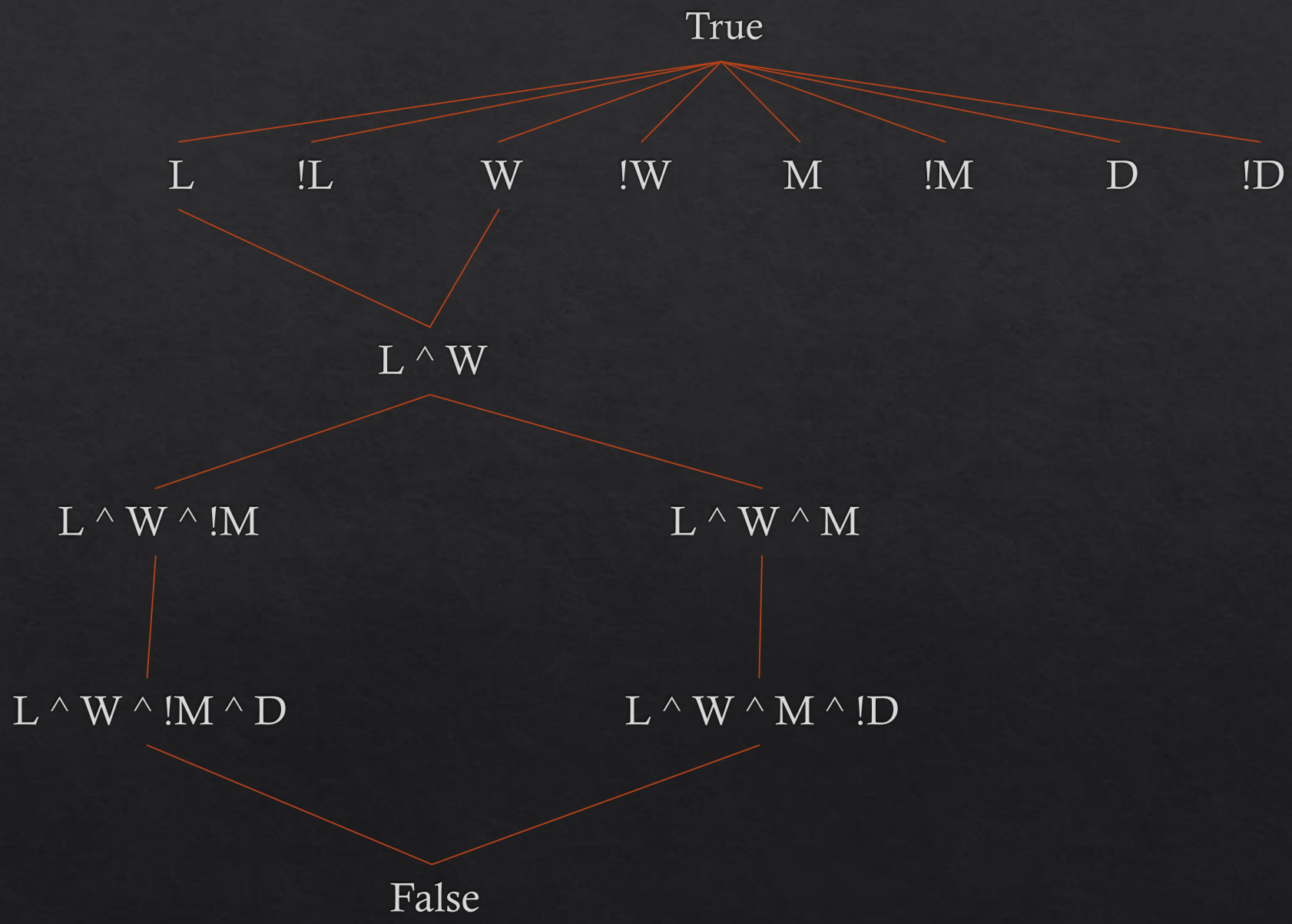
- Our first (very basic) learning algorithm
- Look at data and learn rules as conjunctions
  - I.e., you will get an ‘A’ if you got an ‘A’ last year and you work hard
- Treats learning as a search problem
- Very simple method (compared to others)

Example

Student	A last year?	Male?	Works Hard?	Drinks?	A this year?
Richard	Yes	Yes	No	Yes	No
Allen	Yes	Yes	Yes	No	Yes
Alison	No	No	Yes	No	No
Jeff	No	Yes	No	Yes	No
Gail	Yes	No	Yes	Yes	Yes
Simon	No	Yes	Yes	Yes	No

- Treat given variables as true false only:
  - L = "Got an A last year"
  - M = "Is a Male"
  - W = "Works hard"
  - D = "Drinks a lot"
- Learn rules of the form:
  - L ^ W

Search Tree



Algorithm

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Current Example E:

NONE

MEMORY:

G = {T}

S = F

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Current Example E:

T T F T Result: F

**From Richard's row in table*

MEMORY:

G = {T}

S = F

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Current Example E:

T T F T Result: F

**From Richard's row in table*

MEMORY:

G = {¬L, ¬M, W, ¬D}

S = F

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Current Example E:

T T T F Result: T

MEMORY:

G = {¬L, ¬M, W, ¬D}

S = F

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Current Example E:

T T T F Result: T

MEMORY:

G = {¬L, ¬M, W, ¬D}

S = L ^ M ^ W ^ ¬D

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Current Example E:

T T T F Result: T

MEMORY:

G = {W, ¬D}

S = L ^ M ^ W ^ ¬D

**What have we learned so far?*

- *Most general rule is Work Hard and don't drink always leads to an A*
- *Most specific rule?*
- *Other rules fall between these two*

Example:

```
Initialize G = {T}, S = F
For each example E{
    If (E is positive example){
        If (S is false for E)
            S := first formula UP tree that is true for E
        Delete elements in G not true for E
    }
    If (E is negative example){
        If (Any formula F is true for E and in G)
            F := first formula DOWN tree that is false for G
        Delete elements in G more specific than S
    }
}
```

Problems with this?

When does it work well?

When does it not work well?