Machine Learning: Part I

UMaine COS 470/570 – Introduction to AI

Spring 2019

Introduction

Why learn?

Why learn?

- Why not static agent/knowledge?

Why learn?

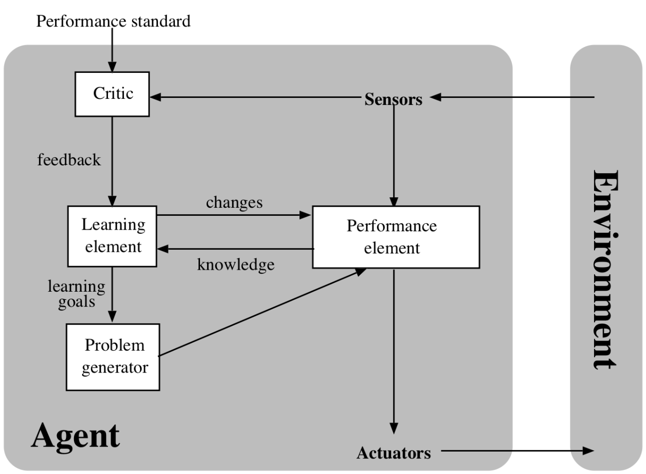
- Why not static agent/knowledge?
- Changing world
- Different tasks
- Bad initial design/knowledge
- Incomplete/uncertain knowledge of world

Artificial Intelligence

Learning agents

Machine Learning: Part I
Introduction
Why learn?
Learning agents
Kinds of learning?
Supervised learning: Induction
Unsupervised learning
Reinforcement learning
"Classical" ML

Agents and learning



Machine Learning: Part I
Introduction
Why learn?
Learning agents
Kinds of learning?
Supervised learning: Induction
Unsupervised learning
Reinforcement learning
"Classical" ML

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Kinds of learning?

Machine Learning: Part I
Introduction
Why learn?
Learning agents
Kinds of learning?
Supervised learning: Induction
Unsupervised learning
Reinforcement learning
"Classical" ML

What are the kinds of learning?

- Feedback available: supervised vs unsupervised vs reinforcement
- What's learned: axioms, rules, plans, weights, . . .
- Symbolic vs sub-symbolic (neural, GA, LCS)
- Ensemble learning
- PAC learning

Supervised learning: Induction

Machine Learning: Part I

Supervised learning: Induction

- Decision-tree learning
- Bayesian learning
- Decision-theoretic
- Neural network approaches

Machine Learning: Part I

Unsupervised learning

Machine Learning: Part I

Unsupervised learning

- Explanation-based learning
- Regression
- Induction: autoencoders, neural networks, support vector machines (SVMs)

Artificial Intelligence

Reinforcement learning

Reinforcement learning

- Policy learning: P(state)$\rightarrow$ action
- Q-learning: Q(action, state) $\rightarrow$ value (utility)
- Case-based reasoning
- GAs, LCSs
- Deep RL

Artificial Intelligence

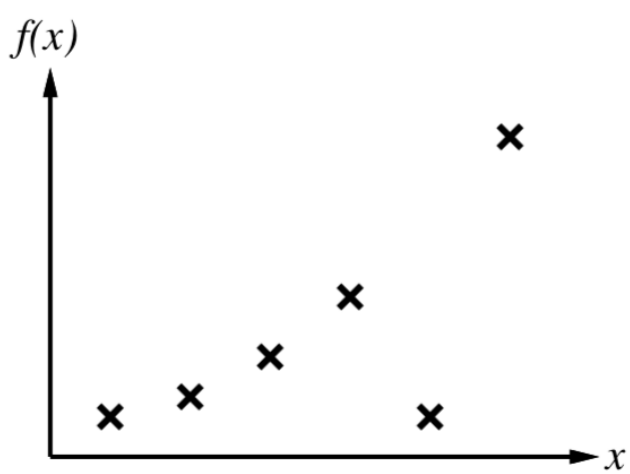
"Classical" ML

Induction

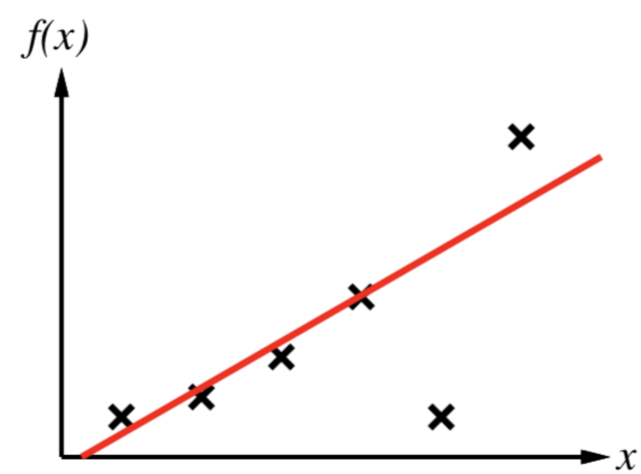
Inductive learning in general

- Problem:
  - Given n examples of $(x, f(x))$ pairs
  - Find $f(y)$ for some new example y
- Approach: Find $h(x)$, an approximation to $f(x)$

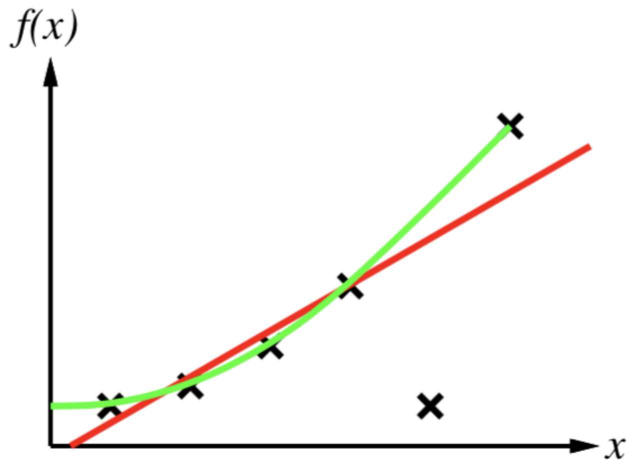
Example: Regression (curve-fitting)



Example: Regression (curve-fitting)

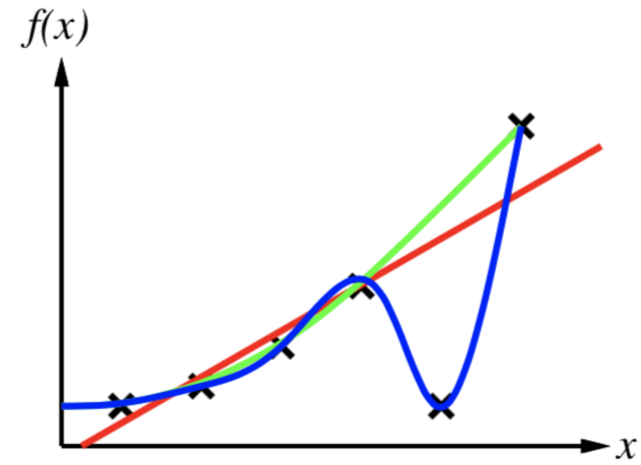


Example: Regression (curve-fitting)



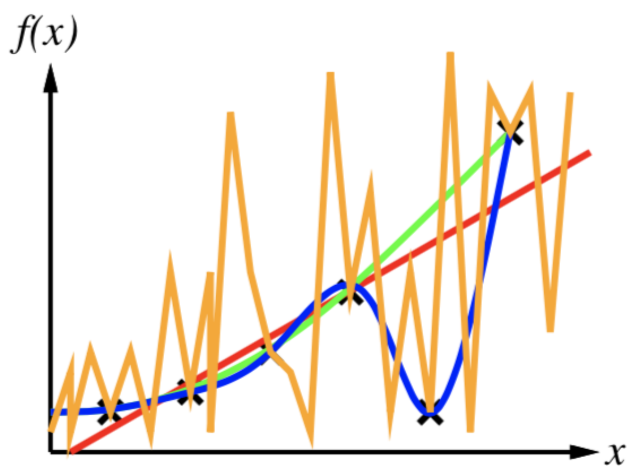
Artificial Intelligence

Example: Regression (curve-fitting)



Artificial Intelligence

Example: Regression (curve-fitting)



Artificial Intelligence

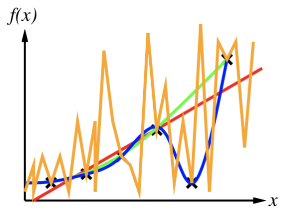
Hypothesis space

- *Hypothesis space* contains all the h you are considering
  - For regression: lines, polynomials, exponentials, ...
  - Other example: weight space of a particular neural network architecture
- Selection critical – needs to contain $f(x)$ or contain good-enough approximation(s)
- *Consistent hypothesis*: agrees with all the data (to some ϵ, perhaps)
- Maybe infinite # of consistent hypotheses
- Use Occam's (Ockham's) razor

Artificial Intelligence

Hypothesis space

- *Hypothesis space* contains all the h you are considering
  - For regression: lines, polynomials, exponentials, ...
  - Other example: weight space of a particular neural network architecture
- Selection critical – needs to contain $f(x)$ or contain good-enough approximation(s)
- *Consistent hypothesis*: agrees with all the data (to some ϵ, perhaps)
- Maybe infinite # of consistent hypotheses
- Use Occam's (Ockham's) razor



Artificial Intelligence

What if there's no consistent function?

Machine Learning: Part I
Introduction
“Classical” ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Wrong hypothesis space (problem is *unrealizable* in space)
- $f(x)$ not deterministic
- Measurement errors for $(x, f(x))$ pairs

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Performance measurement

Machine Learning: Part I
Introduction
“Classical” ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- How do we know h sufficiently approximates f (Hume: *Problem of Induction*)
- Usual approach:
  - Try h on some new test set T of examples
  - T should have same distribution over example space as training set
  - Often: break initial example set into training, test subsets
- Estimate accuracy directly or create *learning curve*
- Learning curve = %correct as function of training set size

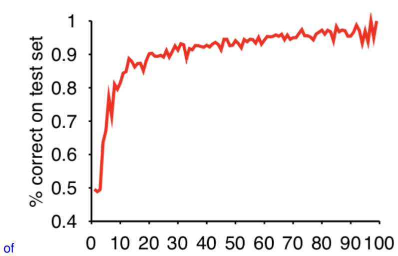
Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Performance measurement

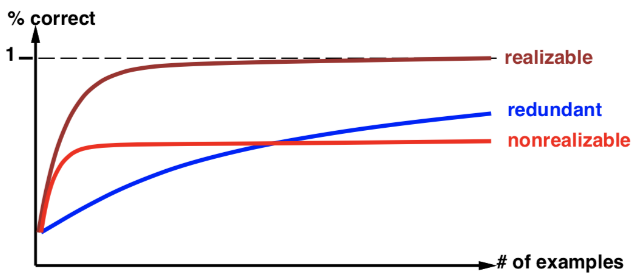
Machine Learning: Part I
Introduction
“Classical” ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- How do we know h sufficiently approximates f (Hume: *Problem of Induction*)
- Usual approach:
  - Try h on some new test set T of examples
  - T should have same distribution over example space as training set
  - Often: break initial example set into training, test subsets
- Estimate accuracy directly or create *learning curve*
- Learning curve = %correct as function of training set size



Artificial Intelligence

Learning curves



Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Decision tree learning

Classification from examples

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Problem: given a set of properties, find or answer to question
- Given set of examplars: sets of features → answers
- Supervised learning

Copyright © 2019 UMaine School of Computing and Information Science

Approach

Machine Learning: Part I

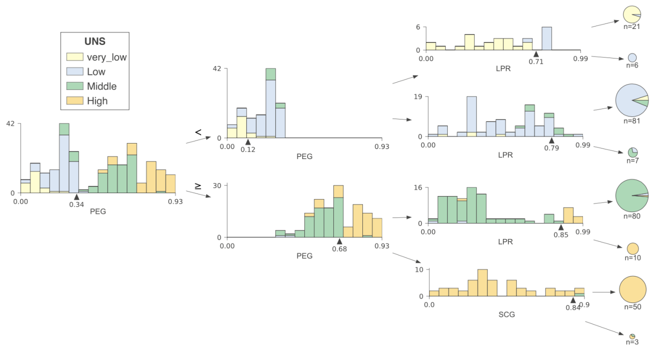
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Identify a feature that cleanly divides the exemplars into subsets
- For each subset, do it again
- Continue until each subset contain indistinguishable exemplars
- Each split ≡ question about new thing to categorize
- E.g., something like Akinator
- Result is a *decision tree*
- Everyday examples: scientific species key, repair manual, etc.

Copyright © 2019 UMaine School of Computing and Information Science

Example: User knowledge

- Classify user knowledge in some domain (UNS)



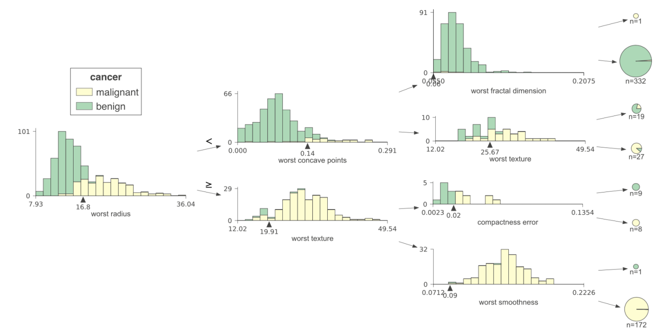
(STG = study time for goal objects, SCG = repititions, STR = study time for related objects, LPR = exam performance for related objects, PEG = exam performance for goal objects)

(from UCI machine learning repository, explained.ai)

Artificial Intelligence

Example: Breast cancer

Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning



(from UCI ML Breast Cancer Wisconsin dataset, explained.ai)

Artificial Intelligence

Learning a decision tree

Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Suppose we have a data set described by *attribute values*
- E.g., to answer: "Should I wait for a table?" (T/F categorization)
- Attributes:

1. *Alternate*: whether there is a suitable alternative restaurant nearby.
2. *Bar*: whether the restaurant has a comfortable bar area to wait in.
3. *Fri/Sat*: true on Fridays and Saturdays.
4. *Hungry*: whether we are hungry.
5. *Patrons*: how many people are in the restaurant (values are *None*, *Some*, and *Full*).
6. *Price*: the restaurant's price range ($, $$, $$$).
7. *Raining*: whether it is raining outside.
8. *Reservation*: whether we made a reservation.
9. *Type*: the kind of restaurant (French, Italian, Thai, or burger).
10. *WaitEstimate*: the wait estimated by the host (0–10 minutes, 10–30, 30–60, or >60).

Artificial Intelligence

Learning a decision tree

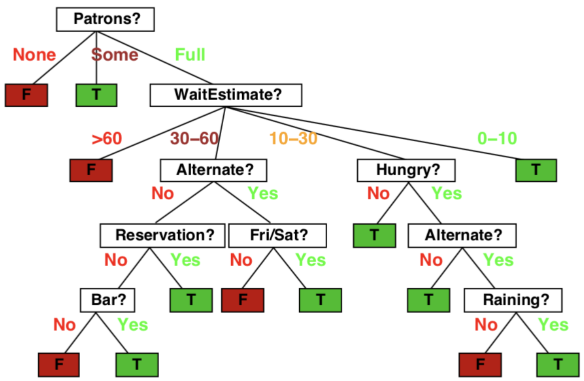
Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Have examples:

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*T*	*F*	*F*	*T*	*Some*	*$$$*	*F*	*T*	*French*	*0–10*	*T*
X_2	*T*	*F*	*F*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*30–60*	*F*
X_3	*F*	*T*	*F*	*F*	*Some*	*$*	*F*	*F*	*Burger*	*0–10*	*T*
X_4	*T*	*F*	*T*	*T*	*Full*	*$*	*F*	*F*	*Thai*	*10–30*	*T*
X_5	*T*	*F*	*T*	*F*	*Full*	*$$$*	*F*	*T*	*French*	*>60*	*F*
X_6	*F*	*T*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Italian*	*0–10*	*T*
X_7	*F*	*T*	*F*	*F*	*None*	*$*	*T*	*F*	*Burger*	*0–10*	*F*
X_8	*F*	*F*	*F*	*T*	*Some*	*$$*	*T*	*T*	*Thai*	*0–10*	*T*
X_9	*F*	*T*	*T*	*F*	*Full*	*$*	*T*	*F*	*Burger*	*>60*	*F*
X_{10}	*T*	*T*	*T*	*T*	*Full*	*$$$*	*F*	*T*	*Italian*	*10–30*	*F*
X_{11}	*F*	*F*	*F*	*F*	*None*	*$*	*F*	*F*	*Thai*	*0–10*	*F*
X_{12}	*T*	*T*	*T*	*T*	*Full*	*$*	*F*	*F*	*Burger*	*30–60*	*T*

Artificial Intelligence

Possible tree



Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Expressiveness

- Decision trees: can express any function of the attributes
- E.g., equiv. to binary satisfiability for Boolean functions
  - Think of a truth table
  - Internal nodes = input variables
- Can find a consistent tree for any training set
  - But trivially, won't generalize –
  - Want a more compact tree
  - Decide on attribute to include next based on amount of *information* they provide

Copyright © 2019 UMaine School of Computing and Information Science

Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Decision tree learning algorithm

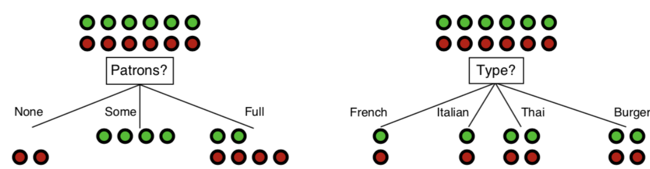
```
function DTL(examples, attributes, default) returns a decision tree
   if examples is empty then return default
   else if all examples have the same classification then return the classification
   else if attributes is empty then return MODE(examples)
   else
      best ← CHOOSE-ATTRIBUTE(attributes, examples)
      tree ← a new decision tree with root test best
      for each value v_i of best do
         examples_i ← {elements of examples with best = v_i}
         subtree ← DTL(examples_i, attributes − best, MODE(examples))
         add a branch to tree with label v_i and subtree subtree
      return tree
```

Copyright © 2019 UMaine School of Computing and Information Science

Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Which attribute to choose?

- Good attribute splits into (ideally) all pos or all neg examples



Patrons? is a better choice—gives **information** about the classification

Copyright © 2019 UMaine School of Computing and Information Science

Which attribute to choose?

- Consider attributes as questions to be asked
- Choose one that gives more information when answered
- Less I initially know, more information answer provides
- Measure information in *bits*
- Need 1 bit to answer yes/no question with prior prob. = <0.5, 0.5>
- If prior = <P_1, P_2, . . . , P_n>, information is the *entropy* of the prior:

$$H(< P_1, P_2, \dots, P_n >) = \sum_{i=1}^{n} -P_i \log_2 P_i$$

Which attribute to choose?

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Suppose we have p pos and n neg examples
- $H(< p/(p+n), n/(p+n) >)$ bits needed to classify new example
- Each attribute i splits examples E into subgroup E_i
- Hopefully, each E_i needs less information to complete than initial problem
- If E_i has p_i pos and n_i neg examples, then need:

  $$H(< p_i/(p_i+n_i), n_i/(p_i+n_i) >)$$

  bits to classify
- *Expected* bits per example over all branches

  $$H_E = \Sigma_i \frac{p_i+n_i}{p+n} H(< p_i/(p_i+n_i), n_i/(p_i+n_i) >)$$

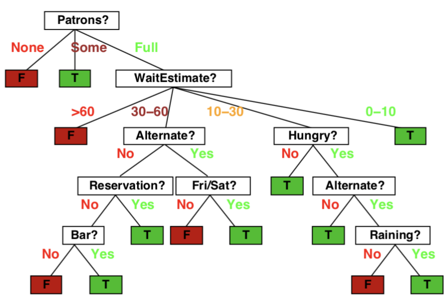
- Prev example: $H_E(Patrons?) = 0.459$ bits, $H_E(Type) = 1$ bit
- $\therefore$ choose *Patrons*?

Result of using information gain

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Russell's

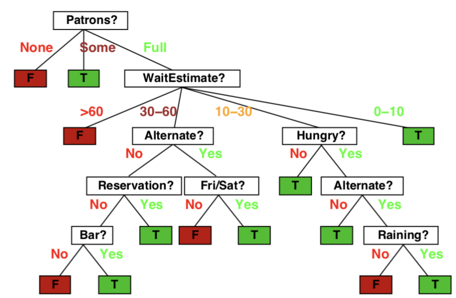


Result of using information gain

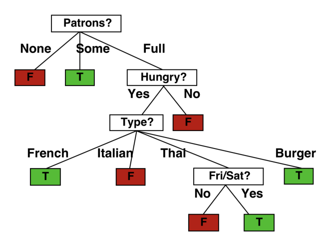
Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Russell's

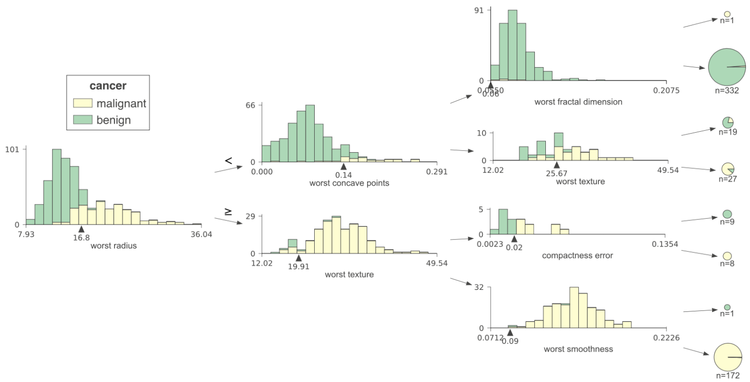


Learned tree:



Drawbacks

- Leaf nodes with multiple examples may not agree on classification



Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Drawbacks

- Overfitting
  - Even if critical info is missing, DT alg. will create *some* DTree
  - Can make spurious distinctions based on irrelevant attribs
    - E.g., predict roll of die
    - Attributes given: hair color, color of die, day of the week, etc.
    - If no duplicate descriptions for different outcomes ⇒ exact hypothesis
  - As number of attribute goes up, so does likelihood of overfitting

Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Drawbacks

- Decision tree learning space
  - With n Boolean attributes, tree is equiv. to truth table
  - So 2^n rows
  - Space of all possible n-attribute decision trees = number of n-variable Boolean functions
  - There are 2^n rows, same as the answer column
  - Thus there are 2^{2^n} decision trees
  - Russell & Norvig: "scary number"

Machine Learning: Part I
Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Drawbacks

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$

Drawbacks

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$

Artificial Intelligence

Drawbacks

Machine Learning: Part I

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Drawbacks

Machine Learning: Part I

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$
4	$2^{16} = 65{,}536$

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Drawbacks

Machine Learning: Part I

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$
4	$2^{16} = 65{,}536$
5	$2^{32} = 4{,}294{,}967{,}296$

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Drawbacks

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$
4	$2^{16} = 65{,}536$
5	$2^{32} = 4{,}294{,}967{,}296$
6	$2^{64} = 18{,}446{,}744{,}073{,}709{,}551{,}616$

Artificial Intelligence

Drawbacks

Machine Learning: Part I

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$
4	$2^{16} = 65{,}536$
5	$2^{32} = 4{,}294{,}967{,}296$
6	$2^{64} = 18{,}446{,}744{,}073{,}709{,}551{,}616$
7	~3.4E38

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Drawbacks

Machine Learning: Part I

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$
4	$2^{16} = 65{,}536$
5	$2^{32} = 4{,}294{,}967{,}296$
6	$2^{64} = 18{,}446{,}744{,}073{,}709{,}551{,}616$
7	~3.4E38
8	~1.2E77

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Drawbacks

Machine Learning: Part I

- Decision tree learning space (cont'd)

#attributes	# poss. decision trees
1	$2^2 = 4$
2	$2^4 = 16$
3	$2^8 = 256$
4	$2^{16} = 65{,}536$
5	$2^{32} = 4{,}294{,}967{,}296$
6	$2^{64} = 18{,}446{,}744{,}073{,}709{,}551{,}616$
7	~3.4E38
8	~1.2E77

- R&N: "We will need some ingenious algorithms to find consistent hypotheses in such a large space."

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence

Recap/Animation

From: A visual introduction to machine learning

Explanation-based learning

Explanation-based learning

- Explanation-based learning (EBL): use prior domain knowledge to learn new facts
- Traditionally (Mooney, DeJong, others) – no statistical inference, no induction
- Given a domain theory Δ and some example X:
  - Use Δ to account for X – e.g., show that $\Delta \models X$
  - Then convert reasoning chain to a new rule

EBL example

- Suppose we know people want money, people value things they love, people will pay to have things they value, . . .
- We learn that Johnny took Peter, Peter is Sally's husband, and Sally paid Johnny \$100,000
- Explain this by such things as: Sally loves Peter, Sally values Peter, Sally pays Johnny to have Peter, etc.
- Ultimately, compress into rule or schema for kidnapping

EBL

- Can be seen more as *operationalizing* existing knowledge
- Problem: no real new knowledge

Artificial Intelligence

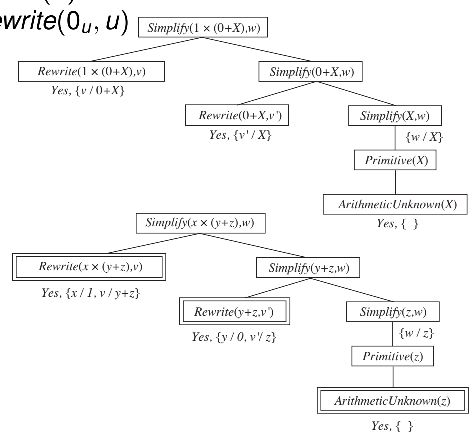
EBL example

(From R&N)
$Rewrite(u, v) \wedge Simplify(v, w) \Rightarrow Simplify(u, w)$
$Primitive(u) \Rightarrow Simplify(u, u)$
$ArithmeticUnknown(u) \Rightarrow Primitive(u)$
$Number(u) \Rightarrow Primitive(u)$
$Rewrite(1 \times u, u)$ $Rewrite(0_u, u)$
...

Artificial Intelligence

EBL example

(From R&N)
$Rewrite(u, v) \wedge Simplify(v, w) \Rightarrow Simplify(u, w)$
$Primitive(u) \Rightarrow Simplify(u, u)$
$ArithmeticUnknown(u) \Rightarrow Primitive(u)$
$Number(u) \Rightarrow Primitive(u)$
$Rewrite(1 \times u, u)$ $Rewrite(0_u, u)$
...



Artificial Intelligence

EBL example

- Can read bottom tree's leaves to extract the (generalized) rule:

$$Rewrite(1 \times (0+z), 0+z) \wedge Rewrite(0+z, z) \wedge$$
$$ArithmeticUnknown(z) \Rightarrow Simplify(1 \times (0+z), z)$$

- Two rightmost terms: true no matter what z is
- $ArithmeticUnknown(z)$ – not every z is an arithmetic unknown, so can't drop it
- Resulting rule:

$$ArithmeticUnknown(z) \Rightarrow Simplify(1 \times (0+z), z)$$

Artificial Intelligence

Learning, expertise, and soundness

(See DeJong's (2005) explanation-based learning paper, distributed on Slack)

- Problem:
  - Statistical learning (induction) :
    - unsound
    - can't readily make use of expert's knowledge
  - EBL:
    - sound
    - can easily incorporate expert's knowledge
    - but only operationalizes the knowledge
- DeJong: use EBL to combine expert knowledge with (some) statistical learning

Explanation-based learning and expertise

- Obtain knowledge from expert, encode in representation
- Problem: Expert knowledge approximate, conflicting
- Problem: If represented in something like FOPC, doesn't apply to real world:

$$\forall x \mathrm{Bird}(x) \Rightarrow \mathrm{Flies}(x)$$

  Statement not so much false as *incomplete*
- Almost *all* logical statements are incomplete with respect to the real world
  - Only sound with respect to some axiom set Δ
  - In general, subject to the *qualification problem*:
    - Most universally quantified sentences will have to include an infinite number of qualifications if they are to be interpreted as accurate statements about the world.
- E.g.: $\forall x \mathrm{Bird}(x) \wedge \neg \mathrm{Dead}(x) \wedge \cdots \Rightarrow \mathrm{Flies}(x)$

Explanation-based learning and expertise

- Proposal: accept expert's knowledge Δ as correct, but not believed
- Belief only comes from seeing examples that Δ can explain
- Learn new rules based on Δ and experience
- E.g.:
  - Δ: birds fly, flying involves volition, dead things have no volition, etc.
  - New example: cooked chicken – refutes birds fly
  - Rule still considered correct (from expert)
  - Explain not flying by proof relying on the chicken being dead
  - Hypothesize: $\forall x \mathrm{Dead}(x) \Rightarrow \neg Flies(x)$
  - Look for more examples to confirm this
  - Confirmed to acceptable level of belief: create new rule:

$$\forall x \mathrm{Bird}(x) \wedge \neg \mathrm{Dead}(x) \Rightarrow \mathrm{Flies}(x)$$

How to derive the new rule?

- New inference rule – AND introduction

$$\frac{\begin{array}{rl} \rho_1 & \Rightarrow \psi \\ \rho_2 & \Rightarrow \psi \end{array}}{(\rho_1 \wedge \rho_2) \quad \Rightarrow \psi}$$

  - Not logically necessary – but if $\exists$ weak evidence for ρ_1 and ρ_2, together might be sufficient to conclude ψ.
- Second rule:

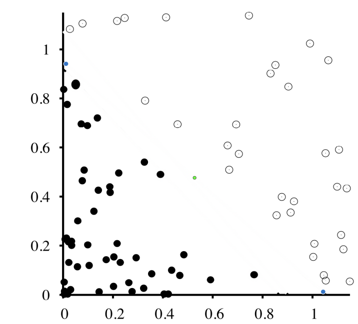
$$\frac{\begin{array}{rl} \rho_1 & \Rightarrow \psi \\ \rho_2 & \Rightarrow \neg\psi \end{array}}{(\rho_1 \wedge \neg\rho_2) \quad \Rightarrow \psi}$$

  - Again, logically redundant
  - However, very useful for conditioning knowledge for real world

Support vector machines

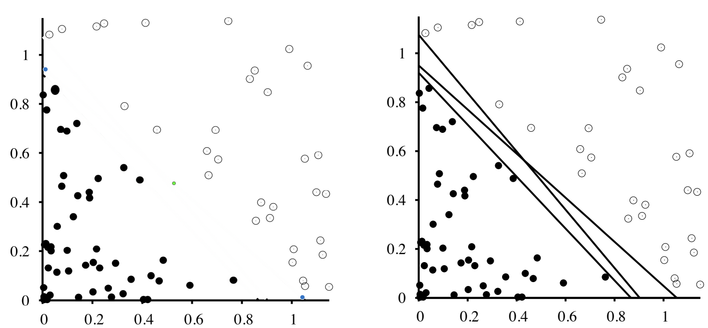
Support vector machines

- Problem: How to categorize examples of something?



Support vector machines

- Problem: How to categorize examples of something?

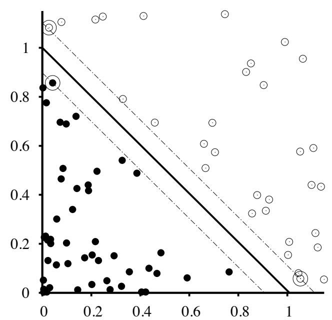


Support vector machines

- Maximize ability to generalize by picking something in the middle
- *Maximum margin separator*

Support vector machines

- Maximize ability to generalize by picking something in the middle
- *Maximum margin separator*

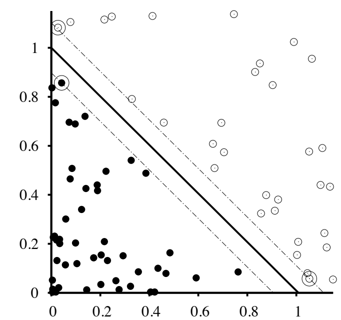


Support vector machines

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Maximize ability to generalize by picking something in the middle
- *Maximum margin separator*



- Mathematical formula (in book) to find this
- *Support vectors*: the points closest to the separator

SVMs

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- What if data no linearly-separable?
- Change dimensionality of data: map to a different space
- Look for *hyperplane* that cleanly separates data in that space
- How?
  - Formula uses dot product of vectors to data points to find maximal separator
  - Replace each vector x with a higher-dimensional vector F(x)
  - Dot products of these yield a *kernel function*: e.g.:

$$\vec{F}(\vec{x}_j) \cdot \vec{F}(\vec{x}_k) = (\vec{x}_j \cdot \vec{x}_k)^2$$

- Different kernel function $\rightarrow$ different mappings
- With n data points, can always separate in $\geq n-1$ dimensions

Example

SVMs: What are they good for?

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Categorizing images
- Detecting possible cancerous lesions (e.g., B. Toner's work on breast cancer)
- Basically the same kinds of things that artificial neural networks (ANNs) are good for

So... SVNs or ANNs?

- SVMs are non-parametric: meaning, size of model can grow with new data
- ANNs are parametric: number of weights fixed
- However, there are parametric variants of SVMs...
- ....and non-parametric versions of ANNs (if we include neuroevolutionary algorithms)
- ANNs discover compact representations of input – useful for other things
- But SVMs don't overfit, like ANNs do
- Not clear – perhaps a religious war? Need data/theory!

Genetic algorithms

Genetic algorithms

- Biological evolution – can be viewed as:
  - a search process
  - a learning process
- Characteristics:
  - Unsupervised
  - Highly parallel
  - Stochastic
  - Feedback: *fitness*
- Use ideas from evolution for AI: *genetic algorithms*

Example: Match a sentence

- Initially, output is random
- Feedback: measure of how close output is to target
- Use biological-like operations (e.g., mutation, reproduction) to evolve better solutions
- Many generations to get to correct/good enough answer

Example

GA is a *local search*

- Recall from CSP:
  - Didn't use a complete state representation in search
  - I.e., didn't start with state $\{v_1 = val_i, v_2 = val_j, \cdots\}$
  - Reason: Search space was *factorial* time exponential
  - But what if we can prune search effectively?

Local search

- Start with complete candidate solution
- If not solution (or within ϵ of solution): change some small part of state
- Different changes ⇒ different *neighborhoods*
  - Defined by operators sometimes (gen. alg., e.g.)
  - Defined by problem other times (e.g., BSAT)
- Choose best neighborhood
- Hill-climbing search; can use sim. annealing
- How to choose neighbors?

Biological evolution (again)

- Evolution: can be viewed as highly-parallel hill-climbing search
  - "Goal": optimize for environment/produce most surviving offspring
  - Species population = state
  - Operators: mating, mutation, crossover, death
  - Pruning: death or lack of offspring
  - So parallel hill-climbing beam search
- Very successful, very flexible
- How to mimic?

Genetic algorithms

- *Genetic algorithm:*
  - Parallel hill-climbing search
  - Fixed beam-size (cf. evolution: population size)
- States: populations of individuals
- Individual: bit string (usually) – candidate solution
- *Fitness function:* applied to individual ⇒ *fitness*
- Best individuals reproduce, get rid of some ⇒ next generation

Reproduction

- Select *n* most fit individuals
- Let pairs reproduce ⇒ new children
  - No paired chromosomes, so no (re)assortment of chromosomes
  - *Mutation*
  - *Cross-over*
- Replace least fit with children

Reproduction

- Select *n* most fit individuals
- Let pairs reproduce ⇒ new children
  - No paired chromosomes, so no (re)assortment of chromosomes
  - *Mutation*

A	C	G	T	A	G	G	C	C	C	G	G	G	G	G	A

↓

A	C	G	T	A	G	G	C	T	C	G	G	G	G	G	A

  - *Cross-over*
- Replace least fit with children

Reproduction

- Select *n* most fit individuals
- Let pairs reproduce ⇒ new children
  - No paired chromosomes, so no (re)assortment of chromosomes
  - *Mutation*
  - *Cross-over*
- Replace least fit with children

Reproduction

- Select *n* most fit individuals
- Let pairs reproduce ⇒ new children
  - No paired chromosomes, so no (re)assortment of chromosomes
  - *Mutation*
  - *Cross-over*

A	C	G	T	A	G	G	C	C	C	G	G	G	G	G	A
G	C	A	C	A	C	G	C	A	C	A	C	A	C	A	C

↓

A	C	G	T	A	C	G	C	A	C	A	C	A	C	A	C
G	C	A	C	A	G	G	C	C	C	G	G	G	G	G	A

- Replace least fit with children

Parameters

- Mutation rate
- Crossover rate

Fitness function

- Can use heuristic function
- Can also simulate solution, measure fitness
- E.g.:
  - Learning a string: How many characters are correct?
  - TSP: combination of penalty for it not being a circuit, not being a Hamiltonian circuit, path cost

Genetic algorithms

- Pros:
  - Good for scheduling
  - Good for optimization problems
  - Design: antennae, solar collection mirrors, robot gait,. . .
- Cons:
  - Fitness function:
    - Applied *many* times, can be costly
    - Maybe difficult to devise
  - Not clear what termination conditions should be sometimes
  - Local optima sometimes

Case-based reasoning

Case-based reasoning

- Some problems:
  - Little domain knowledge
  - Sparse examples
- E.g., law, design
- Even medicine: Rich domain knowledge, but insufficient/too complex
- Approach for these domains:
  - Compare current problem to previous, similar ones solved (*cases*)
  - Transfer information to help solve current one

CBR examples

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Labor mediation
- Meal planning
- Law
- Dispute mediation

Copyright © 2019 UMaine School of Computing and Information Science

How does it work?

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Solve a problem $\Rightarrow$ create a case respresentation $\rightarrow$ memory
- New case: get *reminded* of prior case(s)
- Use old case:
  - Make predictions
  - Try same solution
  - Try similar reasoning

Copyright © 2019 UMaine School of Computing and Information Science

Finding a case: Dynamic memory

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning

- Could use anything
- Often used a *dynamic memory* [Schank, Kolodner]
- Store cases relative to similarities and differences
- Similar cases encountered:
  - create *memory organization packet* (MOP) from similarities
  - identify *predictive features* to discriminate between cases
  - *index* each case based on values of predictive features
- Add new case: traverse MOP structure based on case's features/values
- Model of human memory: have to *elaborate* indices, can forget
- Memory organization/reorganizaton based on use

Copyright © 2019 UMaine School of Computing and Information Science

Schema-Based Reasoning

Motivation

- Problem with CBR:
  - anectdotal reasoning
  - doesn't use induction of generalizations
- But it *does* have generalized cases: MOPs
- Why not use them and only fall back on cases?
- MOPs are *schemas*
- *Schema-based reasoning* [Turner]

Artificial Intelligence

Example: MEDIC

- Doctors *do* use cases: "I remember one time I saw. . . "
- But seems to be the exception rather than rule: "Usually when you see something like this. . . "
- MEDIC program: schema-based reasoner in small area of pulmonology
- Signs and symptoms schemas that represent generalized cases
- *Interpret* the schemas rather than transfer information

Artificial Intelligence

Kinds of schemas

- Contextual schemas (prev: dxMOPs):
  - represent problem-solving sessions or parts of sessions
  - generalized cases
- Procedural schemas:
  - generalized action portions of similar cases
  - can be plans, actions
- Strategic schemas:
  - represent problem-solving strategies
  - e.g., of a novice, expert, . . .
  - "meta-contextual schemas"
  - not currently used

Artificial Intelligence

Current work: Orca

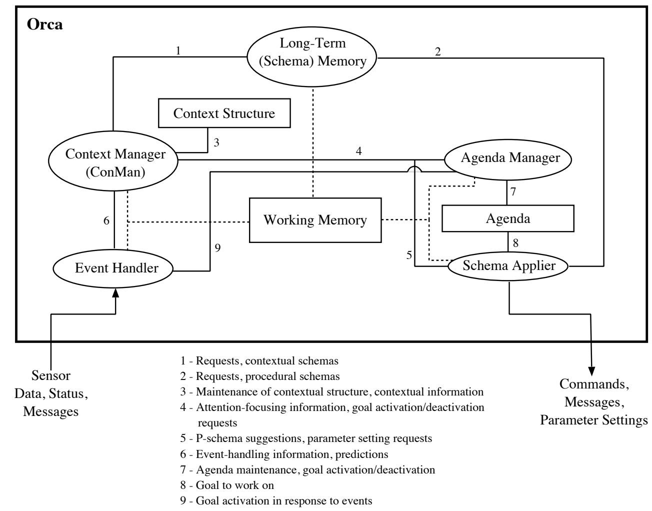
- Initial domain: AUV control
- Uses more mature form of SBR: *context-mediated behavior* (CMB)
- Identify current context
- Use c-schema(s) to:
  - Provide semantic knowledge
  - Make predictions about situation
  - Decide goal priorities
  - Diagnose, assess, and handle unanticipated events
  - Achieve goals

Artificial Intelligence

Orca

Machine Learning: Part I

Introduction
"Classical" ML
Induction
Decision tree learning
Explanation-based learning
Support vector machines
Genetic algorithms
Case-based reasoning
Schema-Based Reasoning



Copyright © 2019 UMaine School of Computing and Information Science

Artificial Intelligence