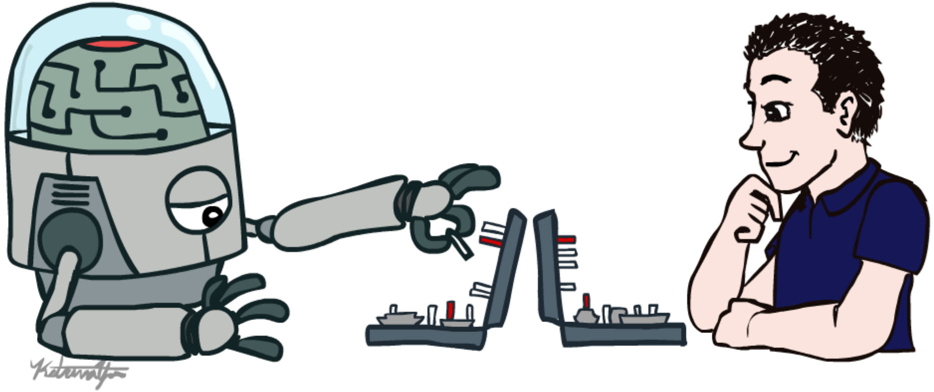
CSCI 446: Artificial Intelligence

Exam 3 Review



Instructor: Michele Van Dyne

Montana Tech

Main Topics

- Perceptrons and Logistic Regression
- Optimization and Neural Networks
- Decision Trees
- Kernels and Clustering
- Propositional Logic
- First Order (Predicate) Logic
- Philosophical Issues
- Future Directions

Perceptrons and Logistic Regression

- Error Driven Classification
  - Feature Vectors
  - Simplified Biology
- Linear Classifiers
  - Inputs
  - Weights
  - Activation
- Weight Updates
  - Adjusting weight vector (when errors)
  - Multiclass perceptrons

Perceptrons and Logistic Regression

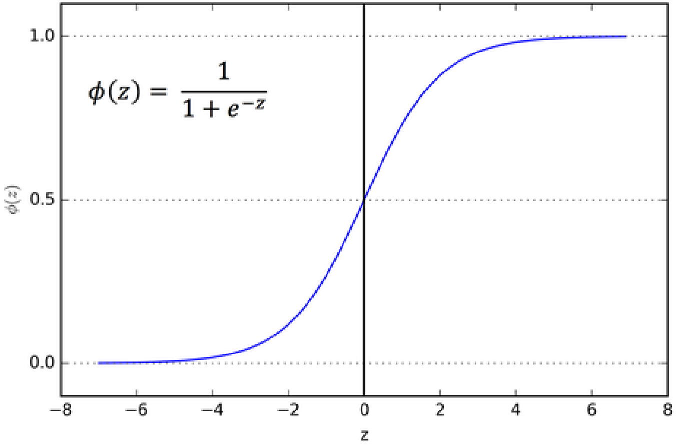
- Improving the Perceptron
  - Properties
    - Separability
    - Convergence
    - Mistake Bound
  - Problems
    - Non-linearly separable data
    - Mediocre generalization
    - Overtraining
  - Improvements
    - Probabilistic Decision – Logistic Regression
    - Multiclass Logistic Regression

How to get probabilistic decisions?

- Perceptron scoring: $z = w \cdot f(x)$
- If $z = w \cdot f(x)$ very positive → want probability going to 1
- If $z = w \cdot f(x)$ very negative → want probability going to 0

- Sigmoid function

$$\phi(z) = \frac{1}{1 + e^{-z}}$$



Best w?

- Maximum likelihood estimation:

$$\max_w \quad ll(w) = \max_w \quad \sum_i \log P(y^{(i)}|x^{(i)}; w)$$

with:

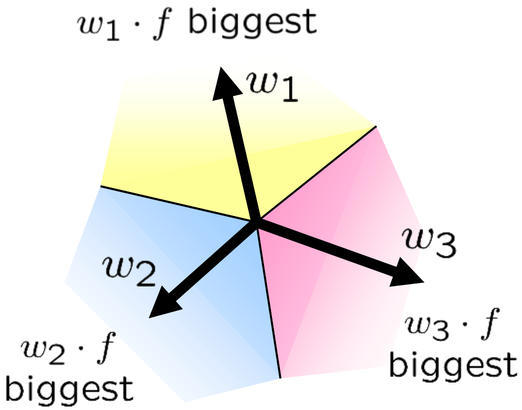
$$P(y^{(i)} = +1|x^{(i)}; w) = \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

$$P(y^{(i)} = -1|x^{(i)}; w) = 1 - \frac{1}{1 + e^{-w \cdot f(x^{(i)})}}$$

= Logistic Regression

Multiclass Logistic Regression

- Recall Perceptron:
  - A weight vector for each class: w_y
  - Score (activation) of a class y: $w_y \cdot f(x)$
  - Prediction highest score wins $y = \arg\max_y \; w_y \cdot f(x)$



- How to make the scores into probabilities?

$$\underbrace{z_1, z_2, z_3}_{\text{original activations}} \rightarrow \underbrace{\frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_2}}{e^{z_1}+e^{z_2}+e^{z_3}}, \frac{e^{z_3}}{e^{z_1}+e^{z_2}+e^{z_3}}}_{\text{softmax activations}}$$

Best w?

- Maximum likelihood estimation:

$$\max_{w} \quad ll(w) = \max_{w} \quad \sum_{i} \log P(y^{(i)} | x^{(i)}; w)$$

with:

$$P(y^{(i)} | x^{(i)}; w) = \frac{e^{w_{y^{(i)}} \cdot f(x^{(i)})}}{\sum_{y} e^{w_y \cdot f(x^{(i)})}}$$

= Multi-Class Logistic Regression

Optimization and Neural Networks

- Optimization
  - Hill Climbing / Gradient Ascent
- Neural Networks
  - Deep Neural Networks
    - Learn Features, not just Weights
  - Activation Functions
  - Properties
    - Universal Function Approximation
  - Computing all those Derivatives
  - How Well do they Work?

Decision Trees

- Formalizing Learning
  - Inductive Learning
  - Consistency / Bias
    - Algorithm Preference
  - Simplicity / Variance
    - Reduce hypothesis space
    - Regularization
- Decision Trees
  - Expressiveness
  - Information Gain
    - Entropy and Information
    - Recursive tree building process
  - Overfitting
    - Pruning

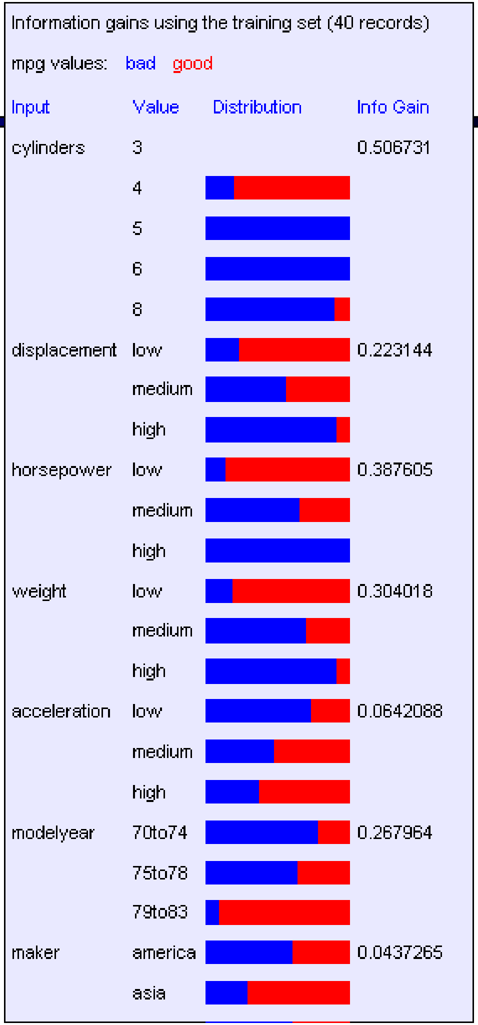
Example: Miles Per Gallon

40 Examples

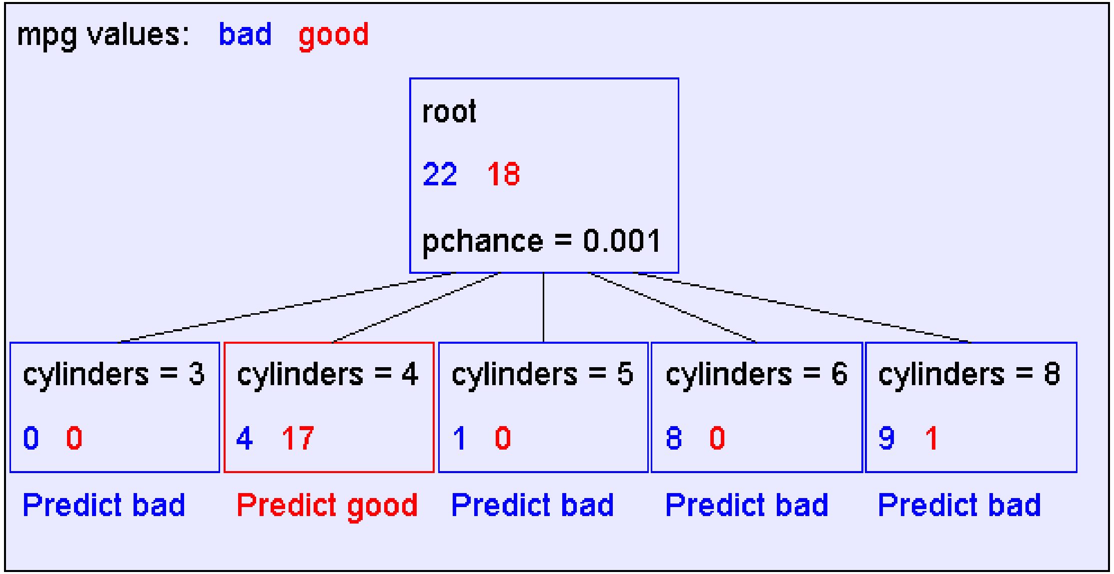
mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low	medium	low	medium	70to74	asia
bad	4	low	medium	low	low	70to74	asia
bad	8	high	high	high	low	75to78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
bad	8	high	high	high	low	70to74	america
good	8	high	medium	high	high	79to83	america
bad	8	high	high	high	low	75to78	america
good	4	low	low	low	low	79to83	america
bad	6	medium	medium	medium	high	75to78	america
good	4	medium	low	low	low	79to83	america
good	4	low	low	medium	high	79to83	america
bad	8	high	high	high	low	70to74	america
good	4	low	medium	low	medium	75to78	europe
bad	5	medium	medium	medium	medium	75to78	europe

Find the First Split

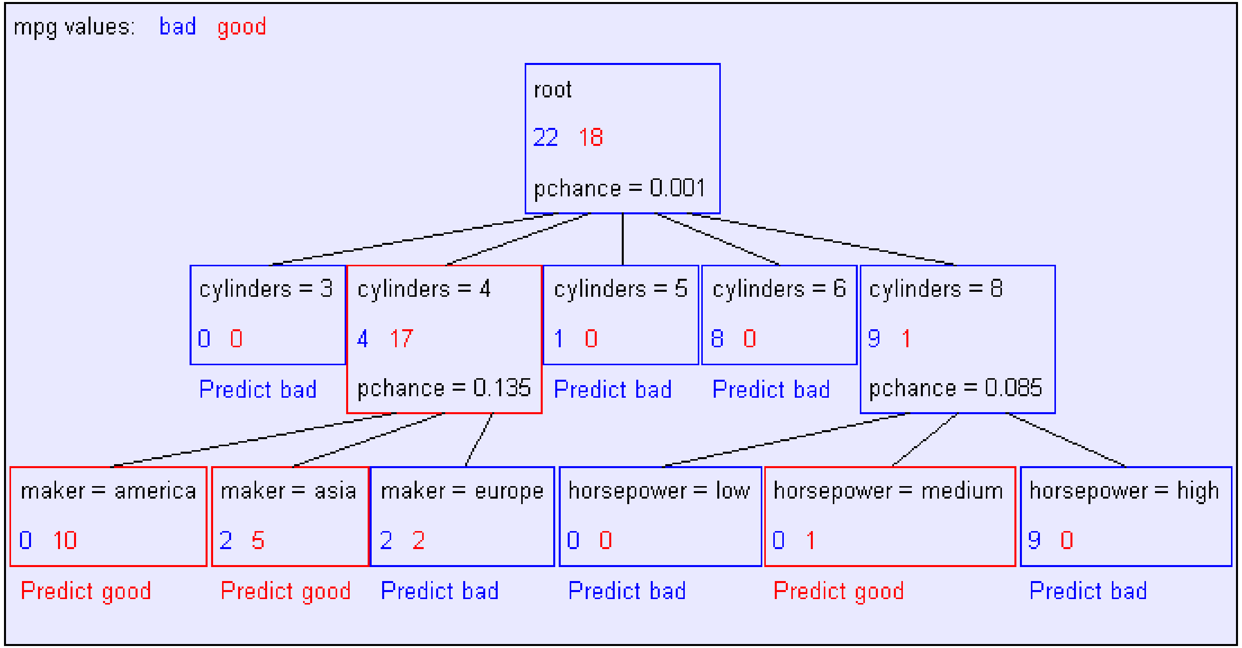
- Look at information gain for each attribute
- Note that each attribute is correlated with the target!
- What do we split on?



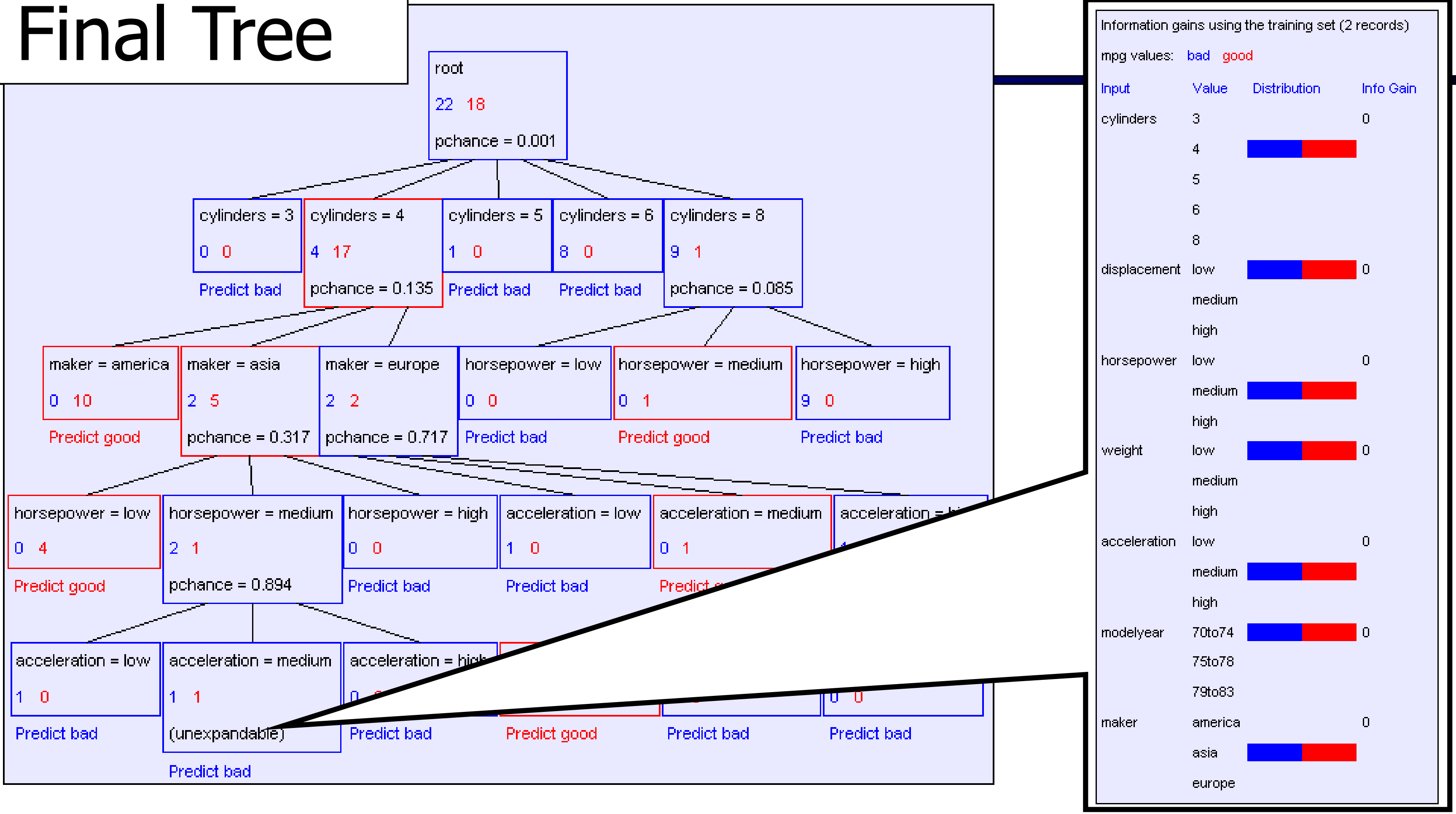
Result: Decision Stump



Second Level

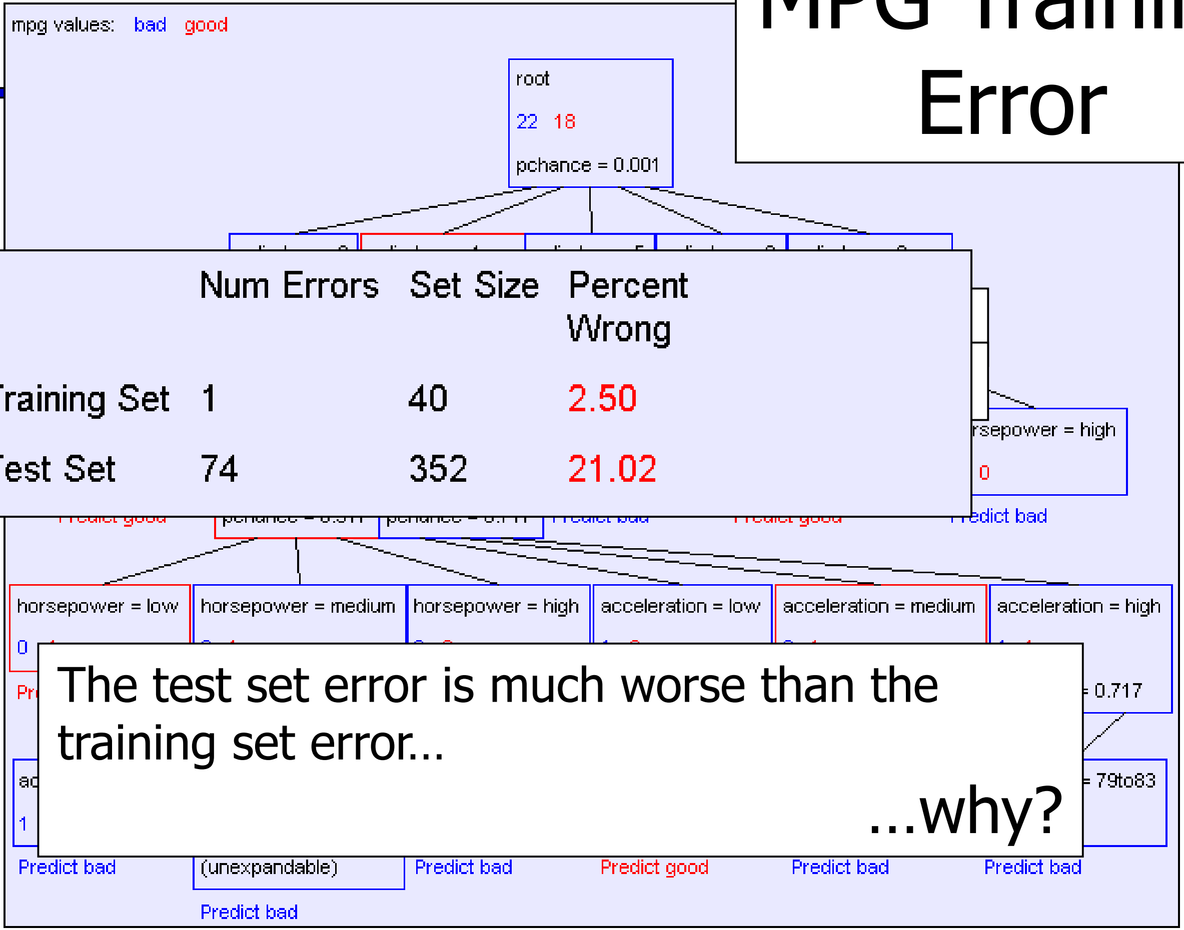


Final Tree



root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
horsepower = low
0 0
Predict bad
horsepower = medium
0 1
Predict good
horsepower = high
9 0
Predict bad
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
horsepower = high
0 0
Predict bad
acceleration = low
1 0
Predict bad
acceleration = medium
0 1
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
acceleration = high
Predict bad
Predict good
Predict bad
0 0
Predict bad
Information gains using the training set (2 records)
mpg values: bad good
Input
Value
Distribution
Info Gain
cylinders
3
0
4
5
6
8
displacement
low
0
medium
high
horsepower
low
0
medium
high
weight
low
0
medium
high
acceleration
low
0
medium
high
modelyear
70to74
0
75to78
79to83
maker
america
0
asia
europe

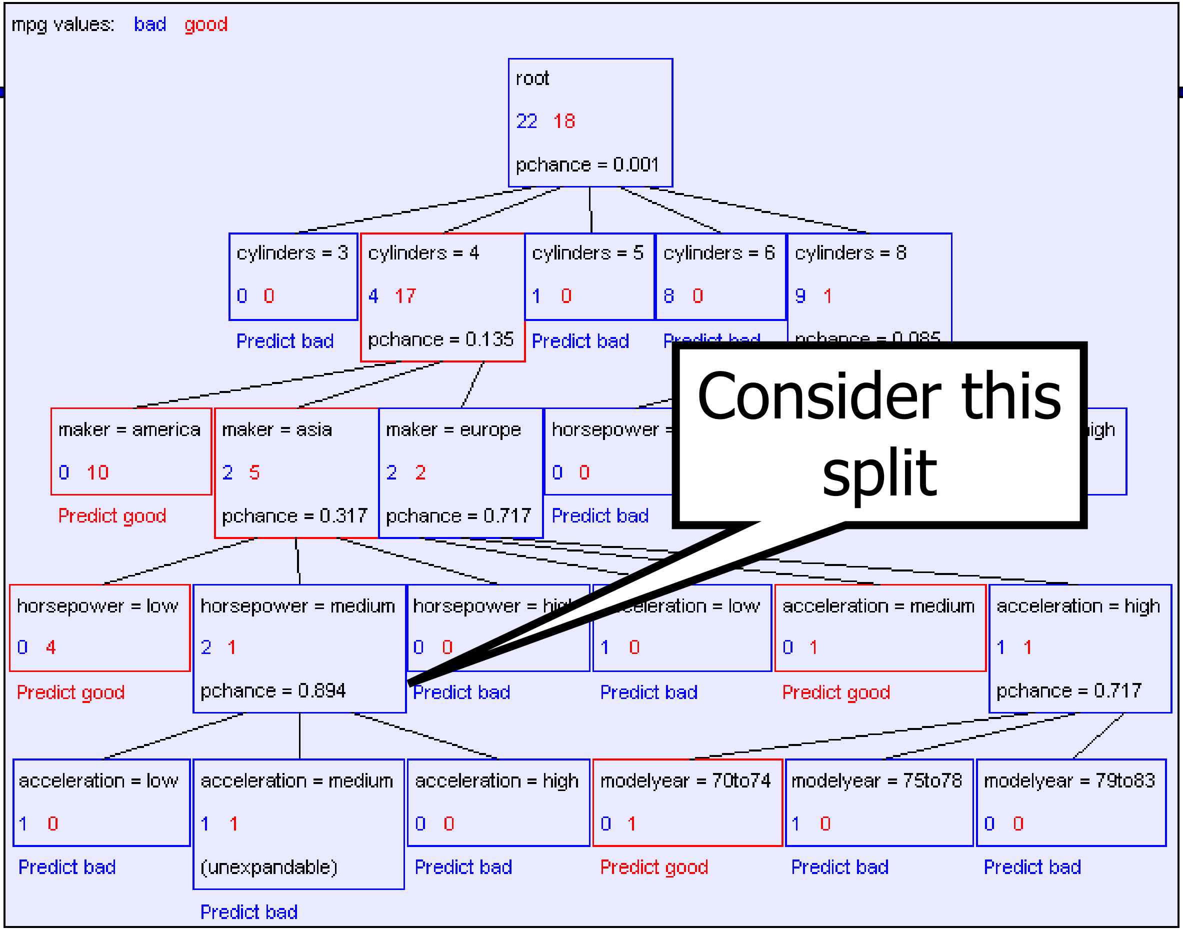
MPG Training Error



	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	74	352	21.02

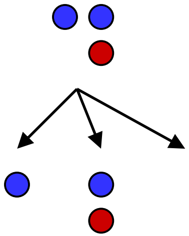
The test set error is much worse than the training set error…

…why?


mpg values: bad good
root
22 18
pchance = 0.001
cylinders = 3
0 0
Predict bad
cylinders = 4
4 17
pchance = 0.135
cylinders = 5
1 0
Predict bad
cylinders = 6
8 0
Predict bad
cylinders = 8
9 1
pchance = 0.085
maker = america
0 10
Predict good
maker = asia
2 5
pchance = 0.317
maker = europe
2 2
pchance = 0.717
0 0
Predict bad
horsepower = low
0 4
Predict good
horsepower = medium
2 1
pchance = 0.894
0 0
Predict bad
1 0
Predict bad
acceleration = medium
0 1
Predict good
acceleration = high
1 1
pchance = 0.717
acceleration = low
1 0
Predict bad
acceleration = medium
1 1
(unexpandable)
Predict bad
acceleration = high
0 0
Predict bad
modelyear = 70to74
0 1
Predict good
modelyear = 75to78
1 0
Predict bad
modelyear = 79to83
0 0
Predict bad
Consider this split

Significance of a Split

- Starting with:
  - Three cars with 4 cylinders, from Asia, with medium HP
  - 2 bad MPG
  - 1 good MPG



- What do we expect from a three-way split?
  - Maybe each example in its own subset?
  - Maybe just what we saw in the last slide?

- Probably shouldn't split if the counts are so small they could be due to chance

- A chi-squared test can tell us how likely it is that deviations from a perfect split are due to chance*

- Each split will have a significance value, p_{CHANCE}

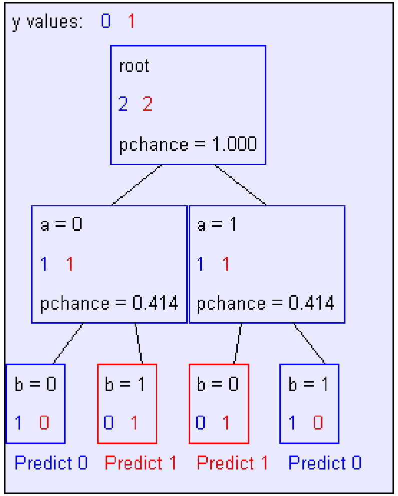
Keeping it General

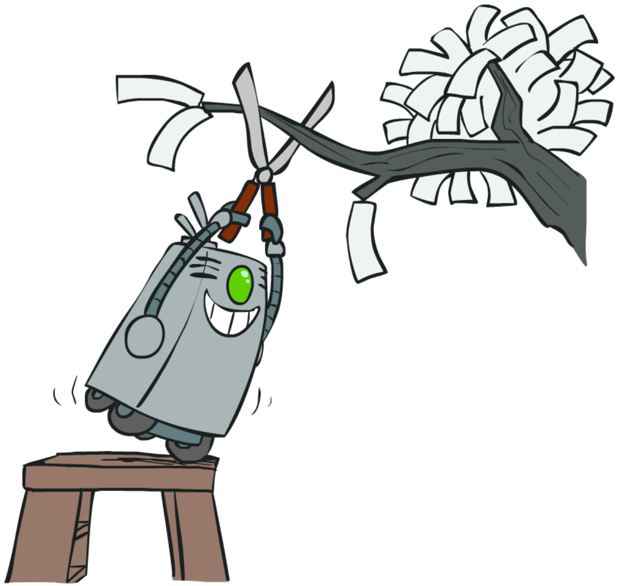
- Pruning:
  - Build the full decision tree
  - Begin at the bottom of the tree
  - Delete splits in which

    $p_{CHANCE} > MaxP_{CHANCE}$
  - Continue working upward until there are no more prunable nodes
  - Note: some chance nodes may not get pruned because they were "redeemed" later

y = a XOR b

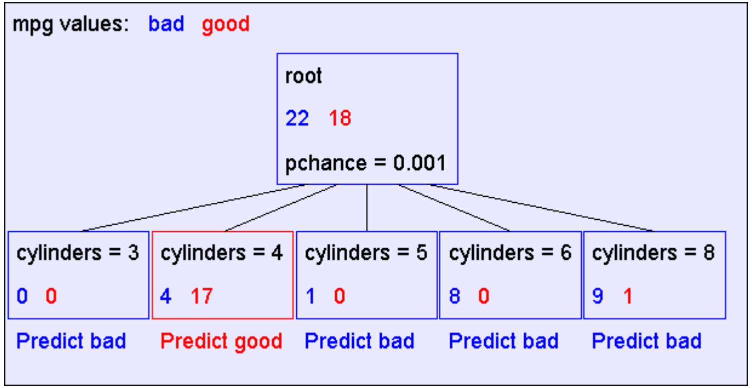
a	b	y
0	0	0
0	1	1
1	0	1
1	1	0





Pruning example

- With $\text{MaxP}_{\text{CHANCE}} = 0.1$:



Note the improved test set accuracy compared with the unpruned tree

	Num Errors	Set Size	Percent Wrong
Training Set	5	40	12.50
Test Set	56	352	15.91

Kernels and Clustering

- Case-Based Learning
  - Similarity Functions
  - k-Nearest Neighbors
- Kernelization
  - Perceptron Dual View
- Non-Linearity
  - Perceptron Kernel Functions

Perceptron Weights

- What is the final value of a weight w_y of a perceptron?
  - Can it be any real vector?
  - No! It's built by adding up inputs.

$$w_y = 0 + f(x_1) - f(x_5) + \dots$$

$$w_y = \sum_i \alpha_{i,y} f(x_i)$$

- Can reconstruct weight vectors (the primal representation) from update counts (the dual representation)

$$\alpha_y = \langle \alpha_{1,y} \;\; \alpha_{2,y} \;\; \dots \;\; \alpha_{n,y} \rangle$$

Dual Perceptron

- How to classify a new example x?

$$
\begin{aligned}
\text{score}(y, x) &= w_y \cdot f(x) \\
&= \left(\sum_i \alpha_{i,y} f(x_i) \right) \cdot f(x) \\
&= \sum_i \alpha_{i,y} \left(f(x_i) \cdot f(x) \right) \\
&= \sum_i \alpha_{i,y} \, K(x_i, x)
\end{aligned}
$$

- If someone tells us the value of K for each pair of examples, never need to build the weight vectors (or the feature vectors)!

Dual Perceptron

- Start with zero counts (alpha)
- Pick up training instances one by one
- Try to classify x_n,

$$y = \arg\max_y \sum_i \alpha_{i,y} K(x_i, x_n)$$

- If correct, no change!
- If wrong: lower count of wrong class (for this instance), raise count of right class (for this instance)

$$\alpha_{y,n} = \alpha_{y,n} - 1 \qquad w_y = w_y - f(x_n)$$

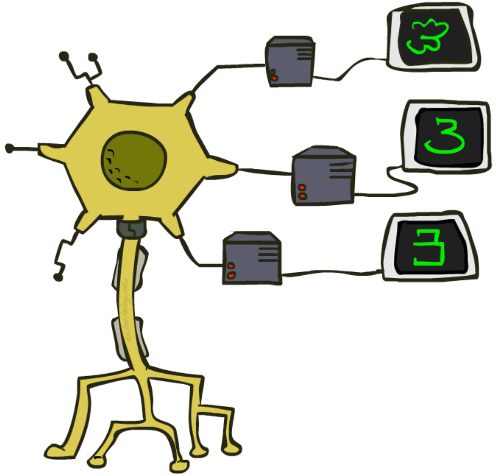
$$\alpha_{y^*,n} = \alpha_{y^*,n} + 1 \qquad w_{y^*} = w_{y^*} + f(x_n)$$

Kernelized Perceptron

- If we had a black box (kernel) K that told us the dot product of two examples x and x':
  - Could work entirely with the dual representation
  - No need to ever take dot products ("kernel trick")

$$\text{score}(y, x) = w_y \cdot f(x)$$

$$= \sum_i \alpha_{i,y} K(x_i, x)$$



- Like nearest neighbor – work with black-box similarities
- Downside: slow if many examples get nonzero alpha

Kernels: Who Cares?

- So far: a very strange way of doing a very simple calculation
- "Kernel trick": we can substitute any* similarity function in place of the dot product
- Lets us learn new kinds of hypotheses

* Fine print: if your kernel doesn't satisfy certain technical requirements, lots of proofs break. E.g. convergence,

Some Kernels

- Kernels implicitly map original vectors to higher dimensional spaces, take the dot product there, and hand the result back

- Linear kernel: $$K(x, x') = x' \cdot x' = \sum_i x_i x'_i$$

- Quadratic kernel: $$K(x, x') = (x \cdot x' + 1)^2$$
$$= \sum_{i,j} x_i x_j \, x'_i x'_j + 2 \sum_i x_i \, x'_i + 1$$

- RBF: infinite dimensional representation
$$K(x, x') = \exp(-||x - x'||^2)$$

- Discrete kernels: e.g. string kernels

Why Kernels?

- Can't you just add these features on your own (e.g. add all pairs of features instead of using the quadratic kernel)?
  - Yes, in principle, just compute them
  - No need to modify any algorithms
  - But, number of features can get large (or infinite)
  - Some kernels not as usefully thought of in their expanded representation, e.g. RBF kernels

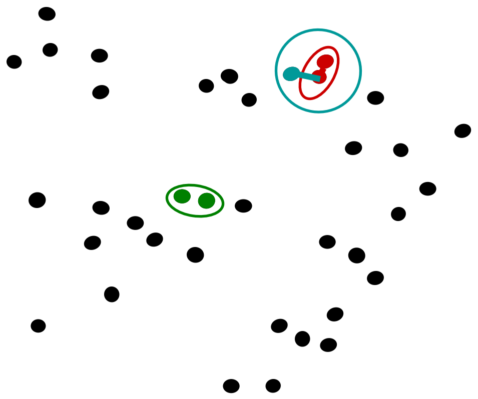
- Kernels let us compute with these features implicitly
  - Example: implicit dot product in quadratic kernel takes much less space and time per dot product
  - Of course, there's the cost for using the pure dual algorithms: you need to compute the similarity to every training datum

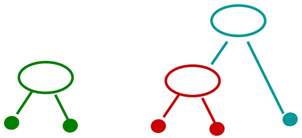
Kernels and Clustering

- Clustering
  - Types of learning
    - Supervised
    - Unsupervised
  - K-Means
    - K-Means Process
    - Issues
  - Agglomerative
    - Agglomerative Process
    - Issues

Agglomerative Clustering

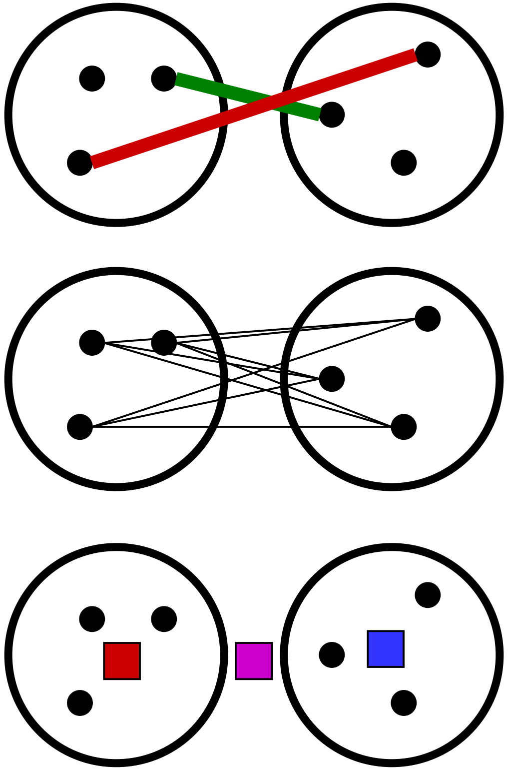
- Agglomerative clustering:
  - First merge very similar instances
  - Incrementally build larger clusters out of smaller clusters
- Algorithm:
  - Maintain a set of clusters
  - Initially, each instance in its own cluster
  - Repeat:
    - Pick the two closest clusters
    - Merge them into a new cluster
    - Stop when there's only one cluster left
- Produces not one clustering, but a family of clusterings represented by a dendrogram





Agglomerative Clustering

- How should we define “closest” for clusters with multiple elements?
- Many options
  - Closest pair (single-link clustering)
  - Farthest pair (complete-link clustering)
  - Average of all pairs
  - Ward’s method (min variance, like k-means)
- Different choices create different clustering behaviors



Propositional Logic

- Knowledge Based Agents
  - Knowledge Base
  - Inference Engine
  - Separation of Knowledge and Process
- An Example
  - Wumpus World
- General Logic
  - Entailment
  - Models
  - Inference

Propositional Logic

- Propositional Logic
  - Syntax
  - Truth Tables
- Equivalence, Validity, Satisfiability
- Inference Rules / Theorem Proving
  - Forward and Backward Chaining
    - Horn Form
    - Modus Ponens
  - Resolution
    - Conjunctive Normal Form (CNF)
    - Conversion to CNF
    - Resolution

Forward and Backward Chaining

$P \Rightarrow Q$

$L \land M \Rightarrow P$

$B \land L \Rightarrow M$

$A \land P \Rightarrow L$

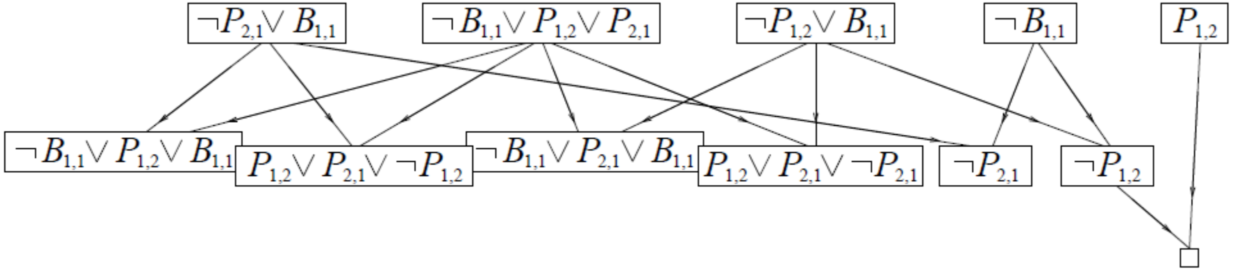
$A \land B \Rightarrow L$

A

B

Resolution Example

$$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1} \quad \alpha = \neg P_{1,2}$$



First Order (Predicate) Logic

- Overview
- Syntax and Semantics
  - Basic Elements
  - Atomic Sentences
  - Complex Sentences
  - Models
  - Universal Quantification
  - Existential Quantification
- Fun with Sentences
  - Equality

Universal Quantification

- ∀ <variables> <sentence>

- Everyone at MontanaTech is smart:

$\forall x\ At(x, MontanaTech)\ ) \Rightarrow Smart(x)$

- ∀ x P is true in a model *m* iff P is true with x being each possible object in the model
- Roughly speaking, equivalent to the conjunction of instantiations of P

(At(KingJohn, MontanaTech)) ⇒ Smart(KingJohn))

∧ (At(Richard, MontanaTech)) ⇒ Smart(Richard))

∧ (At(MontanaTech, MontanaTech)) ⇒ Smart(MontanaTech))

∧ …

A common Mistake to Avoid

- Typically, $\Rightarrow$ is the main connective with $\forall$
- Common mistake: using $\wedge$ as the main connective with $\forall$:

$\forall x\ At(x, MontanaTech) \wedge Smart(x)$

- Means "Everyone is at MontanaTech and everyone is smart"

Existential Quantification

- ∃ <variables> <sentence>

- Someone at MSU is smart:

∃x At(x, MSU) ∧ Smart(x)

- ∃x P is true in a model *m* iff P is true with x being some possible object in the model
- Roughly speaking, equivalent to the disjunction of instantiations of P

(At(KingJohn, MSU) ∧ Smart(KingJohn))
∨ (At(Richard, MSU) ∧ Smart(Richard))
∨ (MSU, MSU) ∧ MSU))
∨ …

Another Common Mistake to Avoid

- Typically, $\wedge$ is the main connective with $\exists$
- Common mistake: using $\Rightarrow$ as main connective with $\exists$:

$\exists$ x At(x, MSU) $\Rightarrow$ Smart(x)

- True if there is anyone who is not at MSU!

Properties of Quantifiers

- $\forall x\ \forall y$ is the same as $\forall y\ \forall x$ (why??)
- $\exists x\ \exists y$ is the same as $\exists y\ \exists x$ (why??)
- $\exists x\ \forall y$ is not the same as $\forall y\ \exists x$

- An Example:
  - $\exists x\ \forall y$ Loves(x, y)
  - There exists a person who loves all people.

  - $\forall y\ \exists x$ Loves(x, y)
  - All people are loved by at least someone.

- Another Example:
  - $\forall n\ \exists s\ n*n = s$
  - For every natural number n, there exists a natural number s such that $n^2 = s$.

  - $\exists s\ \forall n\ n*n = s$
  - There exists a natural number s such that for all natural numbers n, $n^2 = s$.

- Quantifier duality: each can be expressed using the other

$\forall x$ Likes(x, IceCream)
$\neg\exists x\ \neg$Likes(x, IceCream)

$\exists x$ Likes(x,Broccoli)
$\neg\forall x\ \neg$Likes(x,Broccoli)

First Order (Predicate) Logic

- Unification
  - Universal Instantiation
  - Existential Instantiation
  - Reduction to Propositional Inference
  - Unification
- Generalized Modus Ponens
- Forward and Backward Chaining
- Resolution

Unification

- We can get the inference immediately if we can find a substitution Θ such that King(x) and Greedy(x) match King(John) and Greedy(y)
- Θ = {x/John, y/John} works
- Unify(α, β) = Θ, if αΘ = βΘ

Unification

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	
$Knows(John, x)$	$Knows(y, OJ)$	
$Knows(John, x)$	$Knows(y, Mother(y))$	
$Knows(John, x)$	$Knows(x, OJ)$	

Generalized Modus Ponens (GMP)

$$\frac{p_1{}',\ \ p_2{}',\ \ldots,\ p_n{}',\ \ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \qquad \text{where } p_i{}'\theta = p_i\theta \text{ for all } i$$

$p_1{}'$ is $King(John)$ $\quad$ p_1 is $King(x)$

$p_2{}'$ is $Greedy(y)$ $\quad$ p_2 is $Greedy(x)$

θ is $\{x/John, y/John\}$ $\quad$ q is $Evil(x)$

$q\theta$ is $Evil(John)$

- GMP used with KB of definite clauses (exactly one positive literal)
- All variables assumed universally quantified
- GMP is sound

Forward Chaining Proof

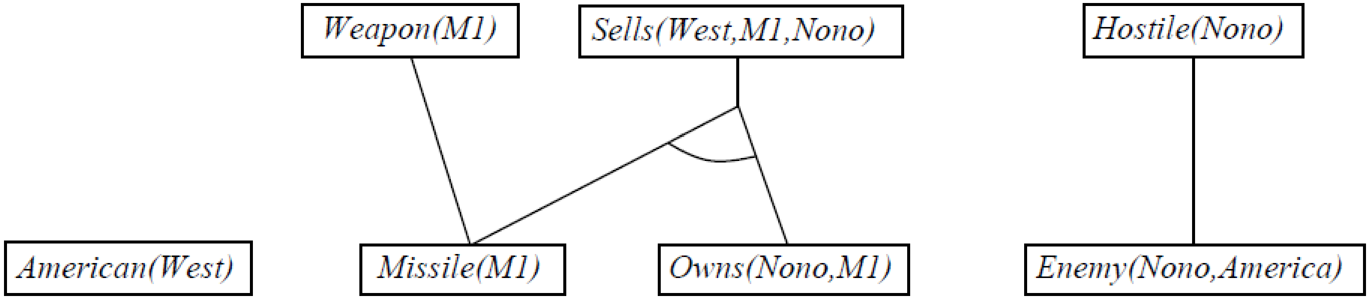
American(West)

Missile(M1)

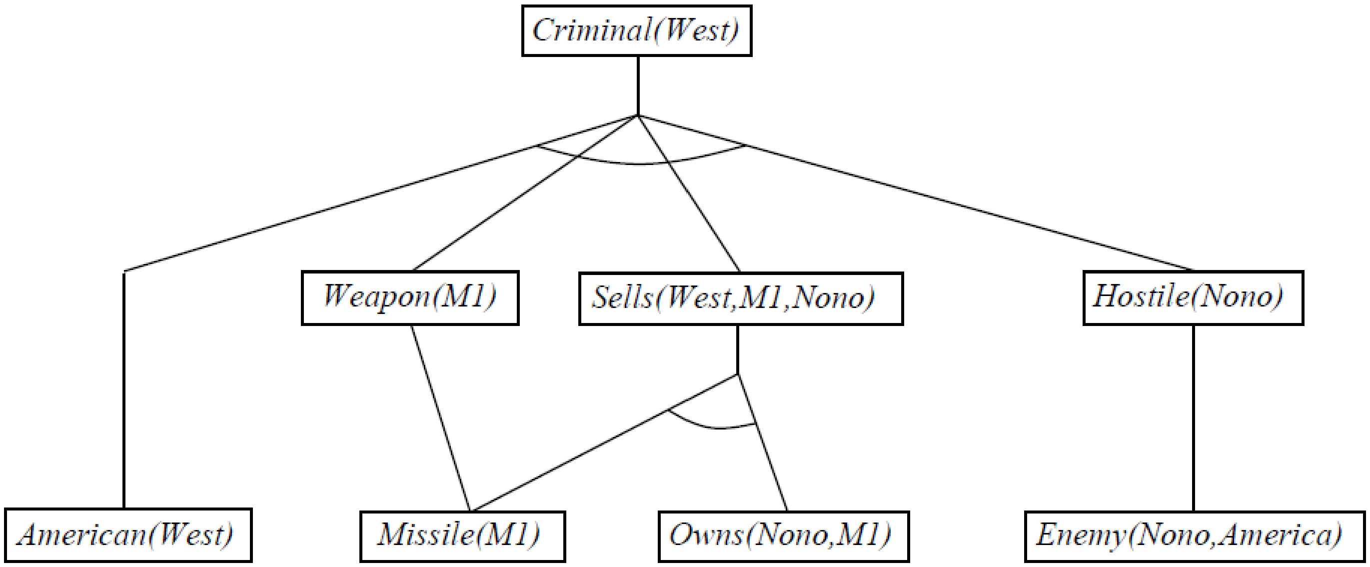
Owns(Nono,M1)

Enemy(Nono,America)

Forward Chaining Proof



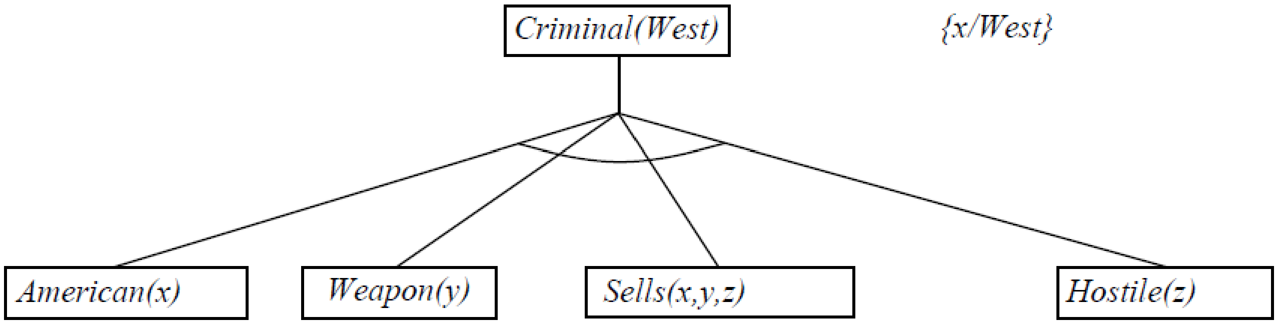
Forward Chaining Proof



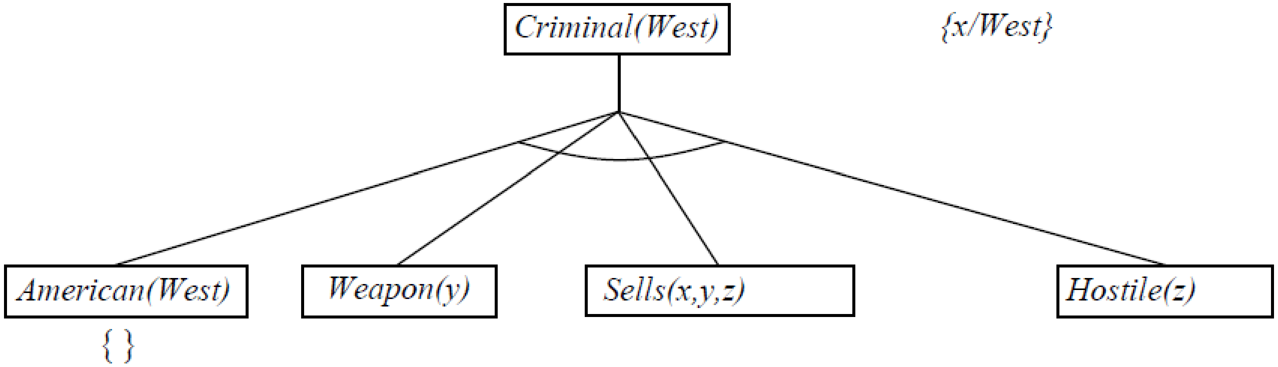
Backward Chaining Example

$\boxed{Criminal(West)}$

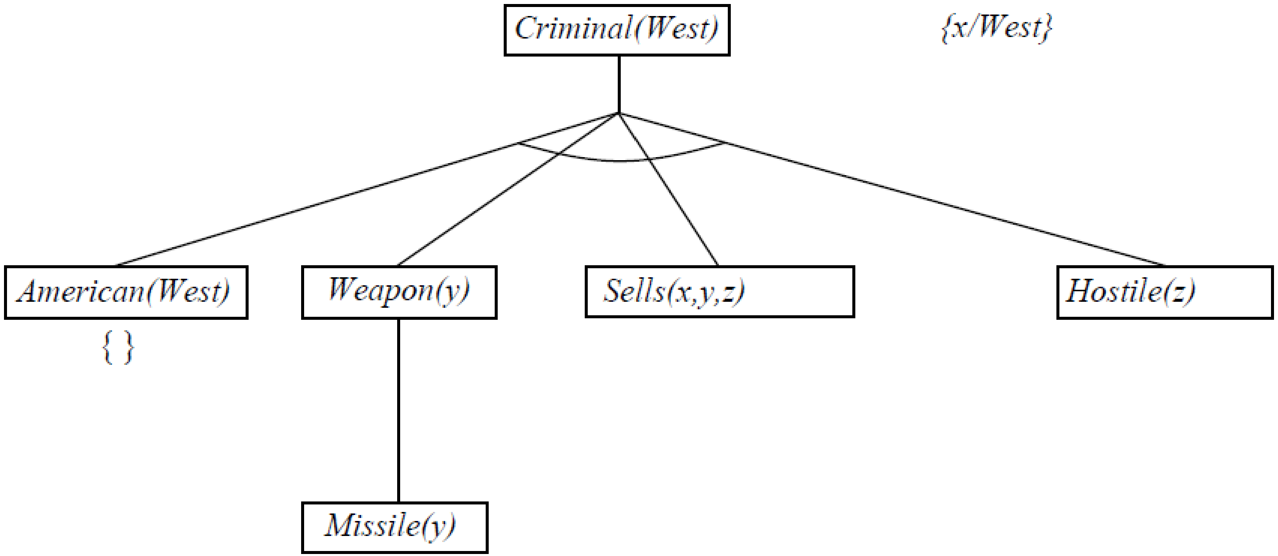
Backward Chaining Example



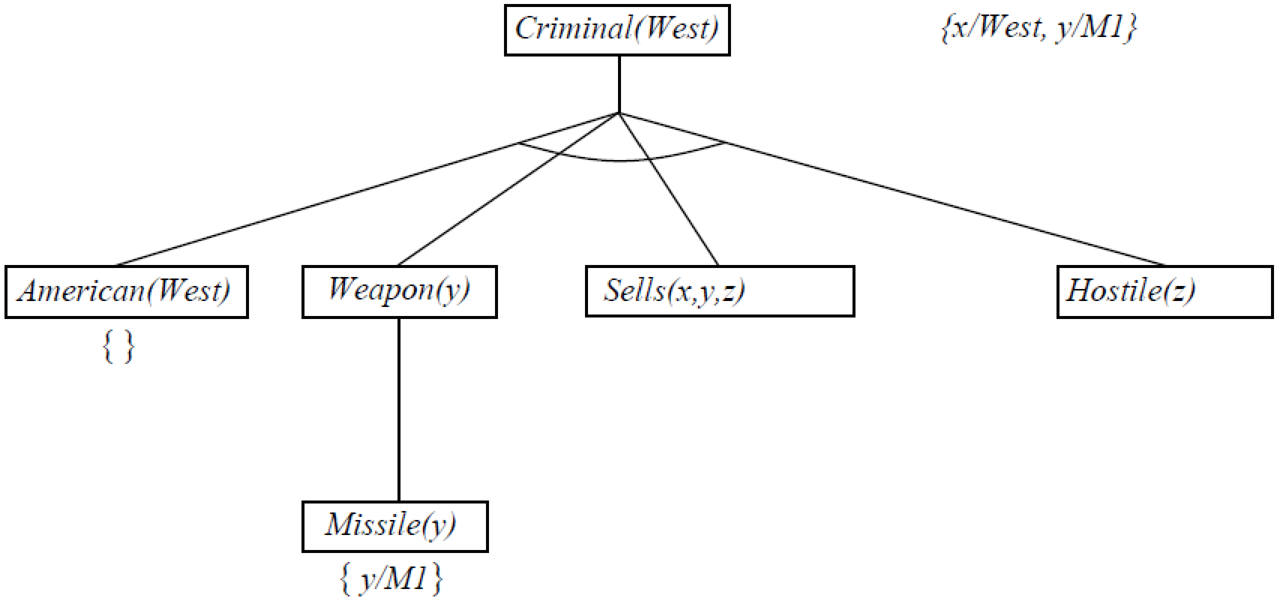
Backward Chaining Example



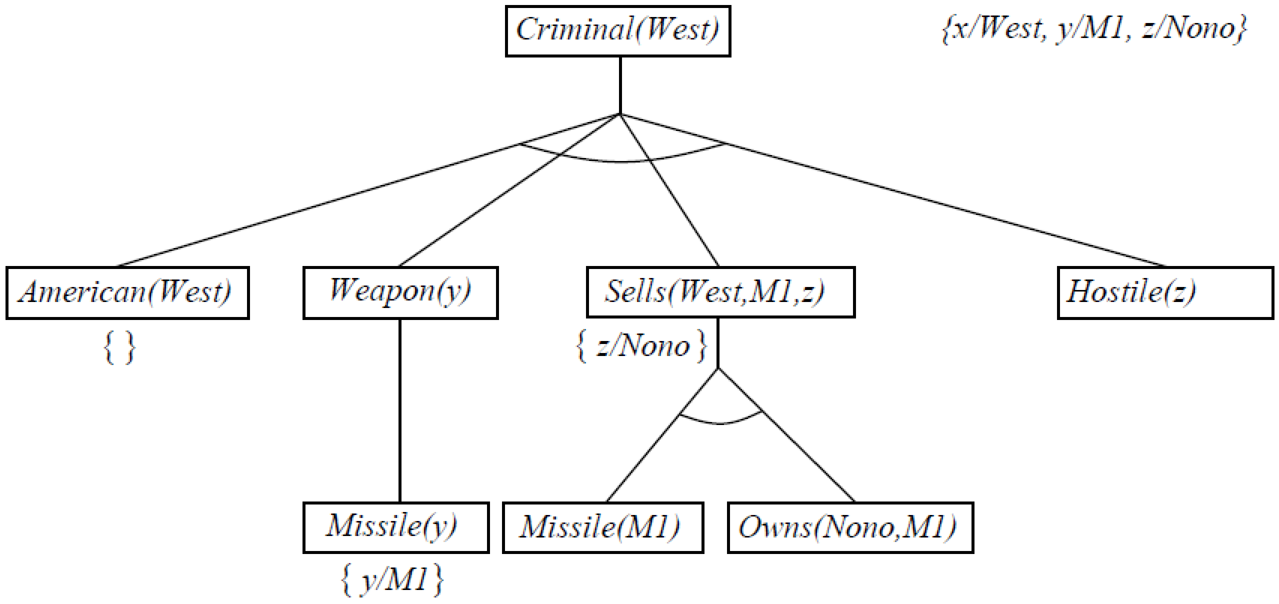
Backward Chaining Example



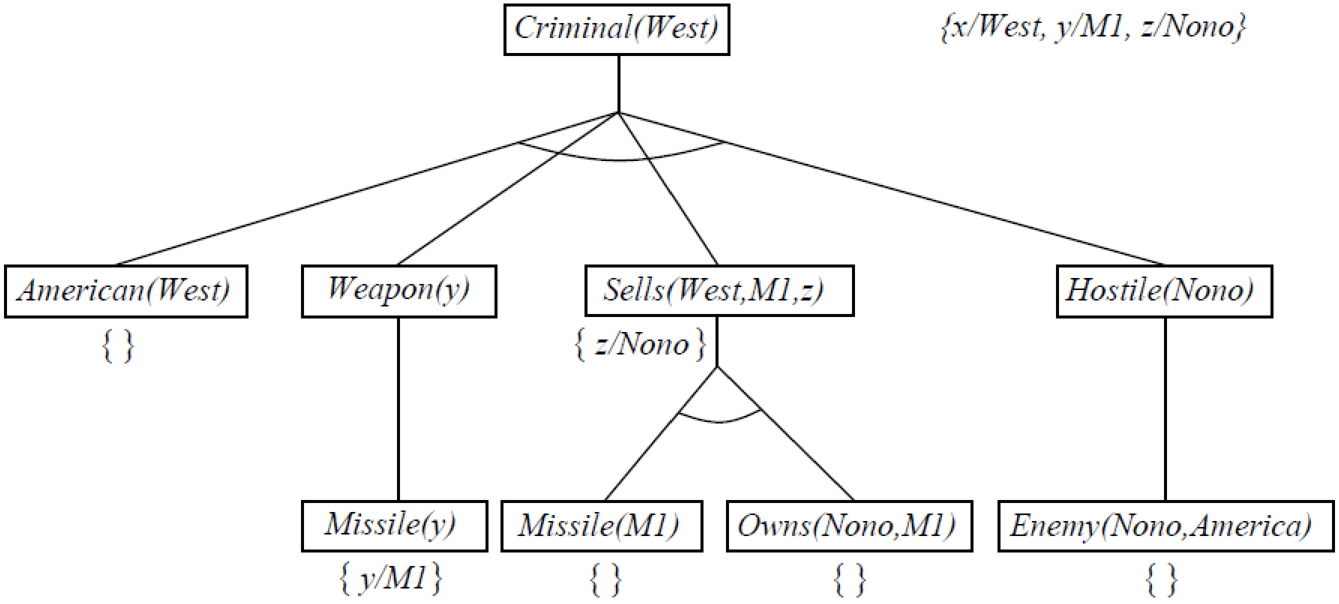
Backward Chaining Example



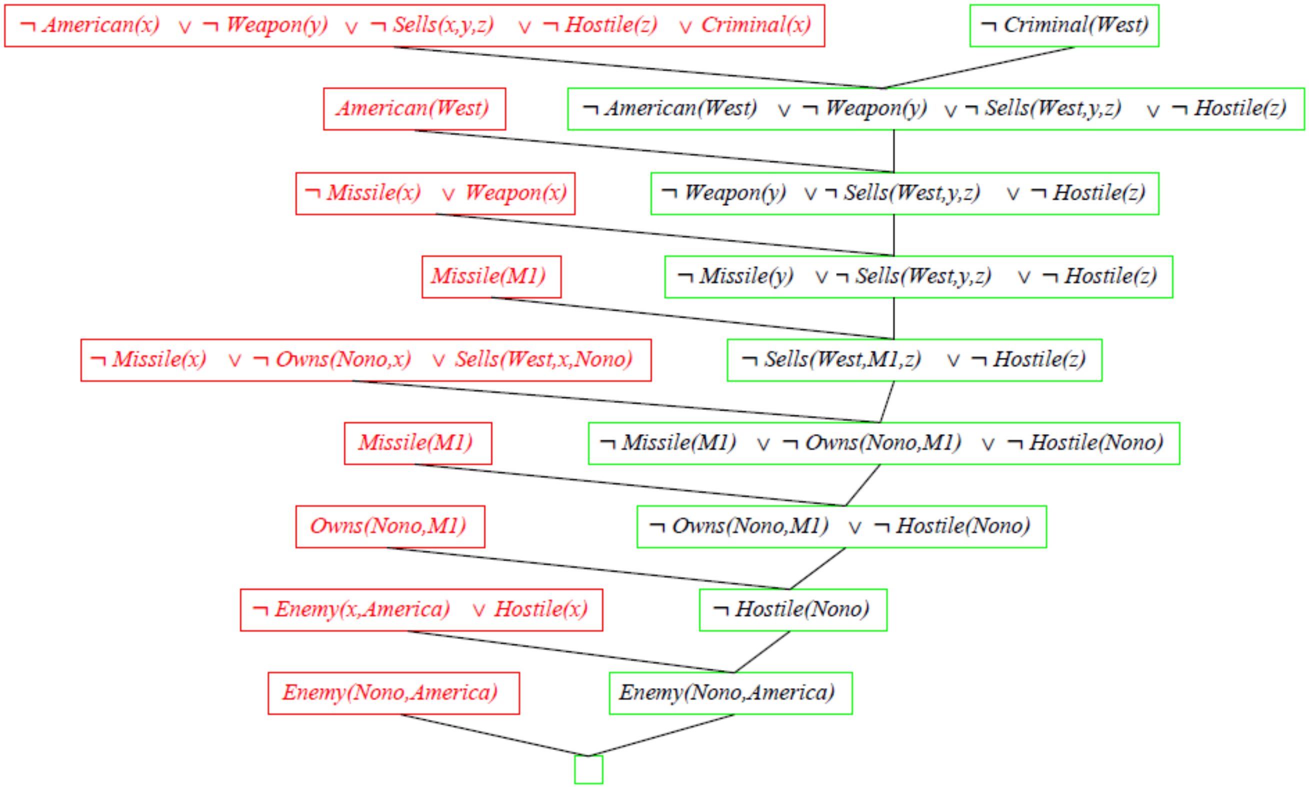
Backward Chaining Example



Backward Chaining Example



Resolution Proof: Definite Clauses



Philosophical Issues

- Weak AI
- Strong AI
- Ethics and Risks

Future Directions

- Agent Components
- Agent Architectures
- Are We Going in the Right Direction?
- What if AI Does Succeed?

Questions