Artificial Intelligence

CPSC 481

Machine Learning

Part A

Overview

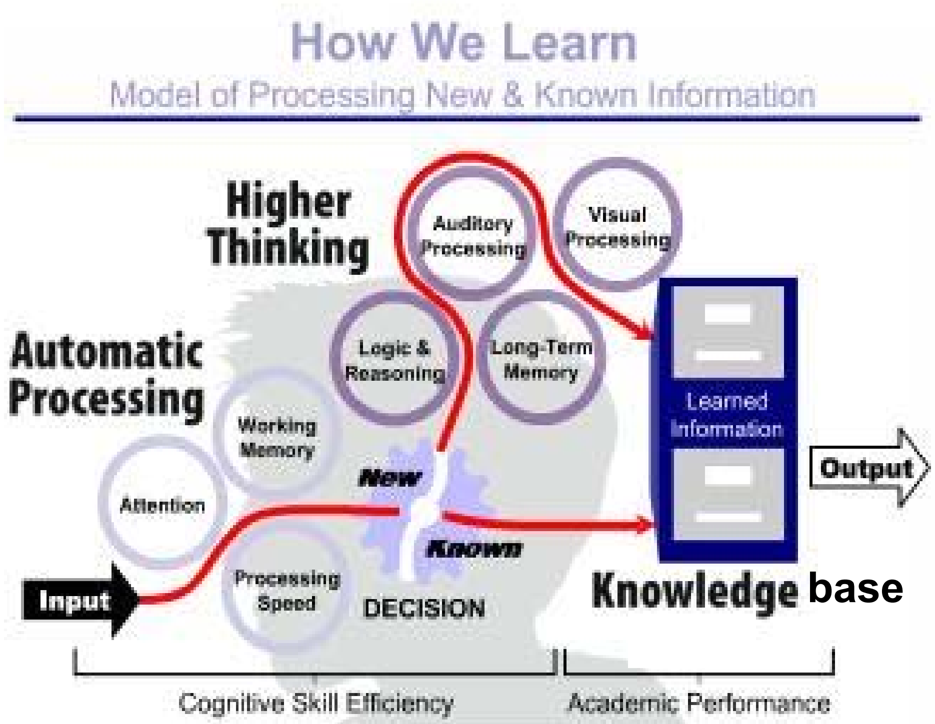
- Cognitive process of human learning
- Concepts and concept space
- What is machine learning?
- Generalization and specialization in learning
- Supervised learning
  - Decision tree induction
- Inductive bias

What is Learning?

- Learning is acquiring new knowledge, skills, values, or understanding.
  - Learning occurs as part of training or education, personal development, or experience

- How do we learn?

Cognitive Process of Human Learning



- Knowledge **acquired** through learning actions such as
  - Reading, observing, touching, doing, thinking, etc.
  - Physical actions can strengthen the knowledge learned.

Concept and Knowledge

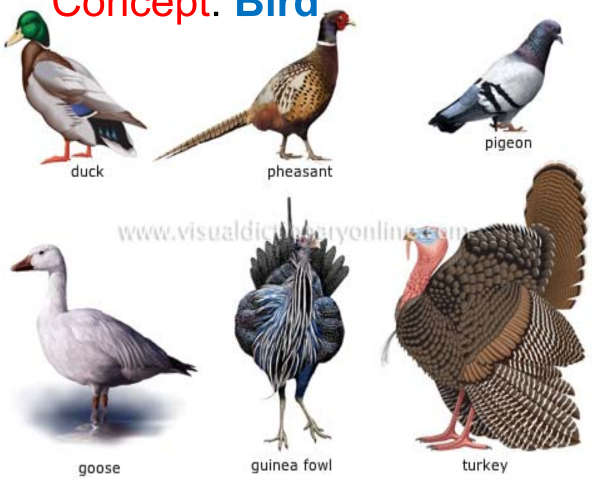
- A **<u>concept</u>** is a cognitive unit of **<u>meaning</u>** (an abstract idea or a mental symbol defined as a **unit of knowledge**).
- What is "meaning"?
- What are some examples of concepts?
- How to represent concepts?
  - Concept/knowledge representation
- What to do with the learned concepts?
  - Application or use of concepts (knowledge)

A Baby's Animal Concept Learning

Concept: **Dog**

Concept: **Cat**

Concept: **Bird**



Some questions to think about

- How does a baby formulate **concepts** of dogs, cats, birds, and animals?
- How can a baby learn the "**names**" of concepts such as dog, cat, bird?
  - Note: A name of **object class** represents a **concept**.
- What if no one tells the baby about the concept name?
- What does each **animal class** mean to a baby?
  - What is the advantage of knowing the name of a concept?

What are Possible Learning Methods?

- Think about how you learn when you try to learn something.
- Various *learning methods* to transform external data to internal knowledge (making data meaningful)
  - **Memorization or rote learning**
    - very weak and low level learning so we don't even consider this as a learning method
  - Synthesizing different types of information
  - Analogy
  - Pattern recognition
  - Generalization and specialization
  - Classification and Categorization
  - Induction
  - Deduction

What is Machine Learning?

- **Machine learning is**:
  - "Any change in a system that allows it to perform better the second time on **repetition** of the same task or on another task drawn from the same population" (Simon, 1983)
    - Which requires pattern recognition or knowledge acquisition

- **Why machine learning?**
  - One of the important AI problems
  - Knowledge engineering bottleneck. Costly to build expert systems with all the necessary knowledge.

  - **Nowadays** mainly for analysis, decision making, forecasting
    - Many real world applications such as pattern recognition, speech recognition, unmanned vehicles, Alpha go, data mining, big data analytics, etc.

Generalization as a Learning Method

- **Generalization**
  - Volleyball, baseball, football generalize to ball games or sports
- **Examples using logical expressions:**
  - Dropping conditions from a *conjunctive* expression:

    shape size color

    - Ball(round) ^ Ball(small) ^ Ball(red) generalizes to Ball(round) ^ Ball(small)
  - Adding a *disjunction* to an expression:
    - Ball(round) ^ Ball(small) ^ Ball(red) generalizes to

      Ball(round) ^ Ball(small) ^ ((Ball(red) v Ball(blue))
  - Replacing constants with variables:
    - Ball(round, red) generalizes to Ball(round, **X**)
  - Replacing a property with its parent class in a hierarchy
    - Ball(red) generalizes to Ball(*primary_color*) if *primary_color* is a superclass of red (in a color class hierarchy)
    - Use of ontology

A Concept Learning through Generalization

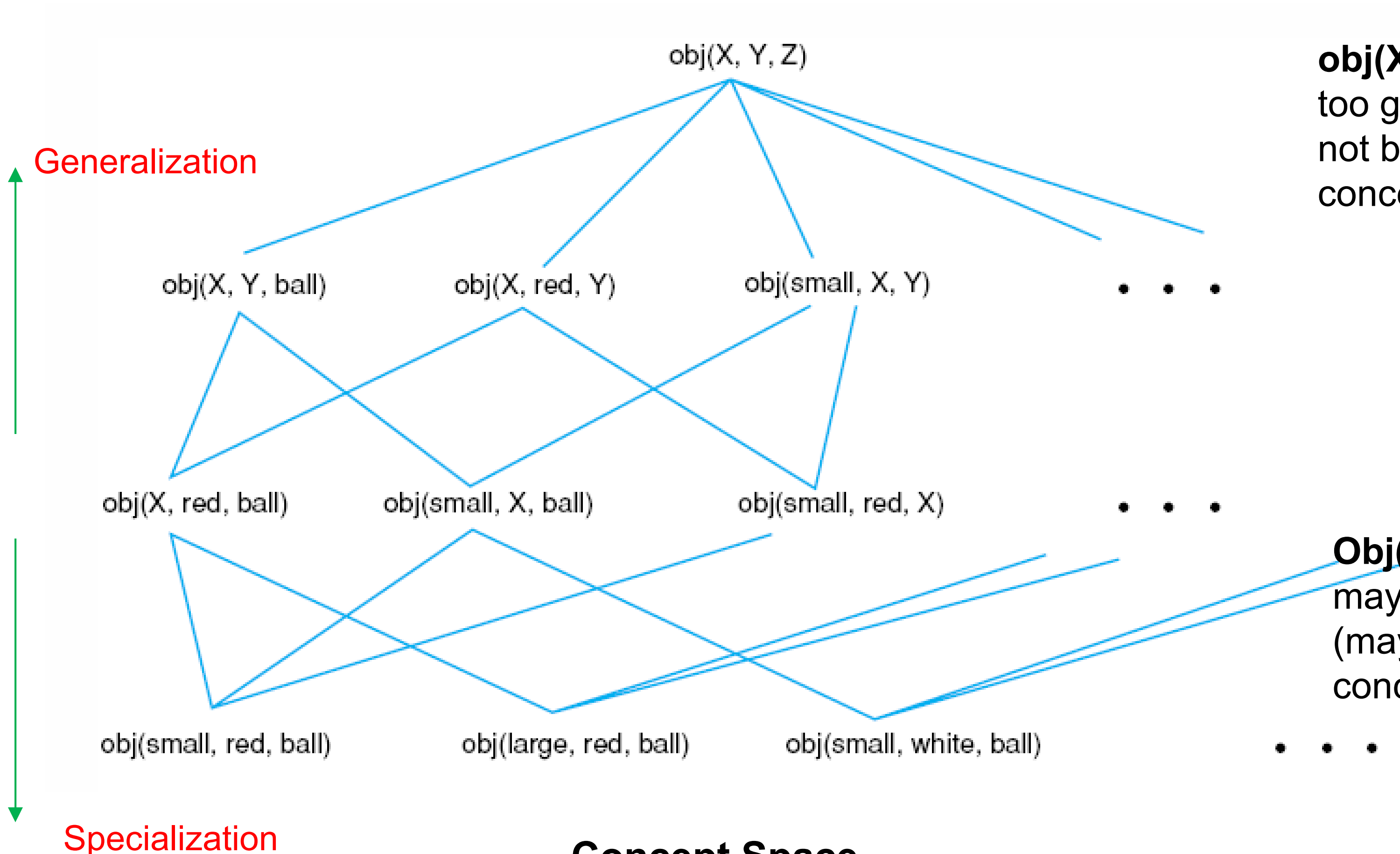
- Concept learning example
  - Learning *Ball concept* from the following ball objects:

    Object1: Ball(small, red, round)

    Object2: Ball(large, blue, round)

    generalized to a **ball** concept: **Ball**(Y, Z, round)
  - The **concept** "ball" can be described by (Y, Z, round), meaning that "Ball is a *round* object that has *variable sizes and colors*."
  - There are several other ways to define "ball" concept.
- If **concept** p is more **general** than **concept** q, we say that p covers q, $p \supseteq q$. Or p covers q iff q(x) is a logical consequence of p(x).
- Concept space is a set of concepts that can be created by a learning method.
  - A learning algorithm is to search for the correct concept in the concept space. See the figure on next slide.

Generalization and Specialization in a Concept Space



obj(X,Y,Z) may be too general (may not be useful concept).

Obj(small, red, ball) may be too specific (may not be a useful concept).

Concept Space

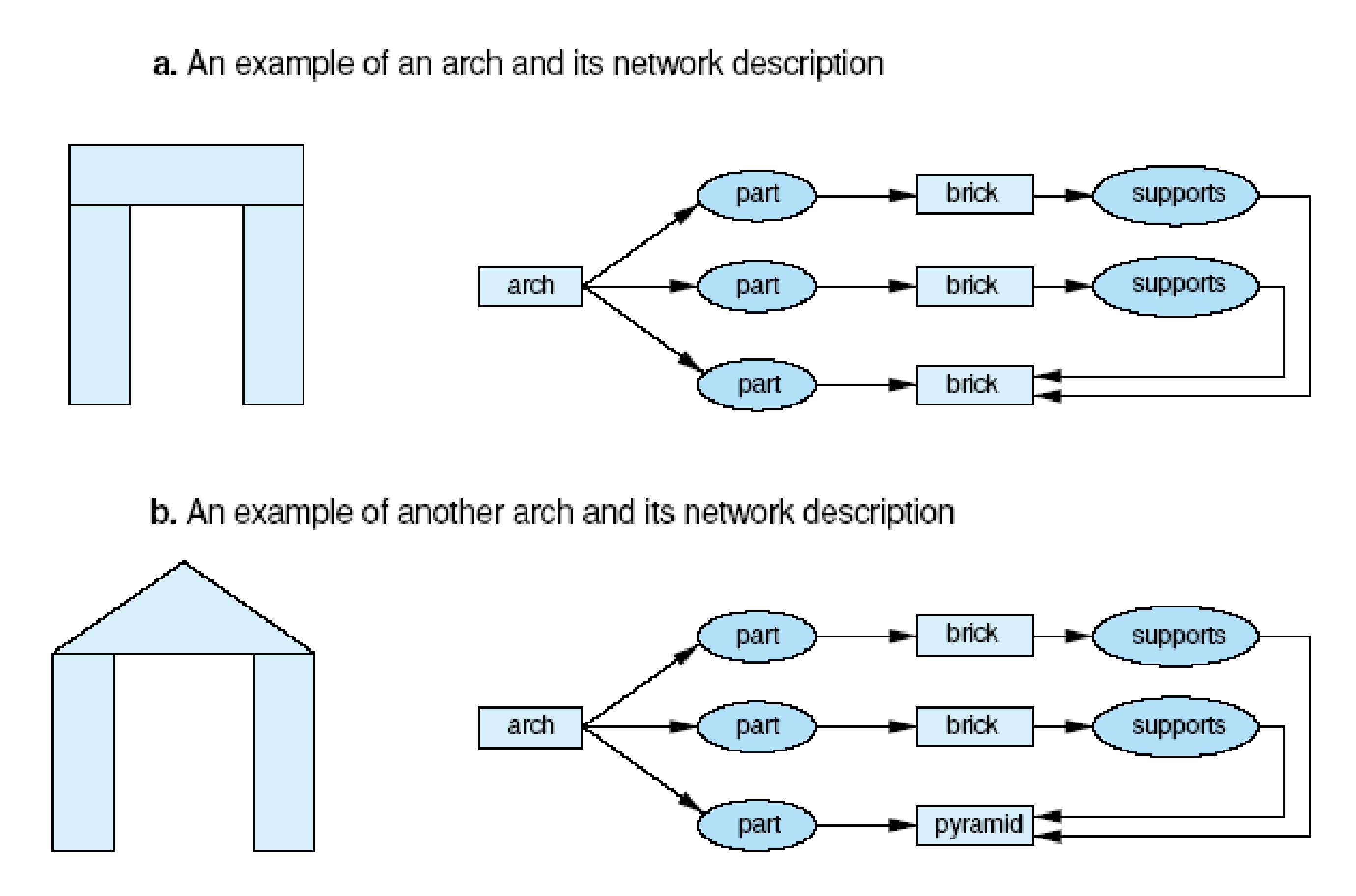
How to Develop a Learning System

- How can we develop a software system that can **learn** the **concept** of **Ball**?
  - What are the **learning methods** and **concept (knowledge) representation scheme** for this learning system?
- **For more complex learning problems**
  - How can we develop a system that can recognize my face?
  - How can we develop a system that can recognize my voice?
  - How can we develop a system that can learn the winning strategies in a chess or Go game?
  - How can we develop a system that can learn the patterns of good credits and bad credits?
  - How can we develop a system that can learn the patterns of successful investment?

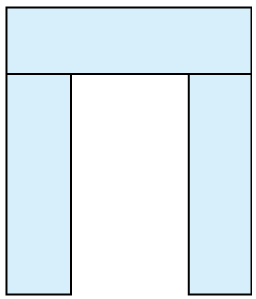
Winston's Learning Program (Winston, 1975) used:

Generalization and *Specialization* *as* learning methods and Semantic Networks for concept representation

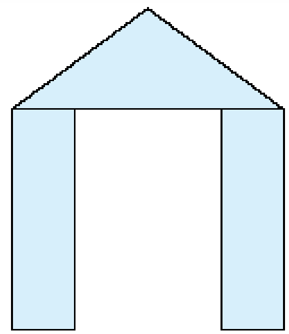
Learning goal in this example: to learn "Arch" concept



Positive and Negative Examples in Learning



Arch



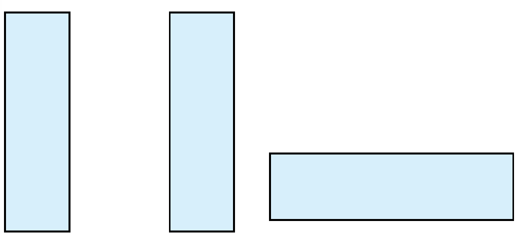
Arch

Positive examples: **arches**

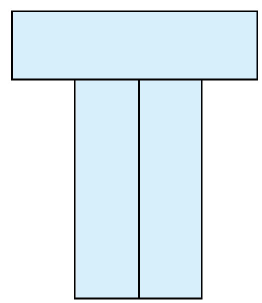
+Use generalization method

+**<u>Correct</u> arch concept** should cover all positive examples or include patterns of all positive examples for arches.

Training data consisting of both positive and negative examples, needed for training the learning algorithm



Near miss



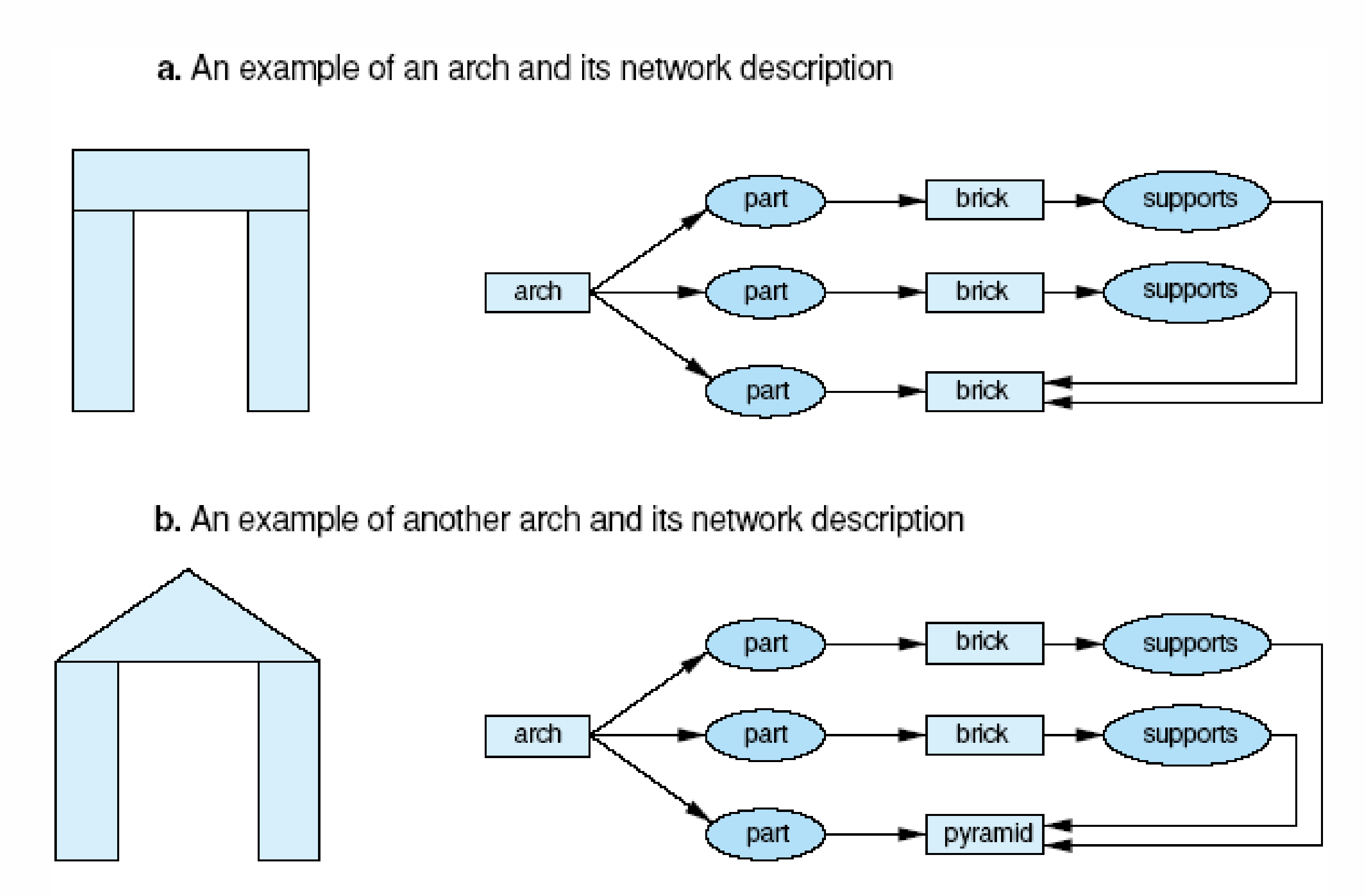
Near miss

Negative examples (also called **near miss**): **not arches**

Use specialization method

<u>Correct</u> arch concept should cover none of the negative examples or exclude patterns of all negative examples.

What is a Generalized Arch Concept from Two Positive Examples?

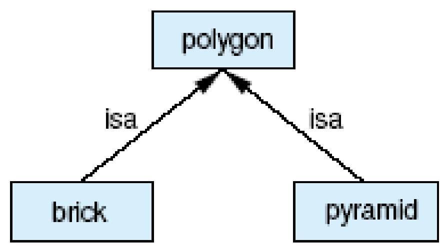


Use generalization method with positive examples:

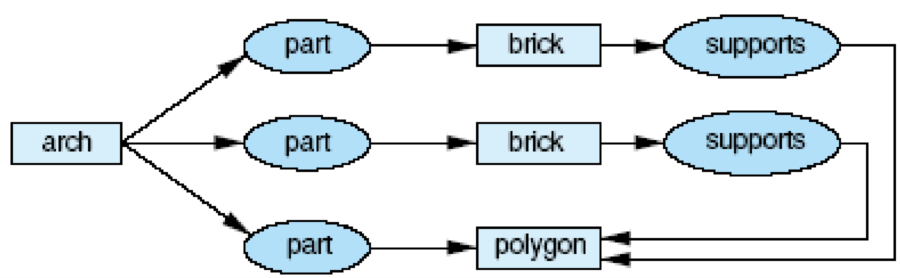
Again, **correct arch concept** should cover all positive examples or include patterns of all positive examples of arches.

Generalization to Cover all Positive Examples

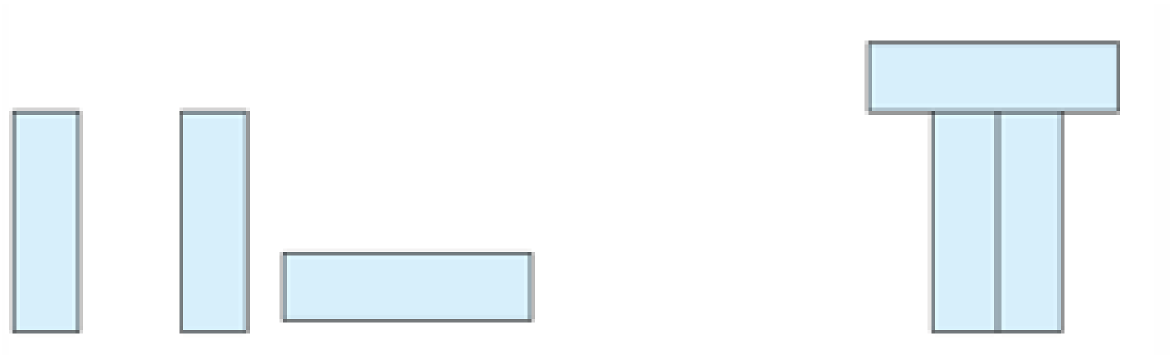
c. Given background knowledge that bricks and pyramids are both types of polygons



d. Generalization that includes both examples



How is Arch Concept **Specialized** from Two **Negative** Examples?



Negative examples: **not arches**

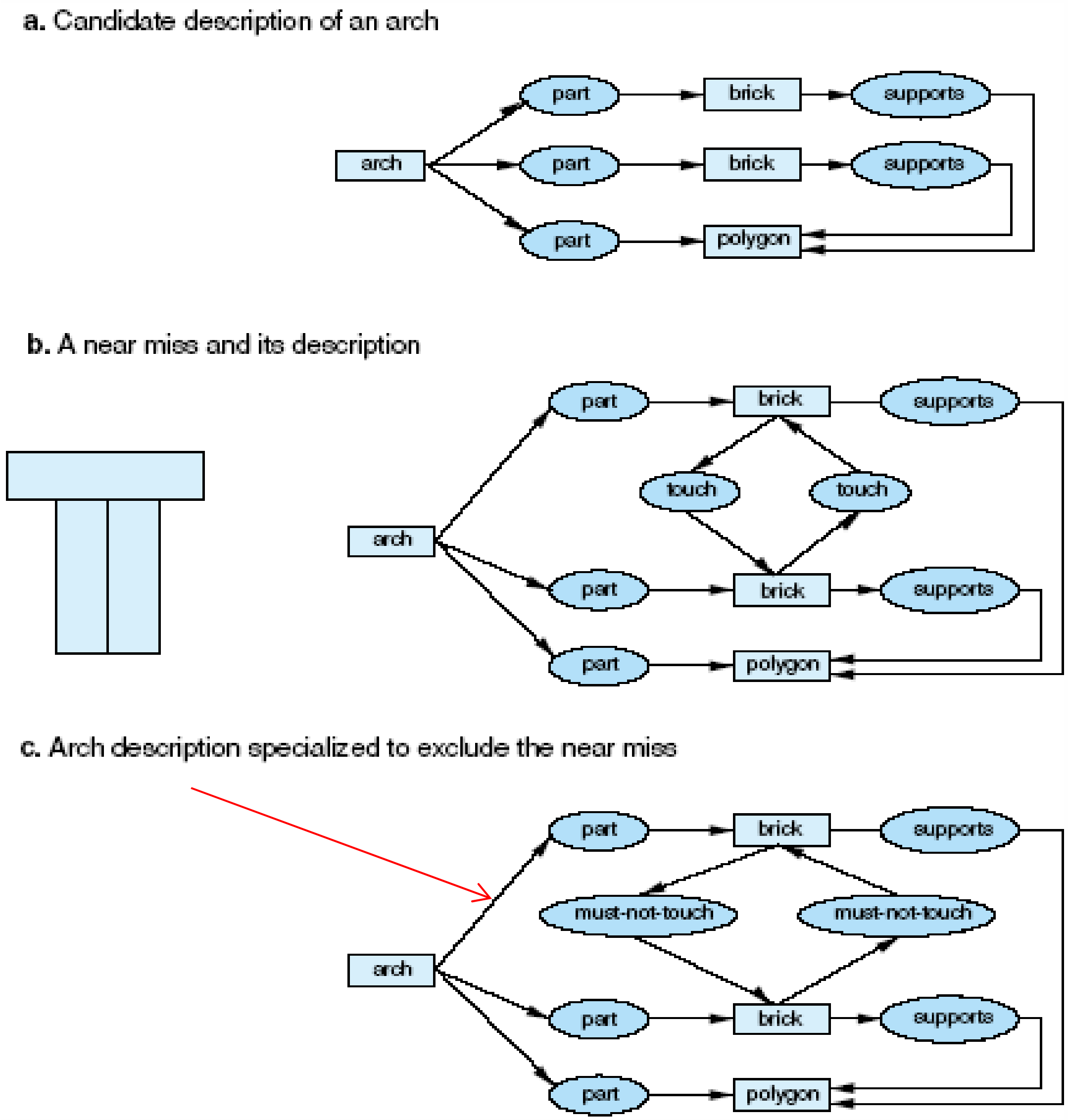
+Use specialization method

+Correct arch concept should cover none of the negative examples or exclude patterns of negative examples.

Specialization to Exclude Negative Examples

*Adds constraints to the semantic network so that it can't match with negative examples.

*Negative example helps the learning algorithm to prevent over-generalization of a concept or concept refinement.



The Role of Negative Examples in Preventing Overgeneralization

Concept induced from positive examples only

Concept induced from positive and negative examples

Lessons Learned from Winston's Learning Program

- **Generalization** and **specialization** operations can be used to learn a concept.
  - Can be considered as a *conceptualization* process
- Need **positive examples** to generalize a concept and **negative examples** to prevent overgeneralization
  - Generalize as little as possible to cover all positive examples AND Specialize as little as possible to exclude all negative examples to eliminate overly general concepts.
    - High quality concept can be obtained from somewhere in between.
- **Problem of Winston's learning program**
  - Seen as a hill climbing search, with no backtracking on the concept space, that is guided by examples in the training data.
    - So learning performance is highly sensitive to the order and quality of the training data.

Predicting Rock-Paper-Scissor

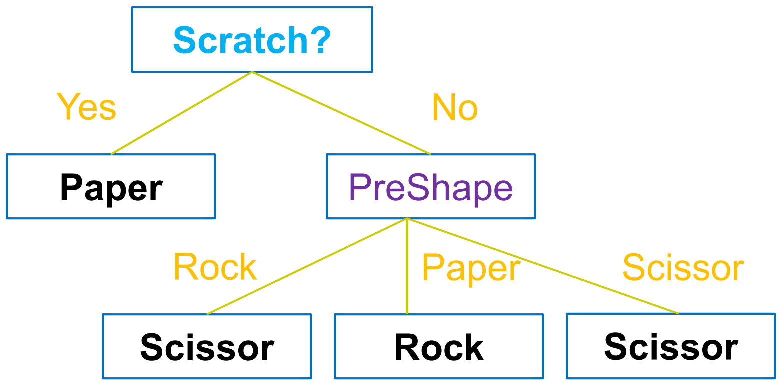
Data collected from **Past** John's Plays

ID	PreviousShape	Scratch?	Decision
1	Paper	Yes	Paper
2	Rock	No	Scissor
3	Paper	No	Rock
4	Scissor	Yes	Paper
5	Scissor	No	Scissor
6	Rock	Yes	Paper

- Which shape will John play next?
- How can we tell John's next shape?

Use of a Decision Tree

Can we construct a decision tree from a data set so that we can decide or classify John's play?



- A decision tree is a popularly used graph model for decision analysis in business, operation research, etc.
- Internal nodes represent tests on attributes.
- Branches are attribute values.
- Leaf nodes represent decisions.

- What are the rules to decide John's play?
- Is there any systematic way to construct a decision tree?

Decision Tree Induction

- **Quinlan** (1986) developed one of the earliest learning algorithms using decision trees called **ID3**.
  - Uses *Decision Trees* as *knowledge representation*
  - Learning method*: decision tree induction using entropy theory*

  - **Input**: Training Data
    - Data to *train* or guide the algorithm to learn the possible rules
  - ***ID3 induces*** *a Decision Tree from a data set*
  - **Output**: a Decision Tree branched into conditional expressions
    - The conditional expressions can be converted to a set of rules.

Problem: How Can we Tell the Risk Level for Each Loan Application?

Risk Level?

CREDIT HISTORY	DEBT	COLLATERAL	INCOME
bad	high	none	$0 to $15k
unknown	high	none	$15 to $35k
unknown	low	none	$15 to $35k
unknown	low	none	$0 to $15k
unknown	low	none	over $35k
unknown	low	adequate	over $35k
bad	low	none	$0 to $15k
bad	low	adequate	over $35k
good	low	none	over $35k
good	high	adequate	over $35k
good	high	none	$0 to $15k
good	high	none	$15 to $35k
good	high	none	over $35k
bad	high	none	$15 to $35k

A Data Set collected from loan applications

Risk level can be:

- High risk
- Moderate risk
- Low risk

Problem: Learning the Risk Level from a Loan Application Data Set

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	$0 to $15k
2.	high	unknown	high	none	$15 to $35k
3.	moderate	unknown	low	none	$15 to $35k
4.	high	unknown	low	none	$0 to $15k
5.	low	unknown	low	none	over $35k
6.	low	unknown	low	adequate	over $35k
7.	high	bad	low	none	$0 to $15k
8.	moderate	bad	low	adequate	over $35k
9.	low	good	low	none	over $35k
10.	low	good	high	adequate	over $35k
11.	high	good	high	none	$0 to $15k
12.	moderate	good	high	none	$15 to $35k
13.	low	good	high	none	over $35k
14.	high	bad	high	none	$15 to $35k

A Data Set collected from loan applications

How can we define the Risk Level concept from this data set?

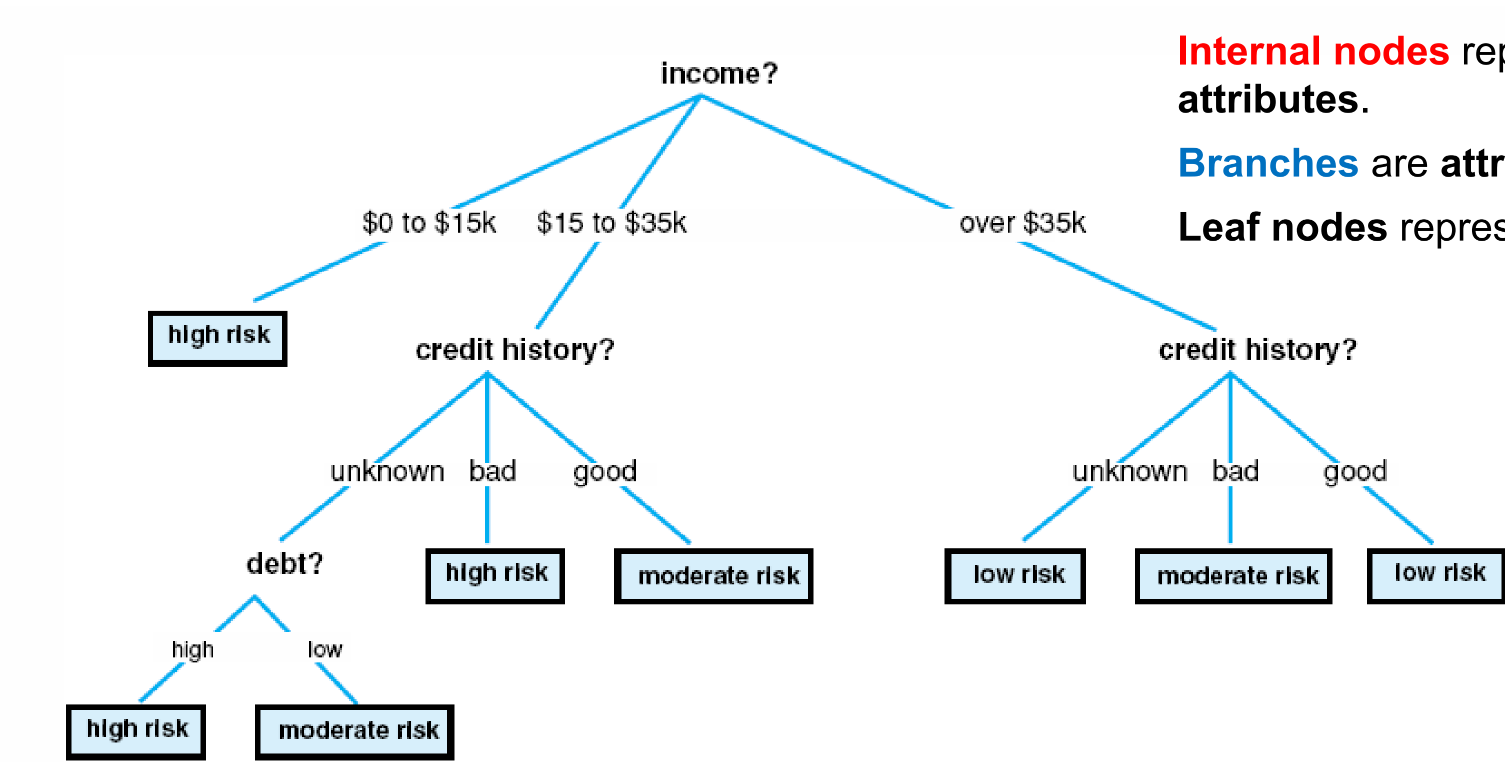
If risk level concept can be properly defined by **attributes**, credit history, debt, collateral, and income, then the loan approval process can be easier (by classification of each application).

Example Definitions for the Risk Level and Classification of Loan Applications

- The Concept of Risk Level can be:
  - Three classes: ***High***, ***Moderate***, ***Low***
- How can each risk level class be defined? What does each class mean?

- The **risk level concept** can be defined by conditional expressions based on the related attributes
  - **High Risk Level Class**: *Bad credit history ^ High debt ^ Low income*
  - **Medium Risk Level Class**: *Unknown credit history ^ Low debt*
  - **Low Risk Level Class**: *Good credit history ^ Low debt ^ High income*
- How can we classify each loan application to one of three risk level classes?
  - *This is a classification problem.*

Decision Tree for Risk Level Concepts



Internal nodes represent tests on **attributes**.

Branches are **attribute values**.

Leaf nodes represent decisions.

- Can we systematically construct this decision tree from a training data set so that we can classify the loan applications?

Training Data Set for Learning

Decision Class **Related Attribute Values**

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	unknown	low	adequate	over \$35k
7.	high	bad	low	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

Historical Data collected from loan applications

Why Training Data Set?
Training data will be used as a guide for learning to discover relationships between classes and attribute values.

How can we create a decision tree from this?

ID3: Decision Tree Induction Algorithm

```
function induce_tree (example_set, Properties)

begin
if all entries in example_set are in the same class
   then return a leaf node labeled with that class
   else if Properties is empty
      then return leaf node labeled with disjunction of all classes in example_set
      else begin
         select a property, P, and make it the root of the current tree;
         delete P from Properties;
            for each value, V, of P,
               begin
                  create a branch of the tree labeled with V;
                  let partition_V be elements of example_set with values V for property  P;
                  call induce_tree(partition_V, Properties), attach result to branch V
               end
      end
end
```

Major Steps for Creating a Decision Tree

1. **Select** an attribute as the current node.
   - How? Which one first?
2. **Use** the **values of the selected attribute** to **partition** the training data set into **subsets**.
3. **Check** if all members in a subset belong to the **same decision class**.
   - If YES, the subset becomes a **leaf node** labelled by its **decision class**.
   - If NO, continue recursively for a subtree of each partition following steps 1 ~ 3.
4. **Perform** the steps 1 ~ 3 for all other branches until all subsets are leaf nodes.

Example: Creating a Decision Tree

Training Data collected from **Past** John's Plays

ID	PreviousShape	Scratch?	Decision
1	Paper	Yes	Paper
2	Rock	No	Scissor
3	Paper	No	Rock
4	Scissor	Yes	Paper
5	Scissor	No	Scissor
6	Rock	Yes	Paper

- Try to create a decision tree from the above data set.
- Try to create a decision tree from the *loan application data set*.

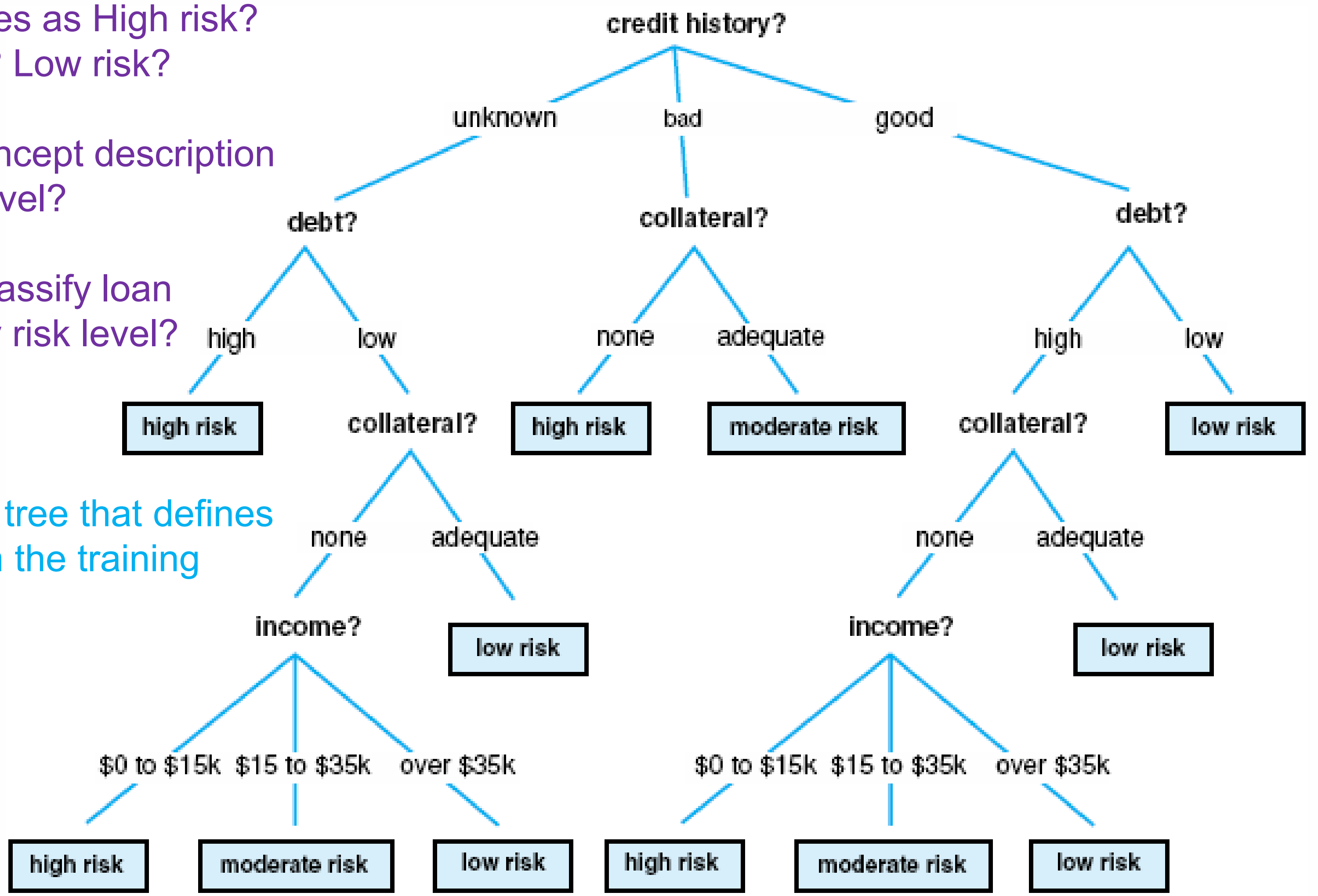
A Decision Tree for Risk Level Assessment

What constitutes as High risk? Moderate risk? Low risk?

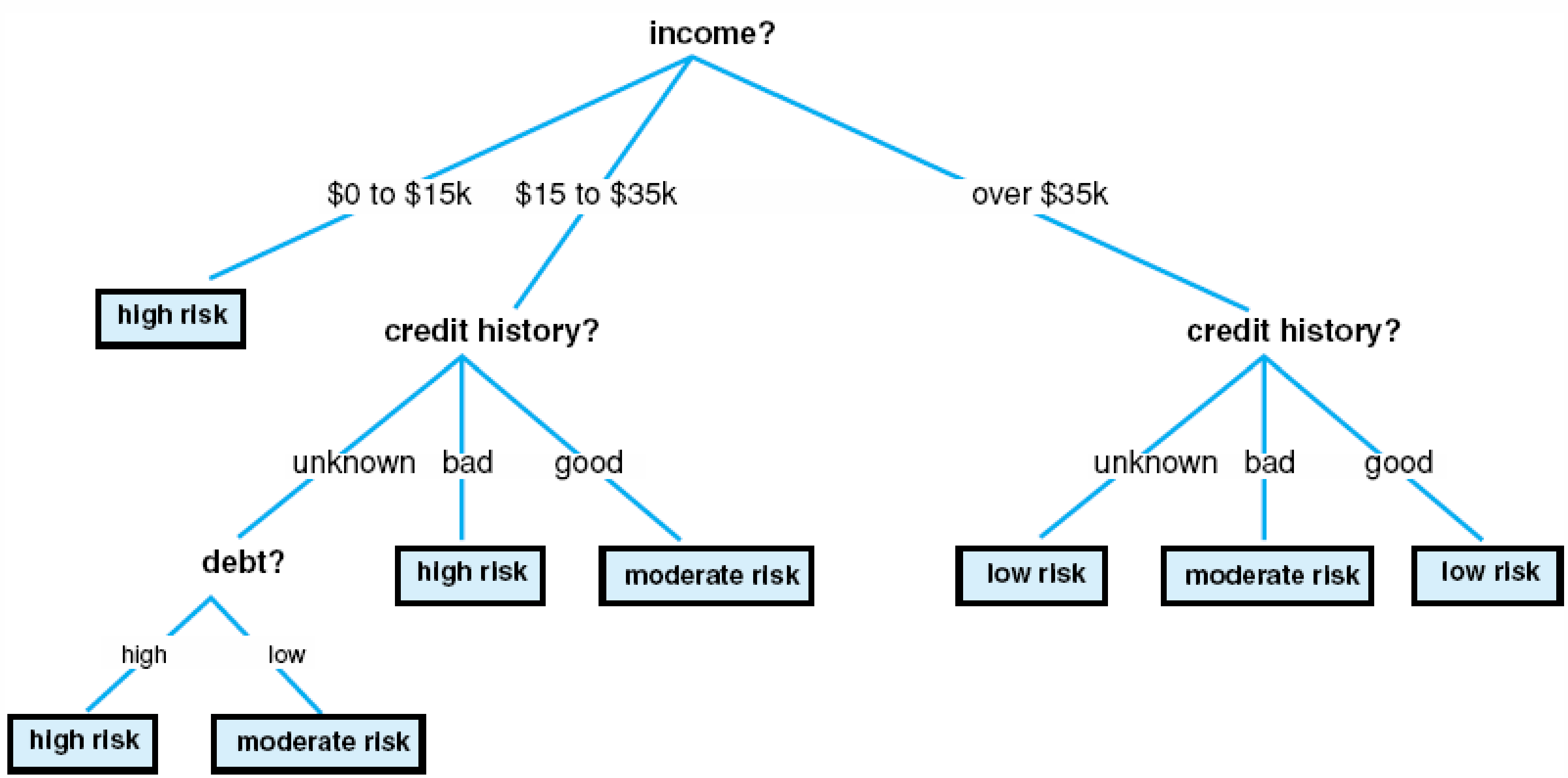
What is the concept description for each risk level?

How can we classify loan applications by risk level?

Is this the only tree that defines risk levels from the training data set?



A Simpler Decision Tree



If multiple trees can be created from the same data set, which tree would you use and why?

Information Theory that Supports Simpler Trees

- A pattern needs to be "simpler" than listing out the data set it describes.

- **Minimal Message Length** (MML) theory
  - Given a data set T
  - Given a set of **hypotheses** H_1, H_2, .., H_t that describe T
    - A hypothesis H_i can be a **concept**, classifier, cluster, theory, etc.

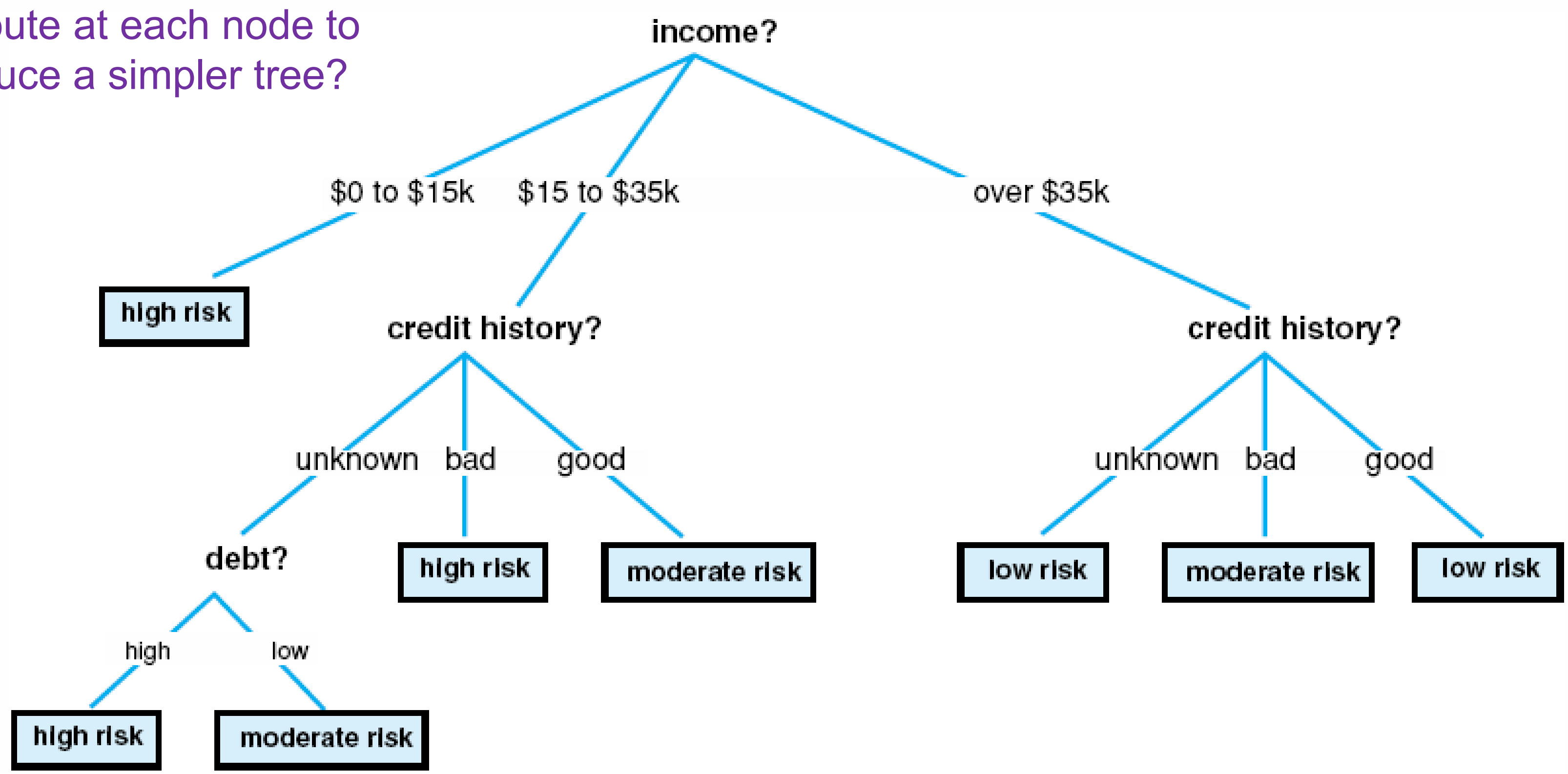
  - MML principle states that:
    - We should choose $\mathbf{H_i}$, for which the quantity: **Mlength(H_i) + Mlength(D|H_i)** is minimized, where Mlength(H_i) is minimum number of bits needed to specify H_i, and Mlength(T|H_i) is minimum number of bits needed to describe the data given that H_i is true.

Example: MML and Simpler Concepts

- **Consider** a data set T = {2, 4, 6, …, 100}
  - **H1**: List all 49 integers:
    - **cost of H1** = cost of listing 49 integers
  - **H2**: “All even numbers <= 100 AND >= 2”:
    - cost of H2 = cost of listing 2 integers + cost of “<=“ + cost of “even” concept
  - According to **MML**, if *cost(solution1) > cost(solution2)*, then *solution2* is worth something since as size of data set increases, the cost of *solution1* increases much more rapidly than *solution2*.
  - So we prefer H2 for the data set.
- **Simpler concept covers all examples** and is the **least likely to include unnecessary constraints**.

Creating a Simple Decision Tree

How can we construct a simpler tree? Or how do we choose the attribute at each node to produce a simpler tree?



Entropy Concept in Information Theory

- **Entropy is a measure of the uncertainty in a random variable**.
  - Probability can be used to measure uncertainty.
  - Uncertainty level can be quantified in binary bits by taking logarithm of the probability, **$-\log_2 p$**.
    - For example, for a random variable X, if the probability of having a value x_i is **0.5**, the uncertainty on whether or not X will have the value x_i is calculated:

      $$-1*(\log_2 0.5) = -1(-1) = 1$$

    - It also means that we need **1 bit of resource** to describe this uncertainty.
- If a random variable X can take a value x_1 with probability p_1, value x_2 with probability p_2, …, x_n with probability p_n, the expected value of X, **E(X)** is calculated:

  $$E(X) = p_1x_1 + p_2x_2 + \ldots + p_nx_n$$ (weighted average or weighted sum)

- **Shannon's entropy** **quantifies** the ***expected uncertainty*** of the information contained in a **message** in **binary bits**.

Shannon's Entropy

- For a random variable X with n outcomes ***{x_1, ..., x_n}***, the Shannon's entropy **H(X)** is defined:
  - **Entropy of X,** $H(X) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i)$ where *p(x_i)* is the *probability* of outcome x_i
  - **H(X)** is the average/expected unpredictability for a random variable X.
- The more unpredictable the outcome is (increased uncertainty), the larger entropy is, resulting in more bits required to describe the outcome. (The opposite case is true.)

Example: Shannon's Entropy -1

- **Assuming that fair coin tossing as a random variable X, what is H(X)?**
  - For the coin tossing, there are two **possible outcomes**, h(head) or t(tail).
  - What is the **probability** of seeing h, p(h) or t, p(t)?
  - How **many bits** do we need to **describe** the outcome of the coin tossing action?
    - **Use** $\mathbf{H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)}$ where X is coin toss; $p(x_i)$ is the probability of outcome, h or t.
    - **H(coinToss)** = $-p(h)\log_2 p(h) - p(t)\log_2 p(t) = -1/2\log_2(1/2) - 1/2\log_2(1/2) = 1$ bit
    - **One bit** is required to convey the message of coin tossing.

Example : Shannon's Entropy -2

- **Assuming that rigged coin tossing as a random variable X, what is H(X)?**
  - With the **rigged coin**, the probability of seeing heads is **75%**. Now we get to know more about the coin. What is H(coinToss)?
    - **H(coinToss)** $= -3/4\log_2(3/4) - 1/4\log_2(1/4) = 0.811$ bits.
  - What's the entropy of a coin toss with a coin that has two heads without any tail?
- **English text** has fairly low entropy (fairly predictable), 0.6~1.3 bits of entropy for each character of a message.

Shannon's Entropy to Measure the Uncertainty of Classes in a Data Set

- Consider a data set **T**, that contains data classified by a set of classes, **C**.
- To calculate the **expected amount** of information in **bits needed** to determine a class each instance/example in T belongs to, we calculate H(T).
  - $H(T) = -\sum_{i=1}^{n} p(C_i) \log_2 p(C_i)$
    - is the average amount of information **needed** to identify the class of an example in **T**.
  - When **H(T)** = 0, the data set **T** is perfectly classified (all instances of **T** are of the same class).

Information Theoretic Test Selection to Build a Simple Decision Tree

- Aiming to create a simpler tree, choose the attribute for each node that results in the smallest entropy.
  - Compute the **expected amount of information needed** E(X) after T is **divided** into the **m** possible subsets **{v_1, v_2, …, v_m}** by the values of attribute **X**. ***$E(X) = \Sigma(|v_i|/|T|)*H(v_i)$***
    - **$|v_i|$** and **$|T|$** are the number of instances that belong to the set **v_i** and **T**, respectively.
    - **$H(v_i)$** is entropy of ***v_i*** calculated in terms classes, **C**.
  - The information gain by selecting he attribute X is computed by

    gain(X) = H(T) – **E(X)**
  - **ID3** chooses the **attribute** that provides the **greatest information gain**, meaning the **smallest E(X)** by "simplicity rule".
  - The amount of information needed to complete the tree can be defined as the weighed average of the information in all its subtrees.
- *The more information/knowledge we have, the less bits we need* to describe it!

Example: Calculating Entropy

Data collected from **Past** John's Plays

ID	PreviousShape	Scratch?	Decision
1	Paper	Yes	Paper
2	Rock	No	Scissor
3	Paper	No	Rock
4	Scissor	Yes	Paper
5	Scissor	No	Scissor
6	Rock	Yes	Paper

- What is the entropy for this data set?
- Which attribute should be selected as the root node for a decision tree?

Example: Calculating Entropy

- **Training Data Set Entropy** for **Loan Applications**, **H(LoanData)** is calculated using

$$H(T) = -\sum_{i=1}^{n} p(C_i)\log_2 p(C_i)$$

  - **H(LoanData)** = $-(6/14)\log_2(6/14) - (3/14)\log_2(3/14) - (5/14)\log_2(5/14)$
    = 1.531 bits.

- **Choose the best attribute for root node**, after computing information gain by each attribute, Income, Credit history, Debt, and Collateral.

Training Data Set for Learning

Decision Class

.

Related Attribute Values (Given)

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	unknown	low	adequate	over \$35k
7.	high	bad	low	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

Training Data collected from loan applications

Example: Calculating Expected Information Needed to determine the Risk Level

- Let's calculate **information gain** by **Income**. **Subsets** by **income** are:

  V_1 = *\$0 ~ 15k* = {1,4,7,11}
  V_2 = *\$15 ~ 35k* = {2,3,12,14}
  V_3 = *>\$35k* = {5,6,8,9,10,13}   See next slides for partitioned data set.

  - Calculate each Subset Entropy by Income

    $H(V_i) = -\Sigma p(C_i)\log_2 p(C_i) = -[p(high)\log_2 p(high) + p(mod)\log_2 p(mod) + p(low)\log_2 p(low)]$

    $H(V_1) = -[1*\log_2(1) + 0*\log_2(0) + 0*\log_2(0)] = 0.0$ since all in "high" class
    $H(V_2) = 1.0$
    $H(V_3) = 0.65$

  - Compute **Expected Information** needed for each Subset by Income

    $E(X) = \Sigma(|v_i|/|T|)*H(v_i)$

    where |T| is total # of examples in Training Data Set T and
    $|v_i|$ = total # of examples in a subset

    $|v_1|/|T| = 4/14$, $|v_2|/|T| = 4/14$, $|v_3|/|T| = 6/14$

    E(income) $= 4/14*H(V_1) + 4/14*H(V_2) + 6/14*H(V_3)$
    $= 4/14*0.0 + 4/14*1.0 + 6/14*0.65 =$ **0.564bits**

Training Data Set Partitioned by Income

Decision Class

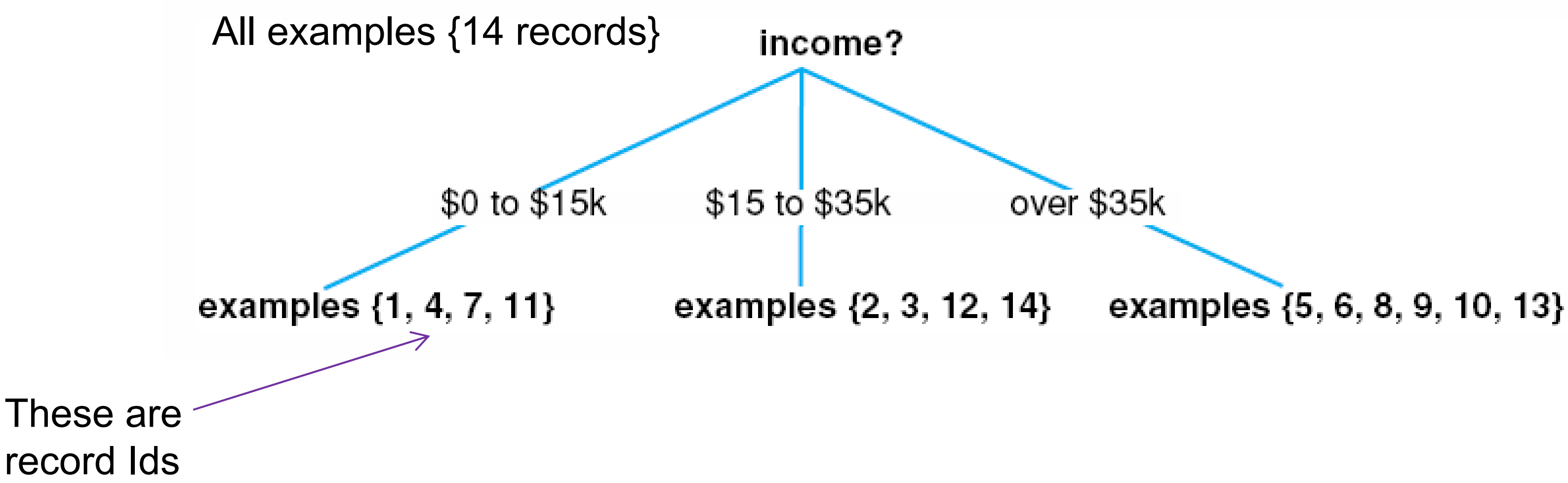
.

Related Attribute Values (Given)

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	$0 to $15k
2.	high	unknown	high	none	$15 to $35k
3.	moderate	unknown	low	none	$15 to $35k
4.	high	unknown	low	none	$0 to $15k
5.	low	unknown	low	none	over $35k
6.	low	unknown	low	adequate	over $35k
7.	high	bad	low	none	$0 to $15k
8.	moderate	bad	low	adequate	over $35k
9.	low	good	low	none	over $35k
10.	low	good	high	adequate	over $35k
11.	high	good	high	none	$0 to $15k
12.	moderate	good	high	none	$15 to $35k
13.	low	good	high	none	over $35k
14.	high	bad	high	none	$15 to $35k

Training Data collected from loan applications

Training Data Set Partitioned by Income



Example: Calculating Information Gain

- **Subsets by Income**:

  V1 = *\$0 ~ 15k* = {1,4,7,11}
  V2 = *\$15 ~ 35k* = {2,3,12,14}
  V3 = *>\$35k* = {5,6,8,9,10,13}

  - **H(LoanData)** = 1.531 bits
  - **E(income)** = 0.564bits

- **Information Gain** by choosing Income attribute

  gain(X) = H(T) – E(X)
  gain(Income) = **H(LoanData) – E(Income)**
  = 1.531 – 0.564 = **0.967 bits**

By partitioning the training data set by Income, we obtained some knowledge about risk levels.

Example: Calculating Information Gain

- *Information gain by choosing Income*
  - gain(Income) = 0.967 bits.
- Similarly we calculate information gain for all other attributes
  - gain(credit history) = 0.266
  - gain(debt) = 0.063
  - gain(collateral) = 0.206.
- For the root node, we choose the attribute **income** with the highest information gain of 0.967 bits.

- **Repeat the same procedure** on resulting subtrees after partitioning by Income. See next slide.
  - How to determine next node for first, **second**, and third branch?
  - Can we choose the same attribute as the parent node?

Training Data for Second Branch

Decision Class

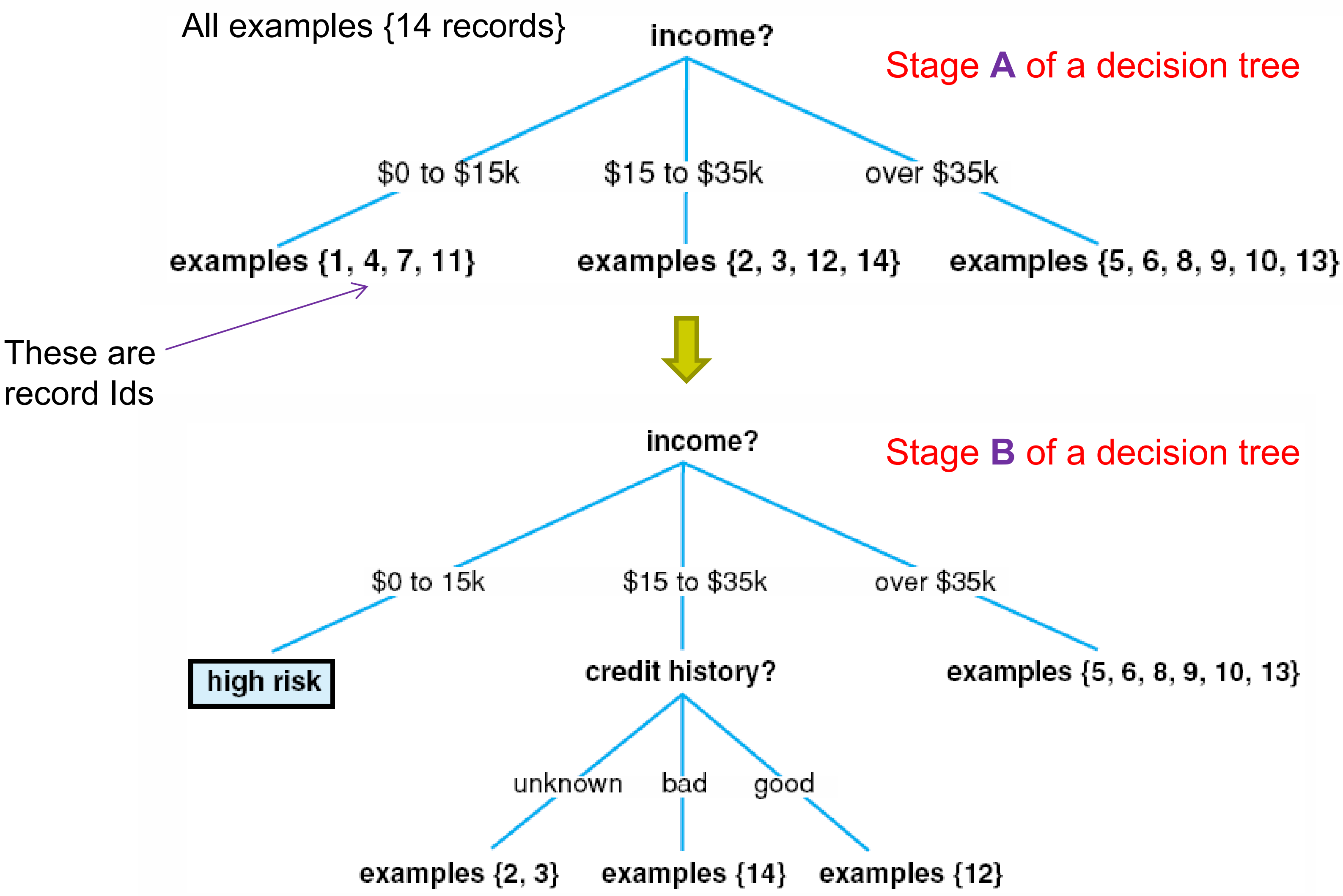
.

Related Attribute Values (Given)

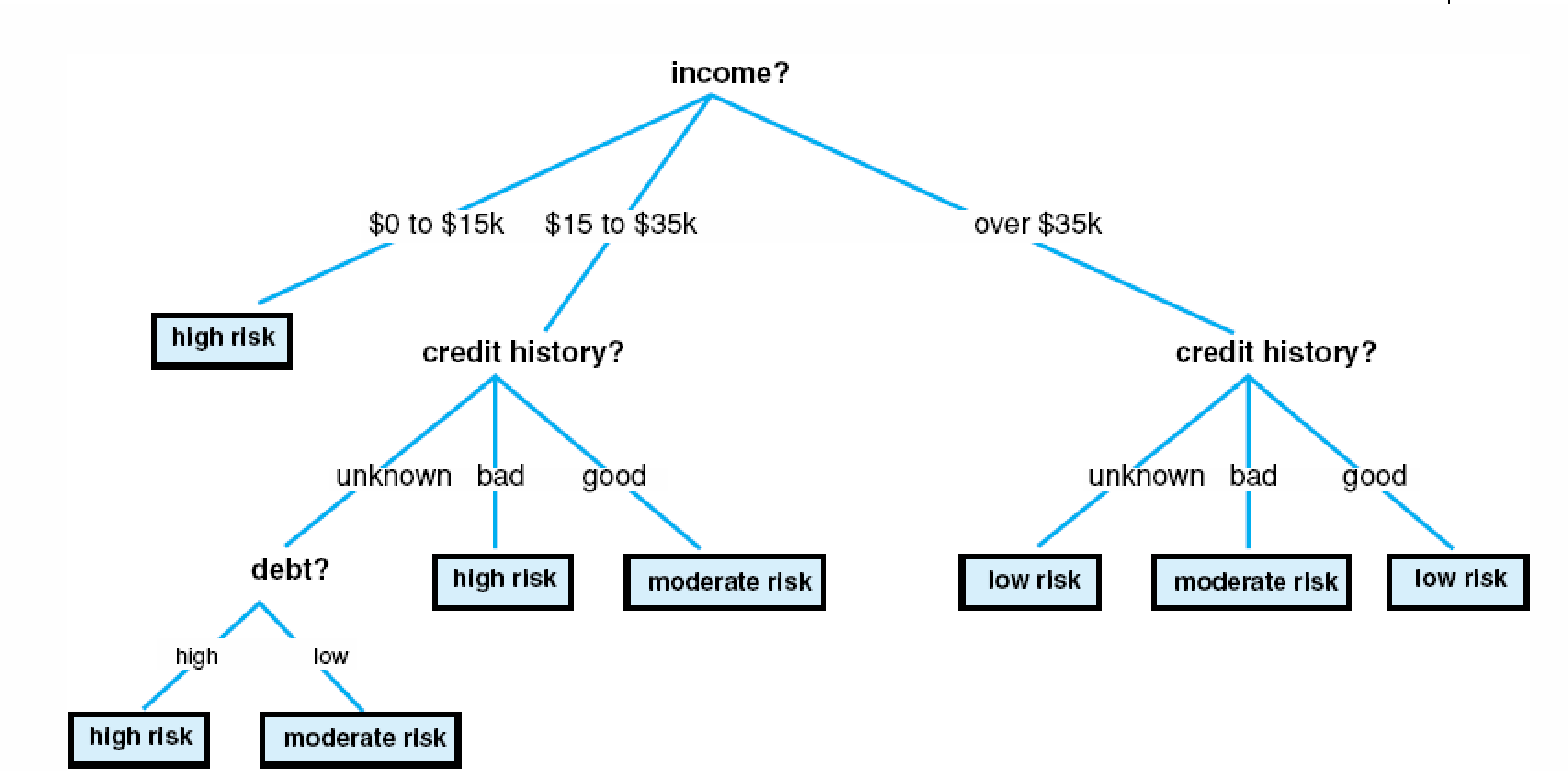
NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	$0 to $15k
2.	high	unknown	high	none	$15 to $35k
3.	moderate	unknown	low	none	$15 to $35k
4.	high	unknown	low	none	$0 to $15k
5.	low	unknown	low	none	over $35k
6.	low	unknown	low	adequate	over $35k
7.	high	bad	low	none	$0 to $15k
8.	moderate	bad	low	adequate	over $35k
9.	low	good	low	none	over $35k
10.	low	good	high	adequate	over $35k
11.	high	good	high	none	$0 to $15k
12.	moderate	good	high	none	$15 to $35k
13.	low	good	high	none	over $35k
14.	high	bad	high	none	$15 to $35k

Data collected from of loan applications

Partially Constructed Decision Tree



The Final Decision Tree Constructed



Risk Level Concepts Represented in Rules

- Concepts for classes can be represented in tree structure or in rules
  - Common rule formats
    - LHS (decisions leading to the leaf node) → RHS (class outcome)
    - IF (condition), THEN (class outcome)
- **Possible rules from the decision tree**

  High risk rule (for high risk concept)

  IF (income = $0~$15k) OR (income = $15k~$35k AND credit history = unknown AND debt = high) OR (income = $15k~$35k AND credit history = bad)

  THEN the application is considered as high risk

  Low risk rule (for low risk concept)

  IF (income > $35k) AND (credit history = unknown OR good)

  THEN the application is considered as low risk

  - How about moderate risk rule?

Decision Tree Induction as Search through a Concept Space

- Representation: Tree
  - Concepts represented in tree
  - Concept space is all possible decision trees that can be created from training examples.
- ID3 as a search algorithm
  - ID3 implements a form of greedy search in the space of all possible trees with no backtracking. Very efficient!

Evaluation of Decision Tree

- **Advantages**
  - Easy to convert a decision tree into a set of rules
- **Common issues**
  - Sensitive to the quality of data (poor performance with inconsistent data, missing data)
  - Problem of continuous values? Break the values into groups.
  - Large data set may produce a large tree.
- **C4.5 or higher version addressed many of these issues**
  - Bagging, Boosting
  - Handling large data set?
    - Divide the data into subsets, build the decision tree on one subset, and then test its accuracy on other subsets.
- Many variations exist, e.g., **CART**

Supervised Learning Process

- **Steps:**
  1. **Training** with both positive and negative examples (training data) to build a learning model (with patterns).
  2. **Testing** for verification of the concept learned

  3. **Use the model** for forecasting, prediction, recognition, etc.
  - Why do we call it "supervised learning"?
  - Why do we also call it learning by examples, inductive learning, classification?
- **Outcomes of supervised learning**
  - Description of learned classes represents patterns or a set of common properties shared among all classes of data.
- Some questions to think about
  - Can we learn multiple concepts?
  - Can we use Winston's learning algorithm for classification purposes?

Inductive Bias

- **Supervised learning** goes through a **induction process**
  - That's why supervised learning is also called inductive learning.
- Induction depends on prior knowledge and assumptions about the nature of the concepts being learned.
- Inductive bias refers to any criteria a learner uses to constrain the concept space or to select concepts within that space.
  - **Examples of inductive biases**
    - Heuristics
    - Syntactic constraints like bit representation, decision tree, feature vectors, rules, etc.
- **Reason for inductive bias**
  - Learning space is so large, that search-based learning becomes impractical.
  - Training data are only a *subset* of all instances in the domain. Data alone is not enough; it must make additional assumptions about "***likely***" concepts, which may be in the form of heuristics.
- Problem: Knowledge learned from induction may not be accurate!

Supervised Learning
vs.
Non-Supervised Learning

- What is the primary difference in learning method?
- What is the primary difference from the outcomes of the learning?
- **Some supervised learning approaches**
  - Winston's learning program, Decision tree induction, Version space learning, Support vector machine, Artificial neural network, etc.
- **Some unsupervised learning approaches**
  - Pattern recognition, Clustering, Expectation-Maximization (EM) algorithm, Artificial neural network, etc.

Review Questions

- What is learning? What is machine learning?
- Why is machine learning important in building an AI system?
- What is “**concept**”? Give some examples of concepts. Explain “meaning” and “knowledge”.
- How can we **represent** concepts or knowledge?
- Why is knowledge representation important in machine learning?
- What are the various learning methods?
- Why is simple memorization without understanding considered as very weak learning method or not very useful?
- How can “generalization” and “specialization” be used to learn a concept?
- What is the principle idea of supervised learning?
- What is the knowledge representation method used in Winston’s learning program?
- What is training data set? Why is it important in supervised learning or why do we need a training data set in learning? How can we create one?
- What’s the importance of having both positive and negative examples?

- If a concept is too general, why it may not be useful?
- If a concept is too specific, why it may not be useful?
- What does it mean by overgeneralization and overspecialization?
- When both concepts C1 and C2 are used to describe the same set of examples, if a concept C1 is more general than a concept C2, what does it mean? Will C1 be able to cover more positive examples?
- What is correct/accurate concept? How can we verify if a concept is correct?
- Why is a machine learning problem also considered as a search problem?
- What is the main limitation of Winston’s learning program?
- What is supervised learning? Why is it also called classification, inductive learning, or learning by examples?
- Who give the name of class or concept in supervised learning? Who is responsible for learning or identifying common patterns or properties shared among all objects in a training data set?
- What are the main steps needed for developing a supervised learning system?
- What are the input and output of supervised learning method?
- Can Winston’s learning program be used for classification?

- Is “class” in a training data set the same as a “concept”?
- How do we determine “class” column in a training data set?
- What is decision tree?
- What is the knowledge representation method used in the ID3 decision tree induction algorithm?
- What is a typical rule format? How can a decision tree be converted into a set of rules?
- How can one create a decision tree from a training data set? What are the steps needed to create a decision tree?
- Why do we prefer simple concept as opposed to complex concept when both have the same meaning? What are the advantages of acquiring simpler concepts?
- How could Quinlan construct a simple decision tree from a given training data set using the ID3 decision tree induction algorithm?
- What is Shannon’s entropy in information theory?
- How can we calculate the entropy for a data set?
- How can we use entropy to measure the uncertainty of a data set? What do we mean by “uncertainty” here?
- If the entropy of a data is zero, what does it mean?

- What do we mean by the statement “The more knowledge, the less entropy”?
- How can simply partitioning a data set by an attribute lower the entropy of the data set?
- What is information gain and how is information gain measured?
- How do we choose the right attribute for each node in building a decision tree by ID3? Try an example.
- What is the theoretical argument that the decision tree created through ID3 is simple (although not necessarily the simplest).
- How can we interpret the decision tree?
- How many rules can we create from a decision tree? What is the minimum number of rules? Define the minimum number of rules from a decision tree.
- What can be applications of decision tree induction?
- What are the pros and cons of the decision tree induction?
- What is inductive bias?
- Is it possible/easy to fully understand/learn a problem domain from a given training data set with 100% accuracy?
- If it is difficult to fully understand a problem domain, what can be the alternative way?

References

- George Fluger, Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 6th edition, **Chapters 10 and 12**, Addison Wesley, 2009.