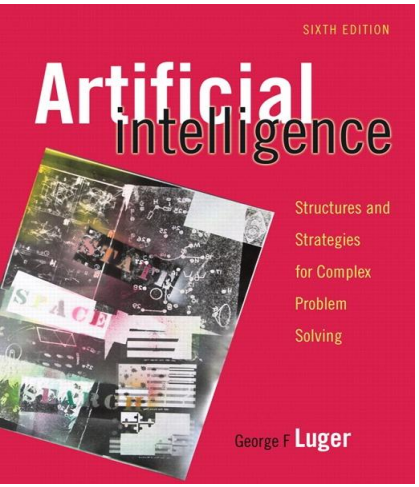
10 Machine Learning: Symbol-Based



George F Luger

ARTIFICIAL INTELLIGENCE *6th edition*

Structures and Strategies for Complex Problem Solving

什么是机器学习？

- 机器学习是研究如何使用计算机来模拟人类学习活动的一门学科。——研究计算机获取新知识和新技能、识别现有知识、不断改善性能、实现自我完善的方法。
- 学习是一个有特定目的的知识获取过程，其内在行为是获取知识、积累经验、发现规律，其外在表现是使系统性能得到改进、系统实现自我完善、自适应环境。
- 机器学习是智能系统不断地积累经验以改善系统性能的过程。

机器学习的方法

- 归纳学习
- 类比学习
- 基于解释的学习
- 遗传算法
- 人工神经网络
- 随机方法

10 MACHINE LEARNING: SYMBOL-BASED

10.0 Introduction

Section 10.1 outlines a variety of learning tasks.

Section 10.2 and 10.3 examine two algorithms used for concept induction ,version space search and ID3.

Section 10.4 considers the role of inductive bias in learning.

The section before 10.6 are supervised learning.

Section 10.6 continues the study of induction by examining unsupervised learning.

In 10.7,we present reinforcement learning.

10.1 A Framework for Symbol-Based Learning

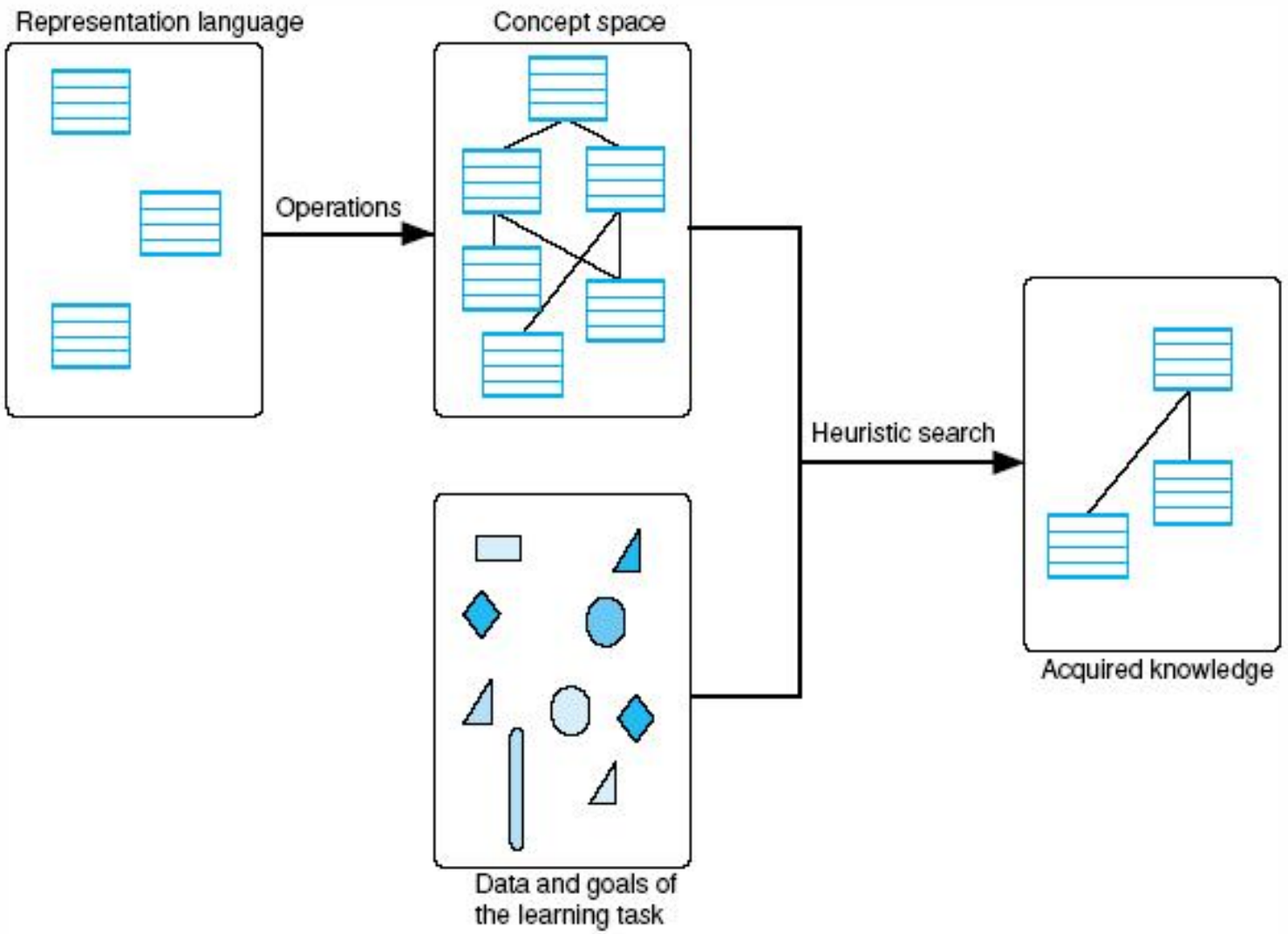


Fig 10.1 A general model of the learning process

1.The data and goals of the learning task.

The concept learning algorithms of section 10.2 and 10.3,begin with a collection of **positive examples** of a **target class**,the goal is to infer a general definition that will allow the learner to recognize future instances of the class.

Explanation-based learning(Section 10.5),attempts to infer a general concept from a single training example and a prior base of domain-specific knowledge.

These algorithms begin with a set of unclassfied instances in section 10.6.

2.The represention of learned knowledge

- For examples , two instances of a ball may be represented by:

Size(obj1,small) ∧color(obj1,red) ∧shape(obj1,round)

Size(obj2,large) ∧color(obj2,red) ∧shape(obj2,round)

The general concept of ball could be defined by:

Size(X,Y) ∧color(X,Z) ∧shape(X,round)

3.A set of operations.

Size(obj1,small) ∧color(obj1,red) ∧shape(obj1,round)

Replacing a single constant with a variable products the generalizations.

Size(obj1,x) ∧color(obj1,red) ∧shape(obj1,round)

Size(obj1,small) ∧color(obj1,x) ∧shape(obj1,round)

Size(obj1,small) ∧color(obj1,red) ∧shape(obj1,x)

Size(x,small) ∧color(x,red) ∧shape(x,round)

4.The concept space .

The representation language,together with the operations described above,defines a space of potential concept definitions.

5.Heuristic search.

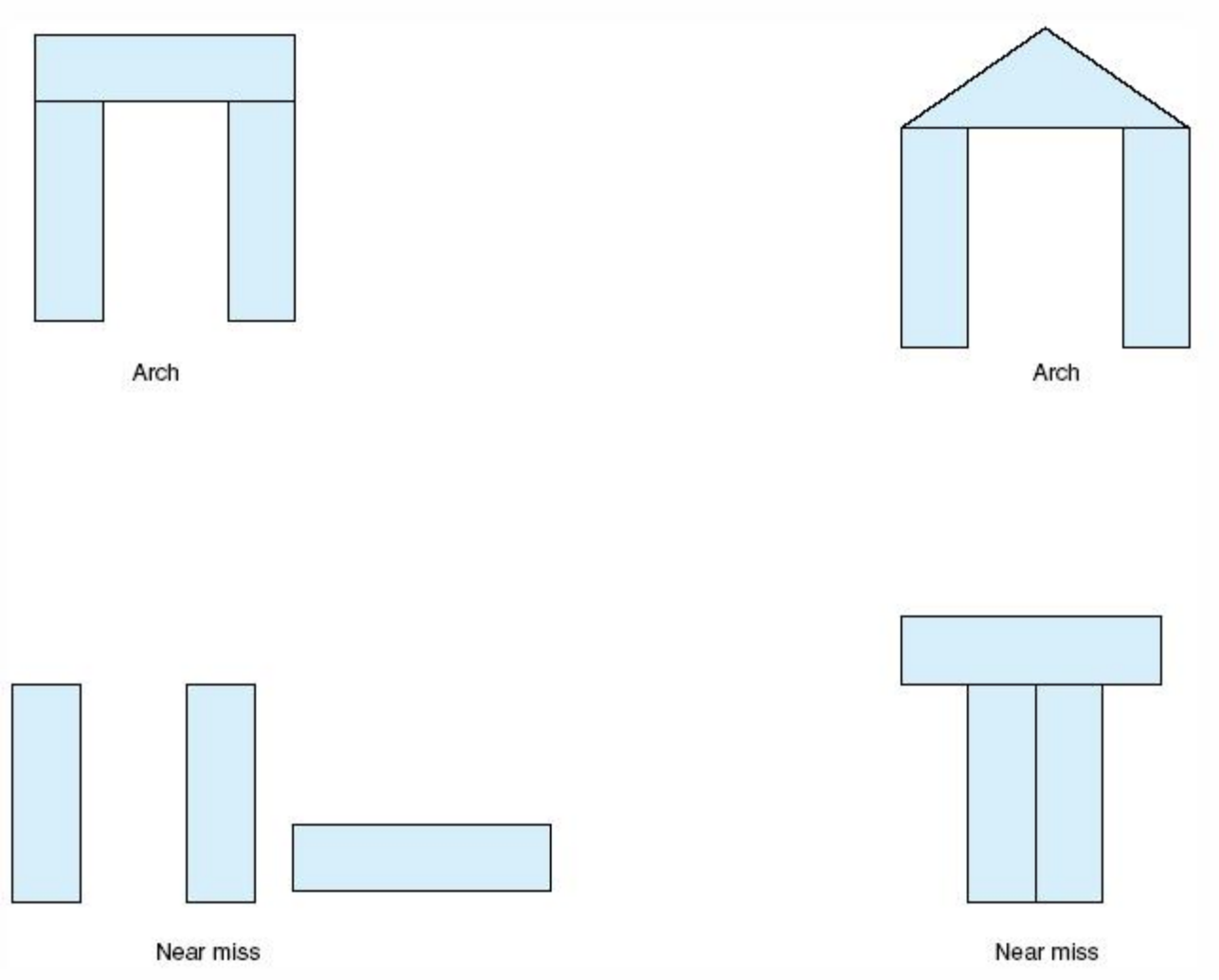
Size(obj1,small) ∧color(obj1,red) ∧shape(obj1,round)

Size(obj2,large) ∧color(obj2,red) ∧shape(obj2,round)

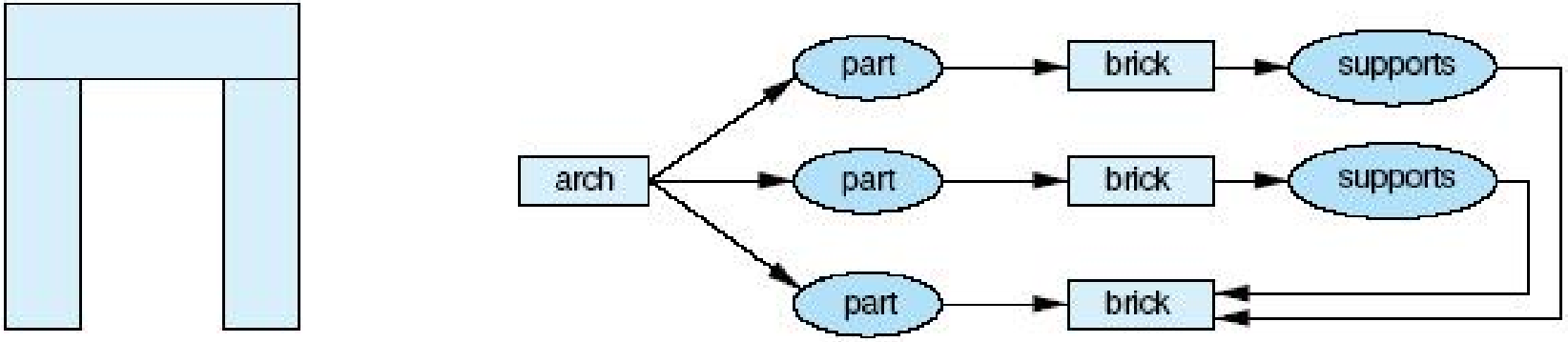
Size(X,Y) ∧color(X,red) ∧shape(X,round)

Patrick Winston's program learns general defininitions of structral concepts ,such as **"arch"**（拱形）, in a blocks world.The training data is a series of **positive and negative examples** （正例、反例）of the concept: examples of blocks world structures that fit in the category,along with **near misses**（小差别）.

Fig 10.2 Examples and near misses for the concept “arch.”



a. An example of an arch and its network description



b. An example of another arch and its network description

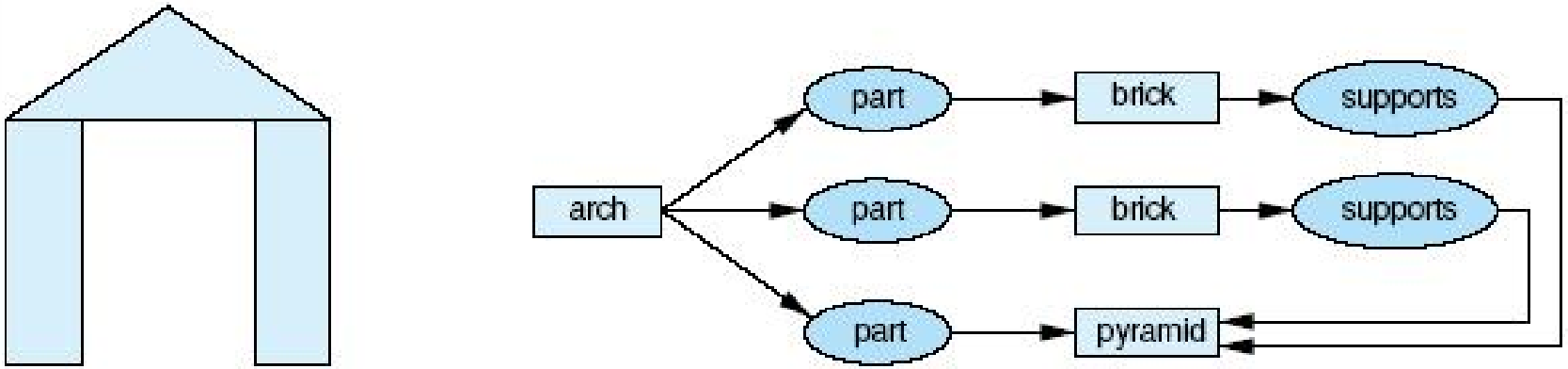
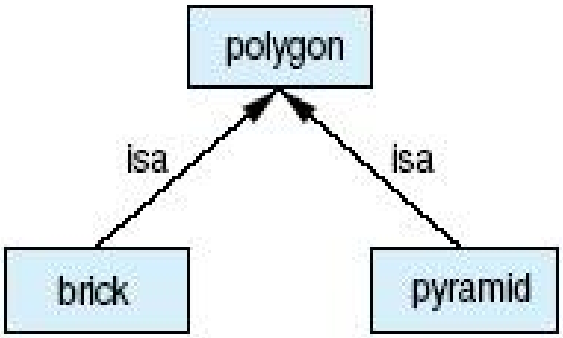


Fig 10.3 generalization of descriptions to include multiple examples *(cont'd)*

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



d. Generalization that includes both examples

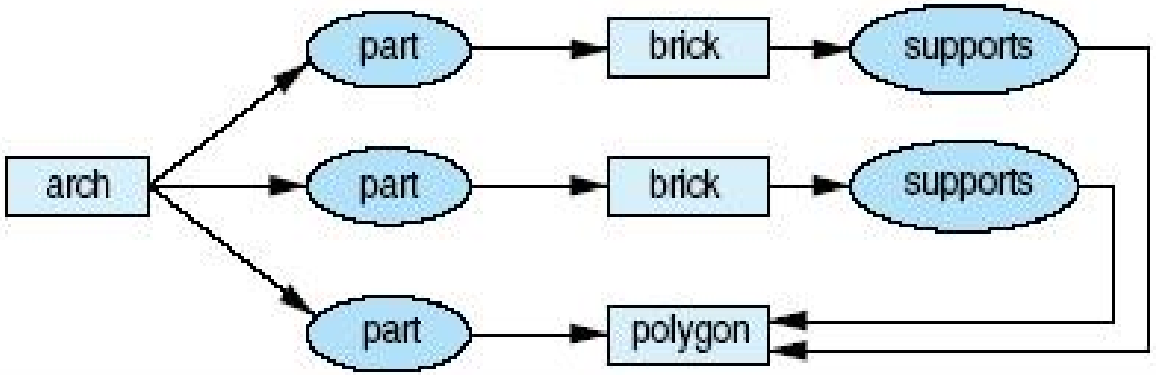
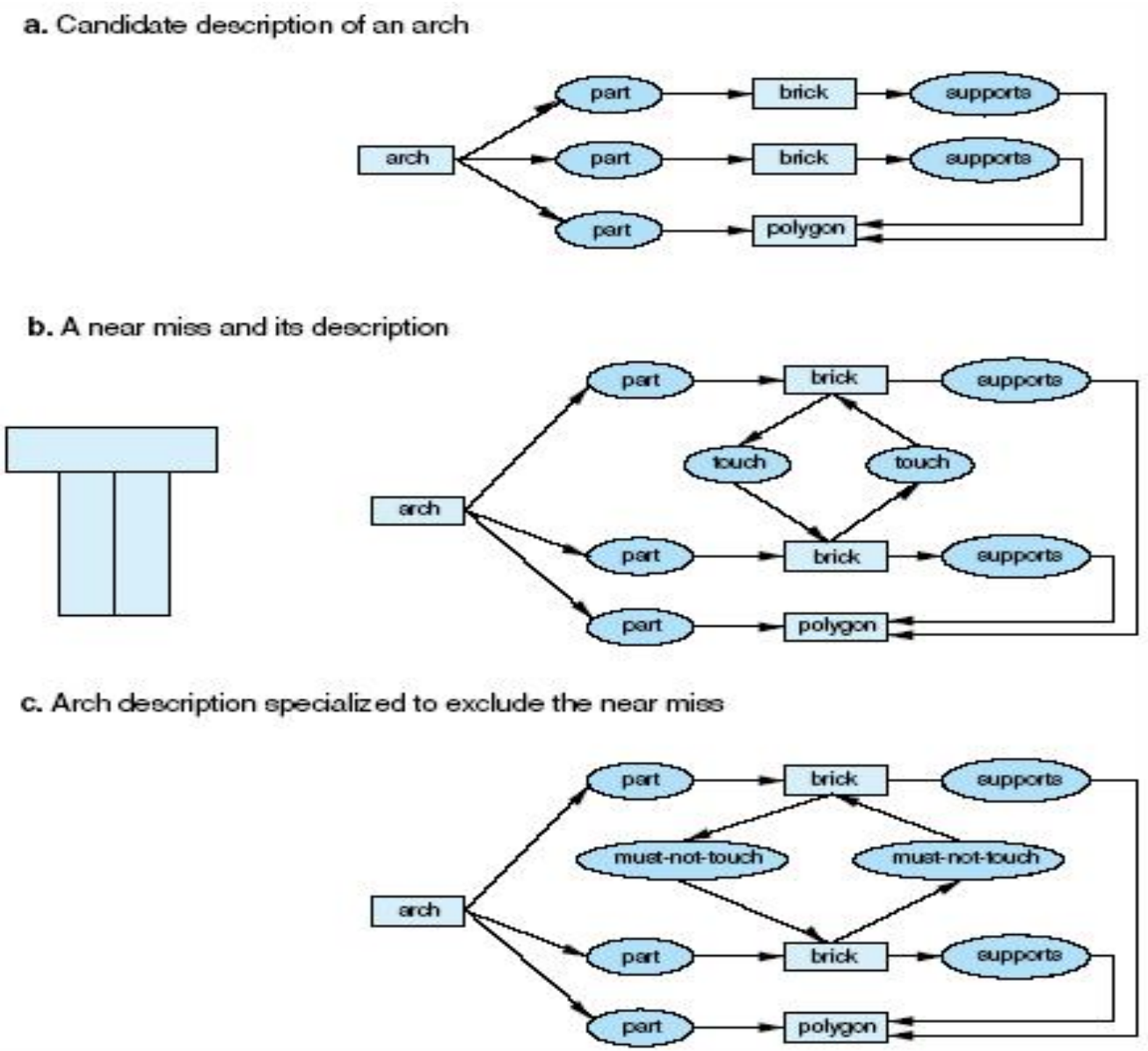


Fig 10.4 Specialization of a description to exclude a near miss. In 10.4c we add constraints to 10.4a so that it can't match with 10.4b.



10.2 Version Space Search

Version space search illustrates the implementation of inductive learning as search through a concept space.Version space search takes advantage of the fact that generalization operations impose an ordering on the concepts in a space,and then uses this ordering to guide the search.

10.2.1 Generalization Operators and the Concept Space

Generalization and specialization are the most common types of operations for defining a concept space.The primary generalization operations used in machine learning are:

1.Replacing constants with variables.For example,

color(ball,red) generalizes to color(X,red)

2.Dropping conditions from a conjunctive expression.

shape(X,round) $\wedge$size(X,small) $\wedge$color(X,red)

generalizes to shape(X,round) $\wedge$ color(X,red)

3.Adding a disjunct to an expression.

shape(X,round) $\wedge$ size(X,small) $\wedge$ color(X,red)

generalizes to

shape(X,round) $\wedge$ size(X,small) $\wedge$ color(X,red)
$\vee$ color(X,blue)

4.Replacing a property with its parent in a class hierarchy.

color(X, red) **generalizes to** color(X, primary_color)

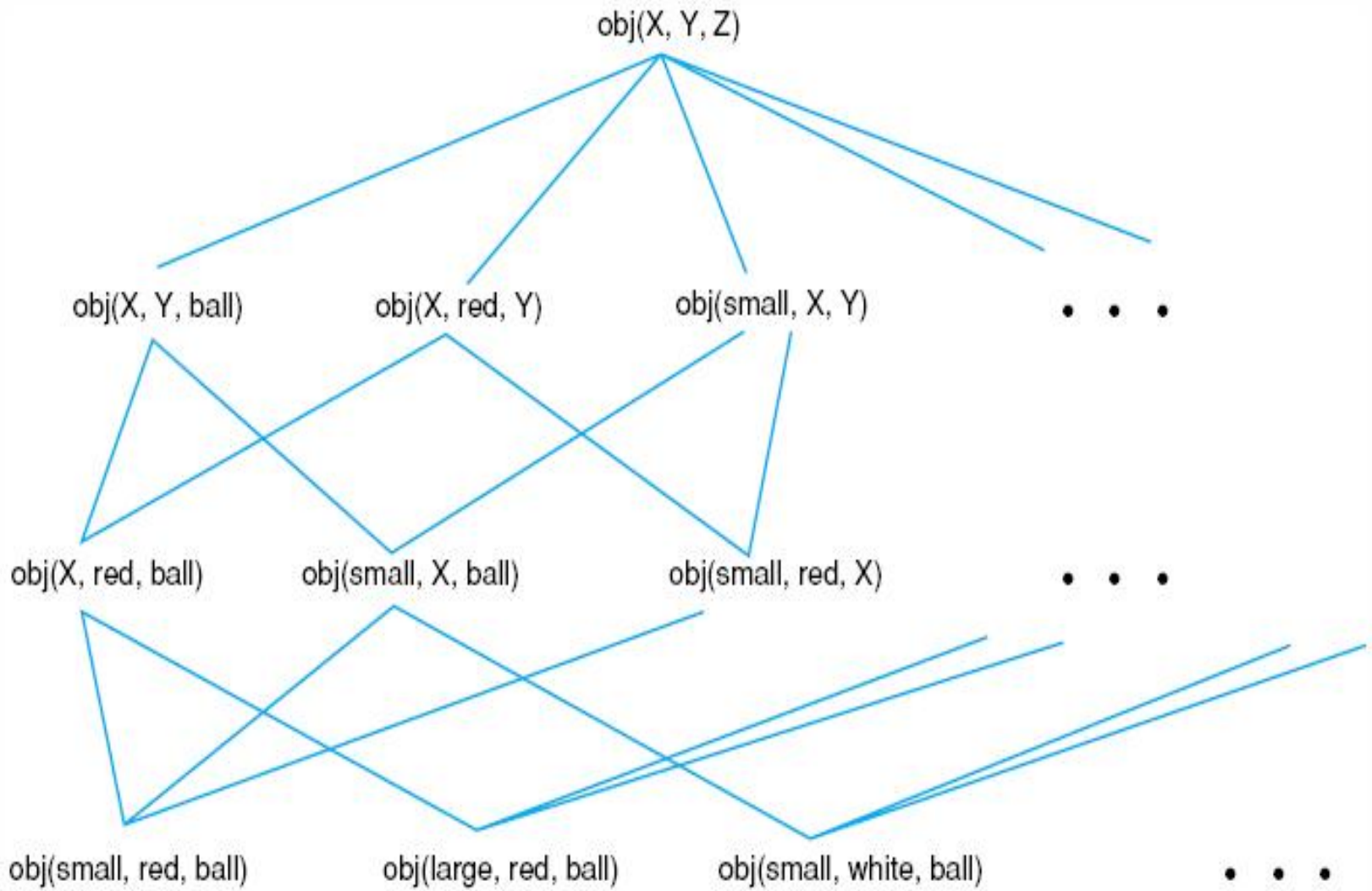
If concept pis more general than concept q,we say that p covers q(p⊇q).p>=q means that p is **more general** than q.

we formalize this relationship through the notion of covering.For an object X, p(x)->positive(x)and q(x)->positive(x).p covers q if q(x)->positive(x) is a logical consequence of p(x)->positive(x).

color(X,Y)covers **color(ball,Z)** covers **olor(ball,red)**

size={large,small},**color**={red,white,blue},**shape**={ball,brick,cube},these objects can be represented using the prediacate **obj**(sizes,color,shapes).

Fig 10.5 A concept space.



10.2.2 The Candidate Elimination Algorithm

This section presents **three algorithms** for searching the concept space.The first two algorithms reduce the version space in a general to specific direction, respectively. The third algorithm,called candidate elimination,combines these approaches into a bi-directional search.

Defining ***specific to general*** *search,* for hypothesis set S as:

```
Begin
Initialize S to the first positive training instance;
N is the set of all negative instances seen so far;

For each positive instance p
      Begin
      For every s ∈ S, if s does not match p, replace s with its most specific
            generalization that matchs p;
      Delete from S all hypotheses more general than some other hypothesis in S;
      Delete from S all hypotheses that match a previously observed negative
            instance in N;
      End;
For every negative instance n
      Begin
      Delete all members of S that match n;
      Add n to N to check future hypotheses for overgeneralization;
      End;
End
```

In this algorithm, negative instances lead to the **specialization** of candidate concepts; the algorithm uses positive instances to eliminate overly specialized concepts.

```
Begin
Initialize G to contain the most general concept in the space;
P contains all positive examples seen so far;

For each negative instance n
      Begin
      For each g ∈ G that matches n, replace g with its most general specializations
            that do not match n;
      Delete from G all hypotheses more specific than some other hypothesis in G;
      Delete from G all hypotheses that fail to match some positive example in P;
      End;

For each positive instance p
      Begin
      Delete from G all hypotheses that fail to match p;
      Add p to P;
      End;
End
```

Fig 10.6 The role of negative examples in preventing overgeneralization.

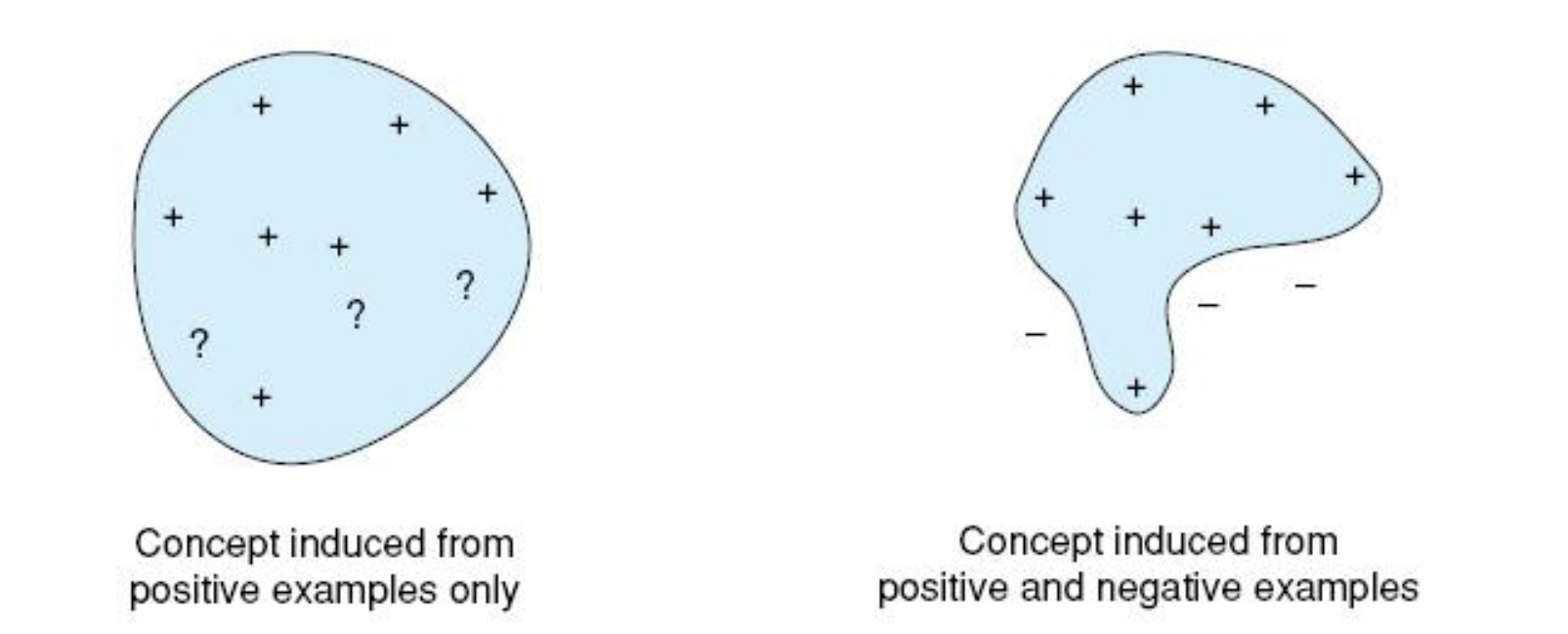
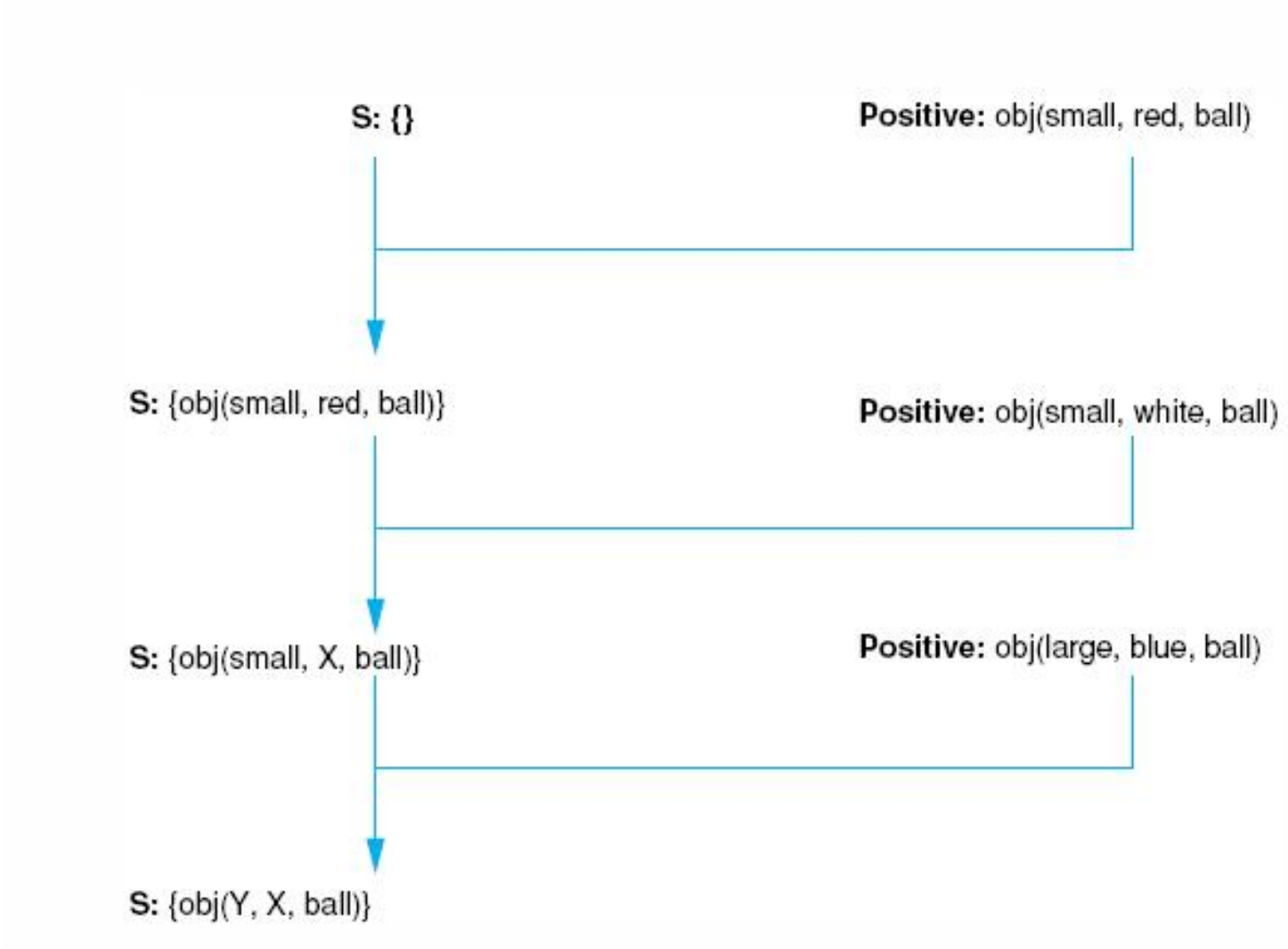


Fig 10.7 Specific to gerneral search of the version space learning the concept "ball."



The algorithm specializes **G** and generalizes **S** until they converge on the target concept. The algorithm is defined:

Begin
Initialize G to be the most general concept in the space;
Initialize S to the first positive training instance;

For each new positive instance p
Begin
Delete all members of G that fail to match p;
For every s ∈ S, if s does not match p, replace s with its most specific generalizations that match p;
Delete from S any hypothesis more general than some other hypothesis in S;
Delete from S any hypothesis more general than some hypothesis in G;
End;

For each new negative instance n
Begin
Delete all members of S that match n;
For each g ∈ G that matches n, replace g with its most general specializations that do not match n;
Delete from G any hypothesis more specific than some other hypothesis in G;
Delete from G any hypothesis more specific than some hypothesis in S;
End;

Fig 10.8 General to specific search of the version space learning the concept "ball."

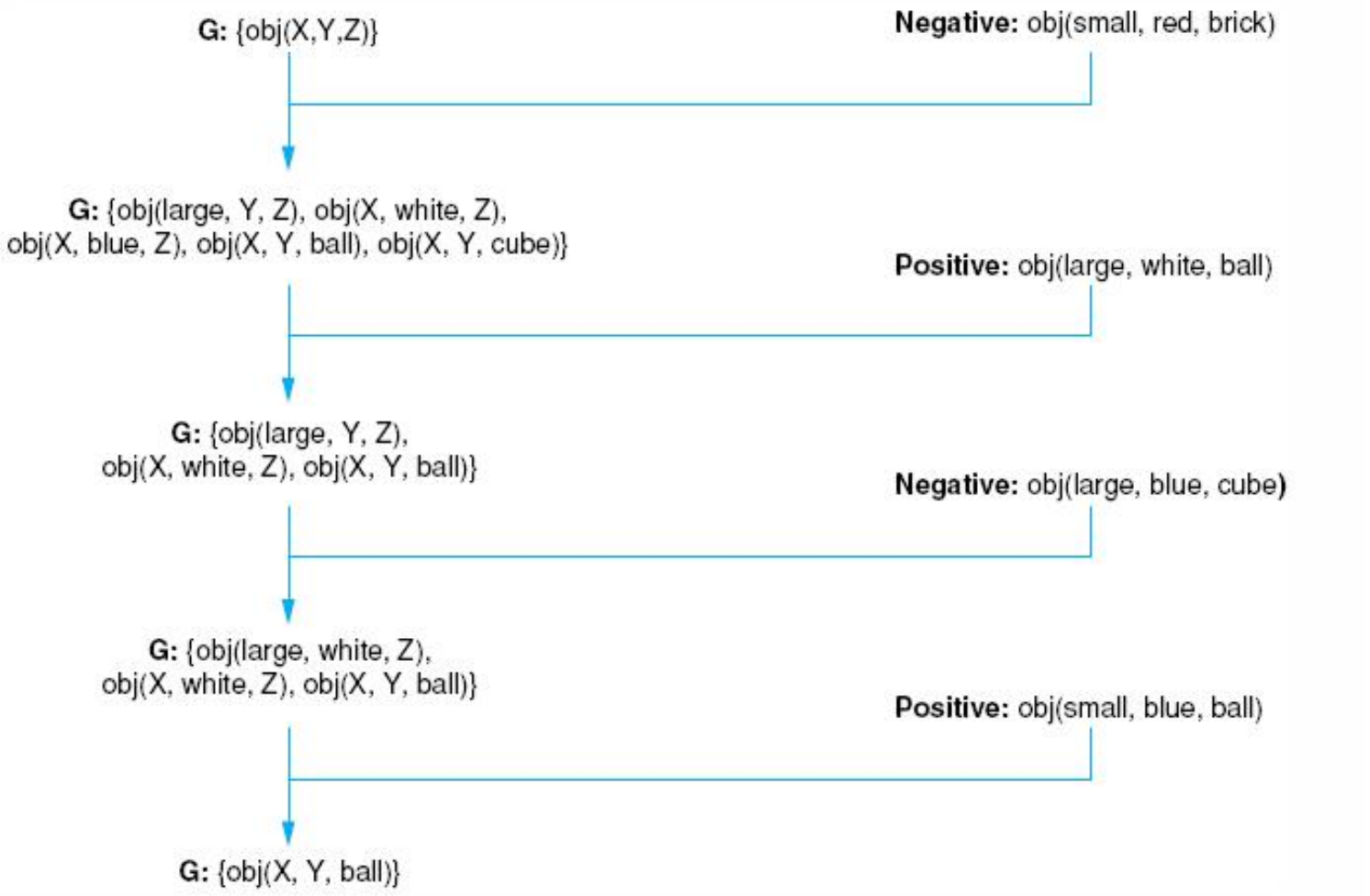


Fig 10.9 The candidate elimination algorithm learning the concept “red ball.”

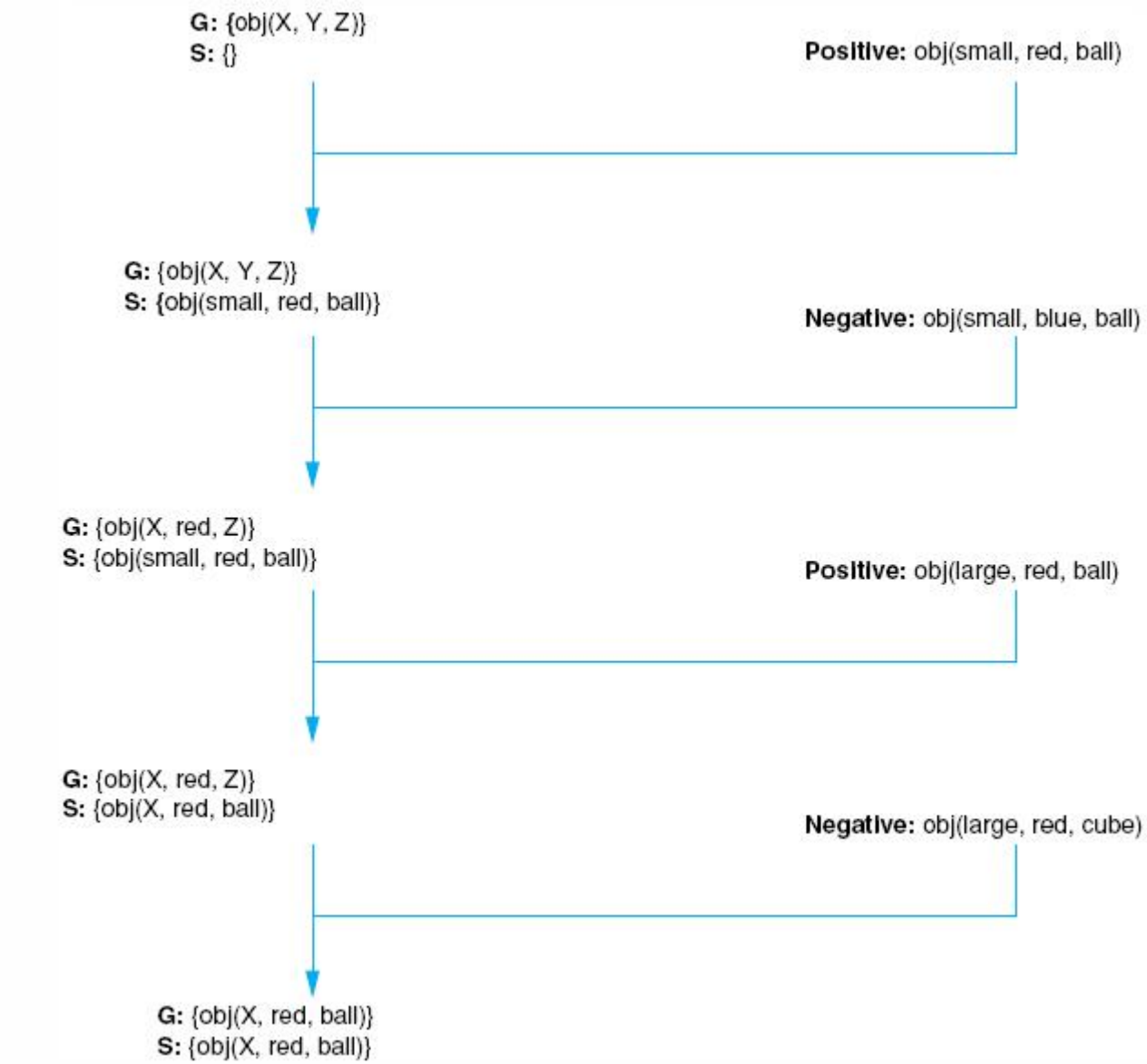
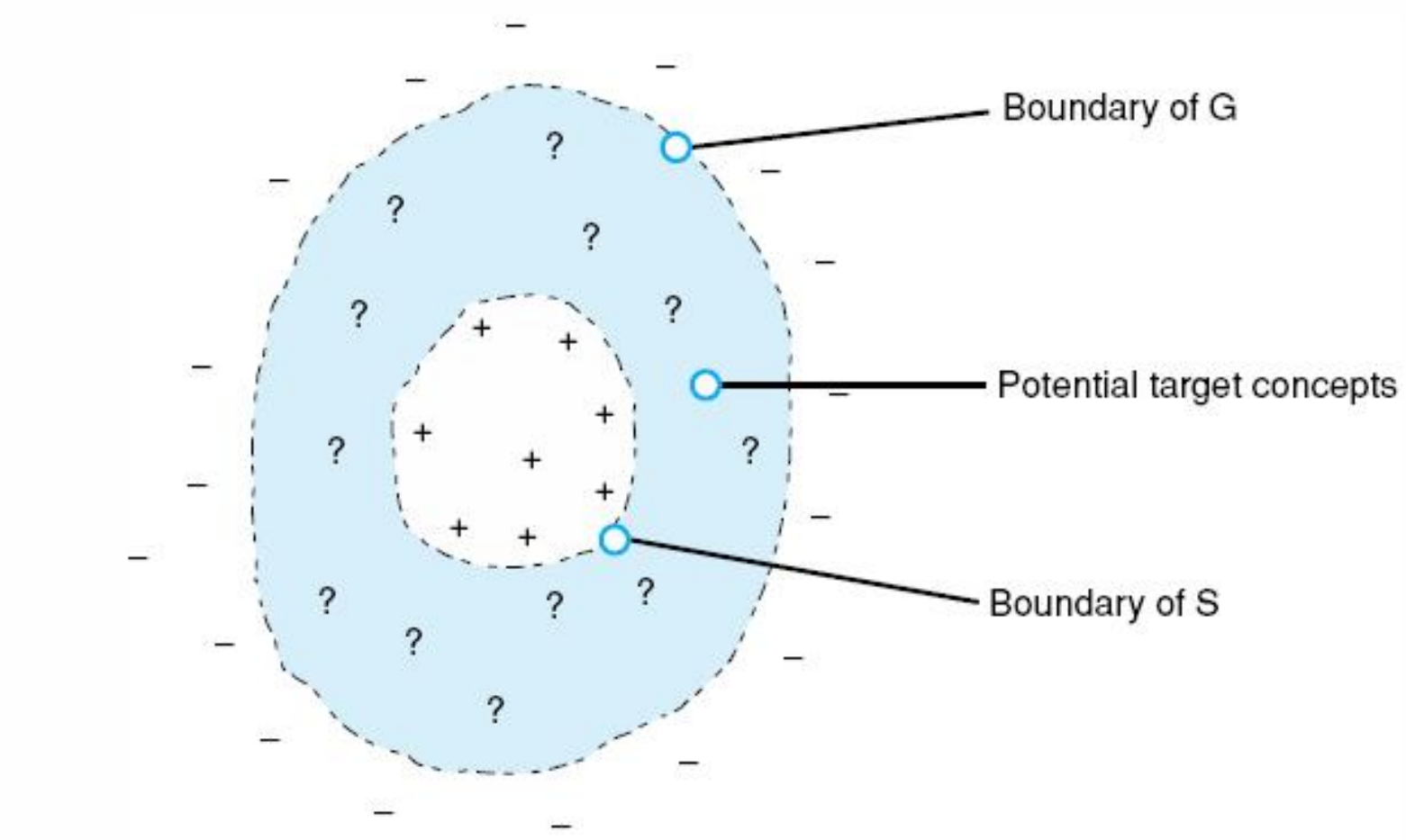


Fig 10.10 Converging boundaries of the G and S sets in the candidate elimination algorithm.



基于描述空间的归纳学习方法

- 在Mitchell提出的描述空间方法中，根据概念的一般性和特殊性把概念空间表示成偏序集的形式。
- 偏序集的最大元素G称为零描述。
- 零描述是概念空间中恒真的概念描述，是概念中最一般的概念的集合。
- 偏序集的最小元素S是所有训练实例，是概念空间最特殊的概念集合。

例 1

设实例空间是一组实体，每个实体用“大小”和“形状”两个特征描述，若“大小”属性值有：small、large，“形状”有属性值：square、circle。

对这个实例空间，请给出其概念空间的偏序集的最大元素和最小元素。

解：每个实体有“大小”和“形状”两个属性，可用(x,y)表示实体概念空间的最一般概念，x描述实体的“大小”，y描述实体的“形状”，(x,y)是该实体概念空间中零描述。

偏序集的最大元素G={(x,y)}。

偏序集的最小元素S是所有的训练实例，

故有：

S={(small square),(small circle),(large,square),(large,small)}

- 对于给定的训练实例，利用偏序集可以非常紧凑地表示与训练实例相一致的合理假设集合（一组概念描述）H，H由两个子集G和S所限定，H是由G和S以及G与S之间的元素构成的偏序集。
- 即 H=G∪S∪{K|S<K<G} 其中，“<”表示概念空间的偏序关系。
- H中还有一组与训练实例相一致的概念描述：

  K={(small y),(large y),(x square),(x circle)},

  G、K、S的偏序关系为S<K<G。

概念空间H= G∪K∪S的偏序集表示如图4。

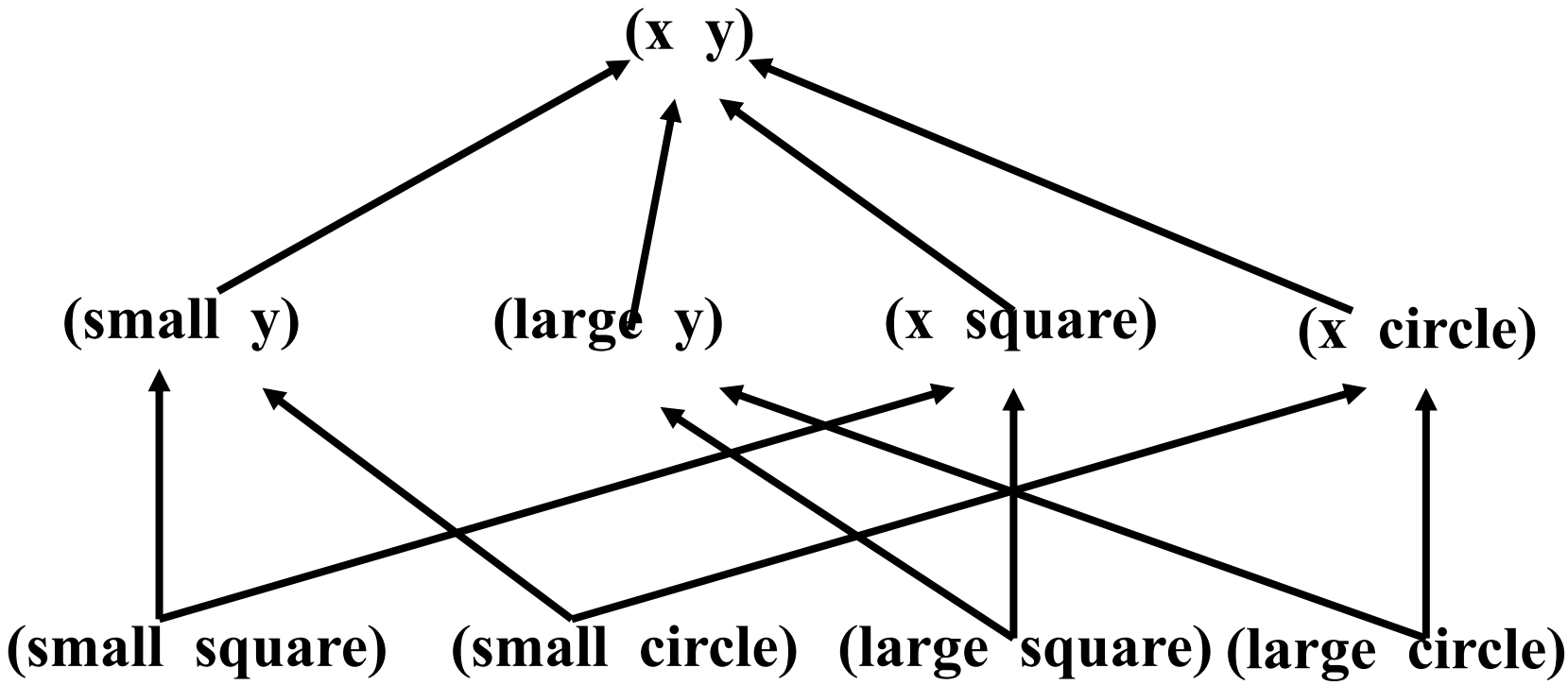


图4 概念空间的偏序集表示

消除候选者算法

(1) 初始化概念空间H=<G,S>，其中，G为零描述集合，S为训练集合，初始化时，S中只含第一个正例。

(2) 接受一个新的训练实例。如果实例为正例，则从G中删去所有与该例不相容的概念，并更新集合S，尽可能小地对S进行一般化，以相容这个新的正实例；如果实例为反例，则首先从S中删去所有与该例相容的概念，并更新集合G，尽可能小地特殊G中的元素，以便它们不相容这个反例。

(3) 如果G ≠S，则重复步骤(2)；否则输出H。

[例2]对例1给出的概念空间，应用消除候选者学习算法，说明学习得到一般概念为“圆”的学习步骤和学习结果。

注意以下术语：

- 概念空间
- 最大元素、最小元素
- 训练实例、正例、反例
- 相容的概念、不相容的概念
- 概念的一般化、特殊化

解：由例1给出的概念空间可得知，“圆”的一般概念显然表示为(x circle)。这一概念可采用消除候选者算法获得，过程如下：

1）初始化概念空间H=<G,S>,其中

G={(x y)}

S={(small square),(small circle),

(larges,square),(large,circle)}

2）若首先提供一个正例(small circle)，由消除候选者算法步骤（2）对H=<G,S>进行更新；G={(x y)}中的概念(x y)是最一般概念，与提供的正例(small circle)相容，故仍有G={(x y)}。删除S中与提供正例不相容的元素(large square)，对S中的另外2个元素尽可能小地一般化，即使这2个元素分别为(small y)和(x circle)，从而使它们与提供的正例(small circle)相容，故有

S={(small y),(small circle),(x circle)}

3) 此时有G ≠ S，若再提供一个反例(large square)，继续对H更新；由于S中的元素与提供的反例不相容，故S不变。对G中的元素(x y)尽可能小地特殊化，可由(x y)得到与提供反例不相容的2个元素(small y)和 (x circle)，故有G={(small y),(x circle)}

4) 此时有G ≠ S，若再提供一个正例(large circle)，继续对H更新：删除G中与提供的正例(large circle)不相容元素(small y)，故有G={(x circle)},删除S中与提供的正例(large circle)不相容元素(small y) 和(small circle)，故有S={(x circle)}。

5) 此时有G=S={(x cirlce)}，故算法终止，由概念空间H=<G,S>得出经过学习后获得的一般概念(x circle)。

基于变形空间的归纳学习1

设实例空间是一组实体，每一个实例用“大小”和“形状”两个特征描述。“大小”的属性值有：small、large，“形状”的属性值有：square、circle、triangle。针对这个实例空间，请回答下面问题：

(1)给出其概念空间偏序集的最大元素和最小元素。

(2)(x square) 和 (large y) 分别表示什么概念？

(3)给出 (x square) 的一个正例和一个反例。

(4)对(x y) 尽可能小地特殊化后，可以得到哪些概念？

(5)给出所有与 (x square) 不相容的实例。

基于变形空间的归纳学习2

- 考虑积木世界的以下属性和属性值:

  颜色={黄色，蓝色，绿色}

  形状={圆锥形，球形，长方形}

  硬度={硬，软}

  尺寸={大，小}

- 通过以下实例学习概念:黄色。

  实例集为:{(黄色，圆锥形，软，大，+)，

  (蓝色，长方形，软，小，-)，

  (黄色，球形，软，小，+)，

  (黄色，圆锥形，硬，大，+)，

  (黄色，长方形，软，大，+)，

  (绿色，球形，硬，大，-)，

  (蓝色，圆锥形，软，大，-)}

10.3 The ID3 Decision Tree Induction Algorithm

ID3, like candidate elimination， **induces concepts** from examples.

ID3 represents concepts as **decision trees**, a representation that allows us to determine the **classification** of an object by testing its values for certain **properties**.

10.3.1 Top-Down Decision Tree Induction

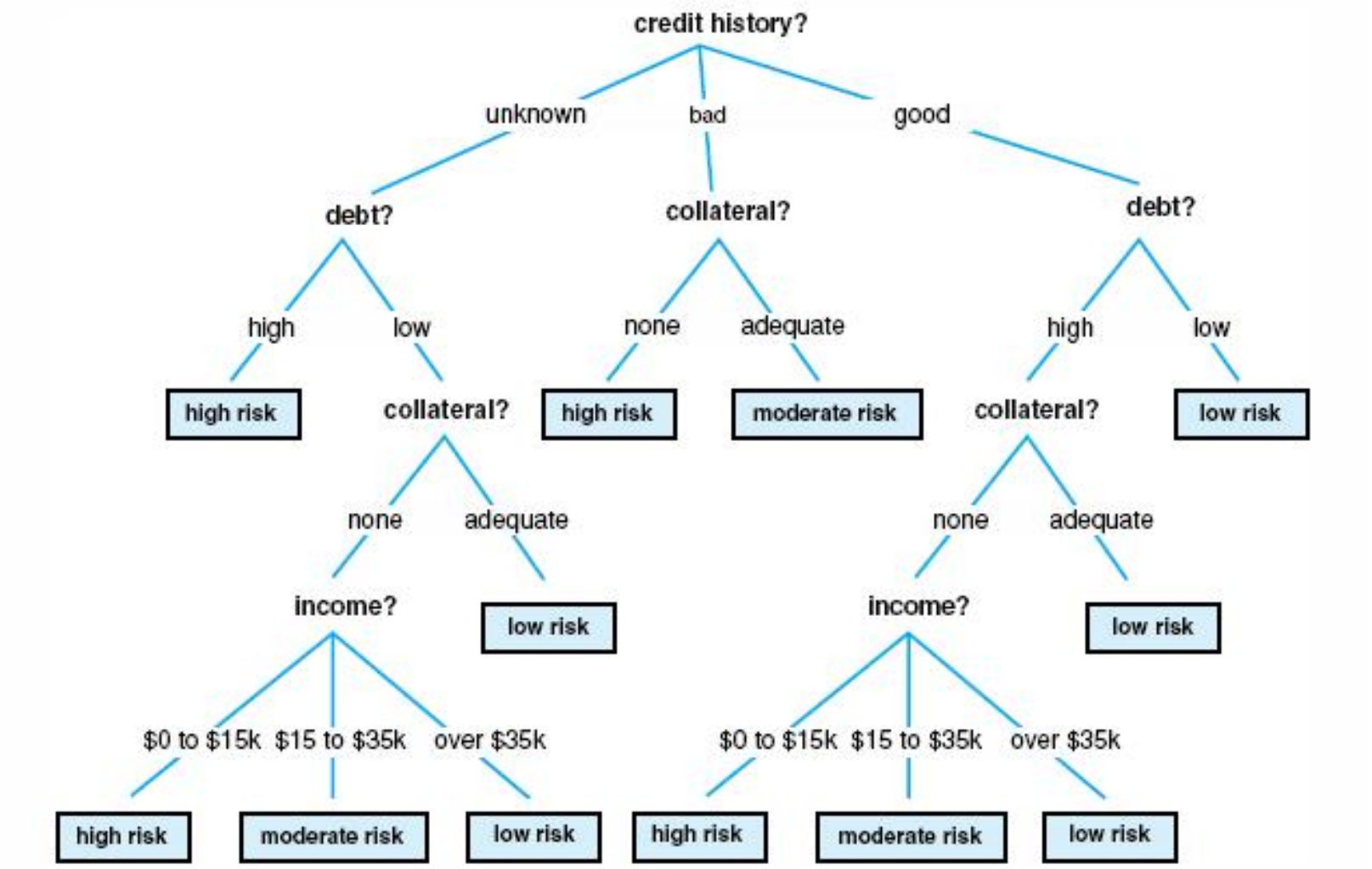
Table 10.1 Data from credit history of loan applications

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

Table 10.1 Data from credit history of loan applications applications

NO.	风险	信用历史	债务	抵押	收入
1	高	坏	高	无	0-15000美元
2	高	未知	高	无	15000-35000美元
3	中等	未知	低	无	15000-35000美元
4	高	未知	低	无	0-15000美元
5	低	未知	低	无	超过35000美元
6	低	未知	低	无	超过35000美元
7	高	坏	坏	无	0-15000美元
8	**中等**	**坏**	**坏**	**充分**	**超过35000美元**
9	低	好	好	无	超过35000美元
10	低	好	好	充分	超过35000美元
11	高	好	好	无	0-15000美元
12	中等	好	好	无	15000-35000美元
13	低	好	好	无	超过35000美元
14	高	坏	坏	无	15000-35000美元

Fig 10.13 A decision tree for credit risk assessment.



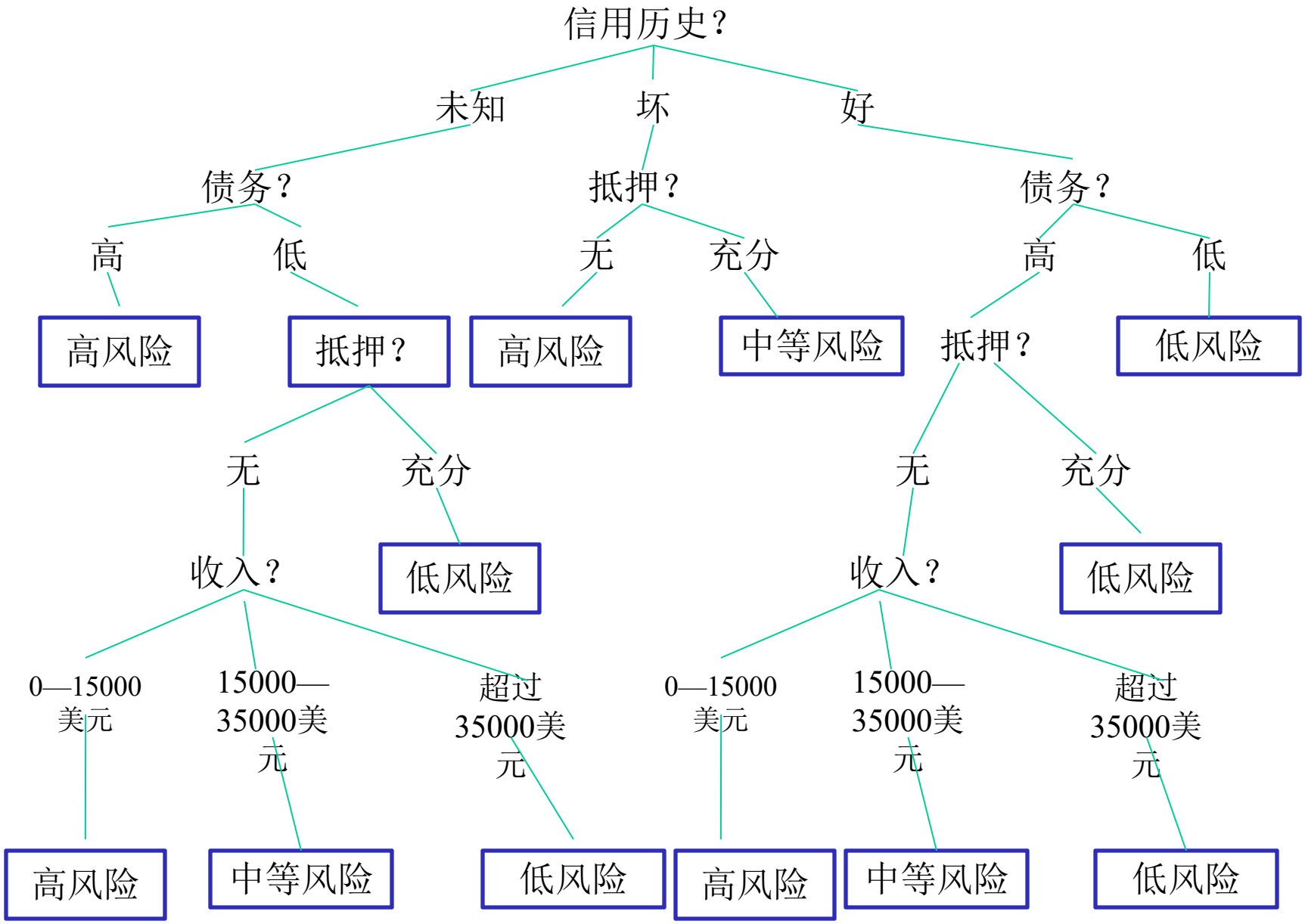
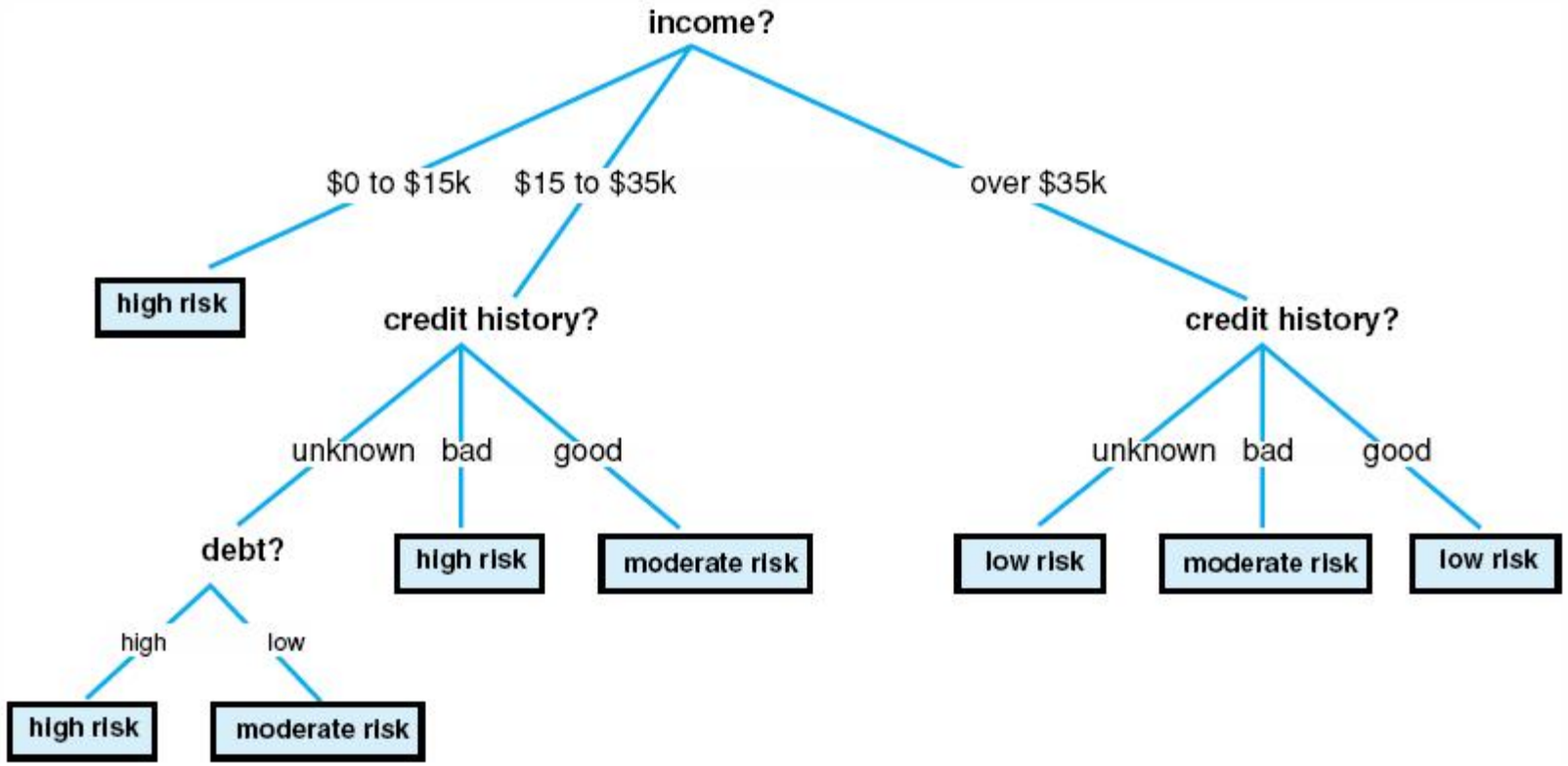


图10-13 信用风险评定的决策树

Fig 10.14 a simplified decision tree for credit risk assessment.



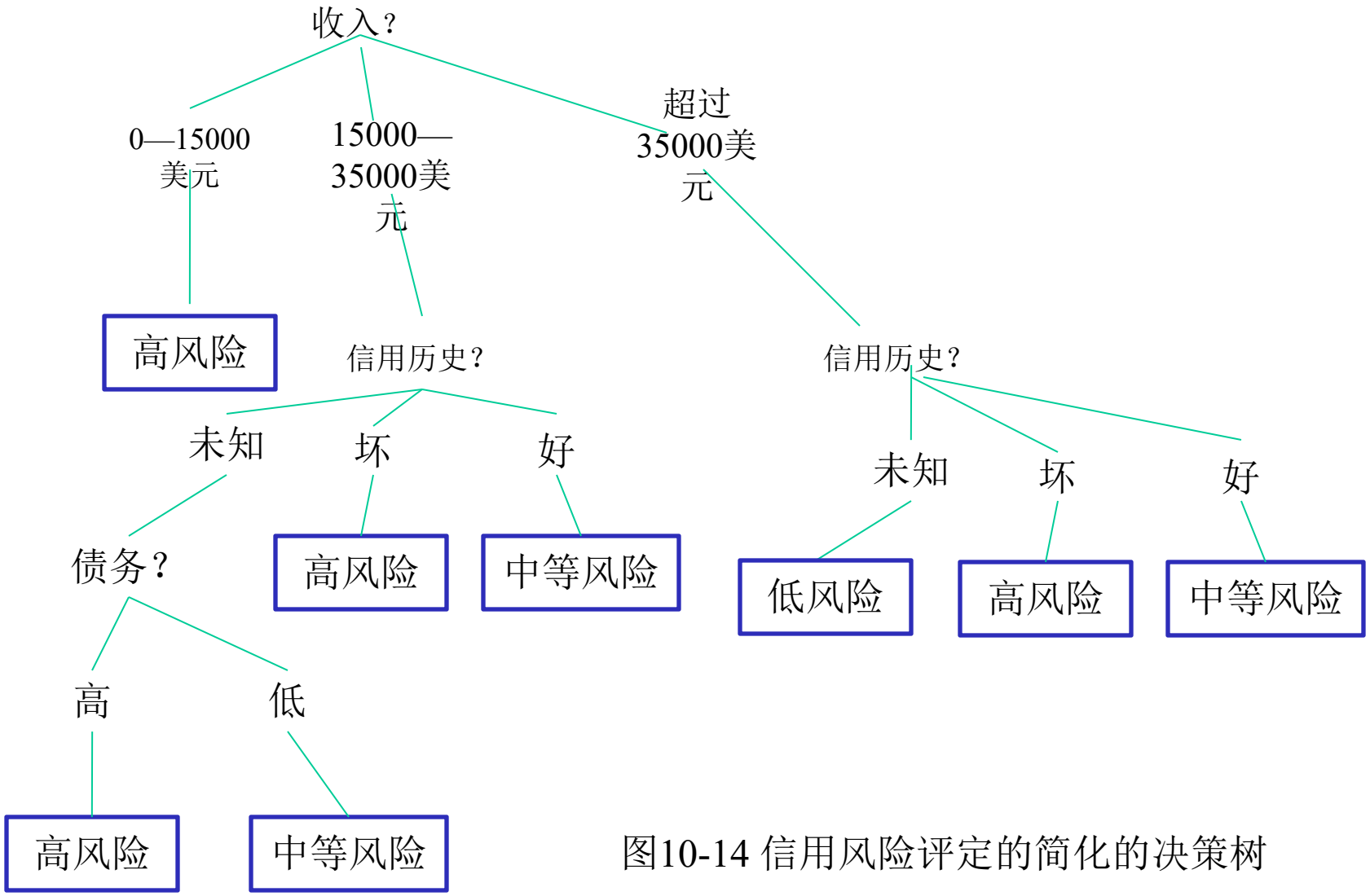
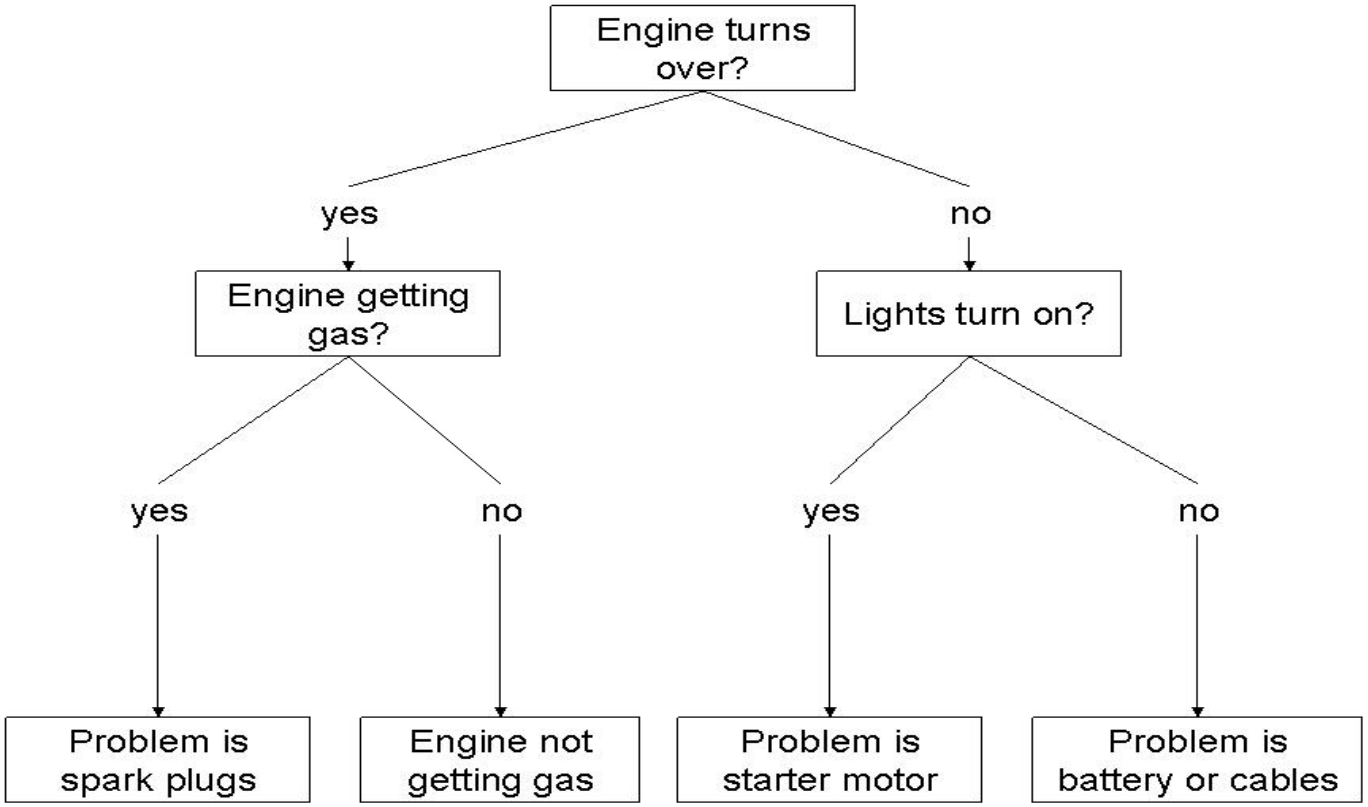


图10-14 信用风险评定的简化的决策树

Construct a decision tree such as is shown in Figures 10.13 and 10.14 for rules 1, 2, and 3 from the automobile diagnosis example from Chapter 8 Section 8.2.1.



The ID3 algorithm assumes that this is the simplest decision tree that covers all the traning examples.

This principle ,known as Occam's Razor:

It is vain to do with more what can be done with less..Entities should not be multiplied beyond necessity.

The induction algorithm begins with a sample of correctly classified members of the target categories. ID3 constructs a decision tree according to the 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
```

基于决策树的归纳学习方法

- Hunt提出的概念学习系统CLS（Concept Learning System)是一种基于决策树的归纳学习系统。1979年，J.R.Quinlan在CLS基础上进行了发展，提出了ID3算法，该算法不仅能方便地表示概念的“属性-值”信息结构，而且能从大量实例中有效地生成相应的决策树模型。

1.决策树及决策树构造算法CLS

在CLS的决策树中，节点对应于待分类对象的属性，由某一节点引出的弧对应于这个属性的可能值，叶节点对应于分类的结果。

[例如]: 假设天气预测分类系统确定用于描述天气的分类有3种属性：outlook、humidity和windy，这3种属性的值域分别为：

ValueType(outlook)={sunny,overcast,rain}

ValueType(humidity)={high,normal}

ValueType(windy)={true,false}

若对天气实例只需给出2种分类结果N和P，并认为P是正实例的分类结果，N是反实例的分类结果，则天气预测的CLS决策树如图5。

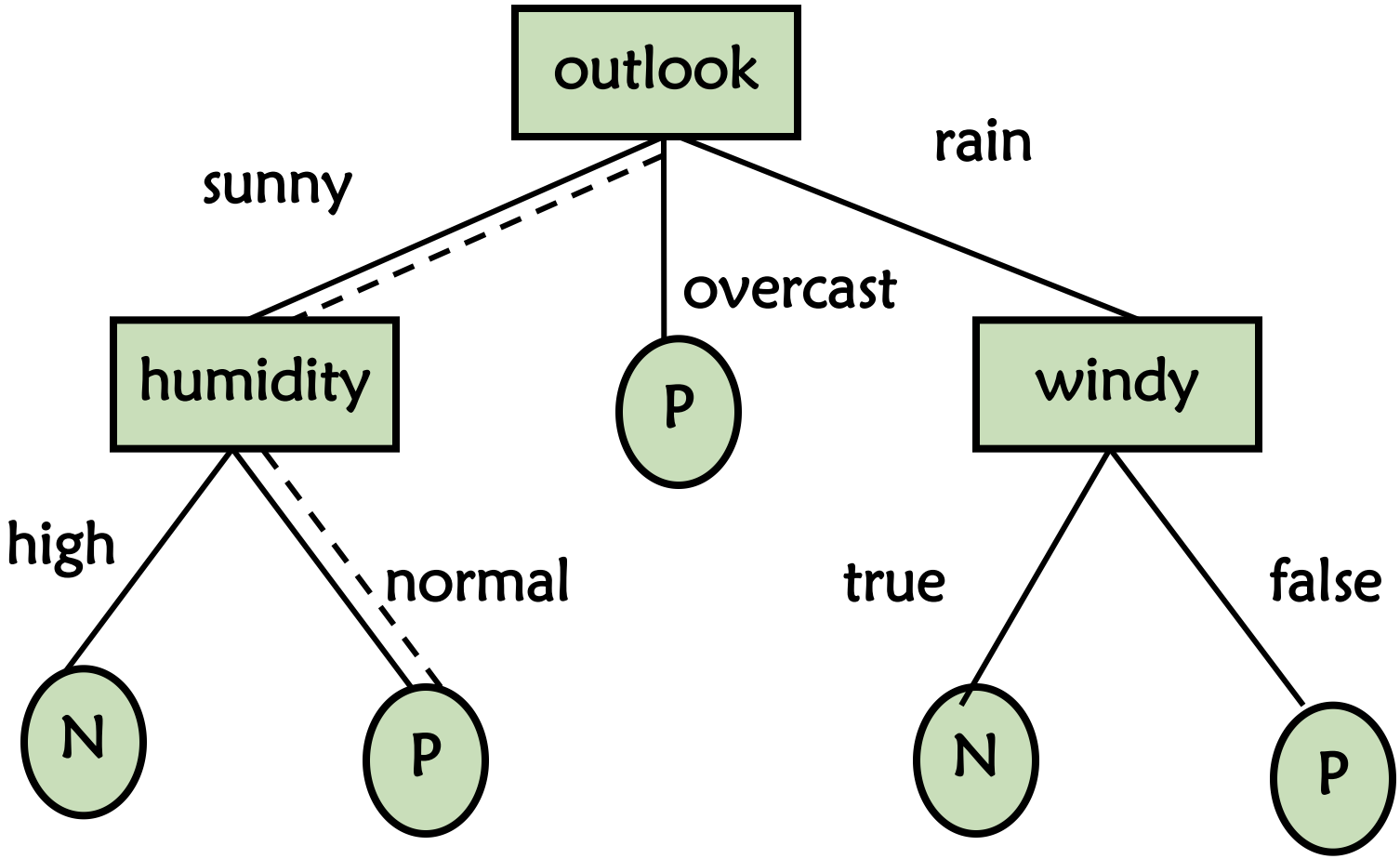


图5　天气预测的CLS决策树

术语：

- 属性：用节点表示。
- 属性值：用弧表示。
- 分类的结果：用叶子节点表示。
- 决策树：根据大量已知实例构造出的一般模型。

由此决策树可对任何的天气实例进行分类

- 设分类对象的属性表为 AttrList={A_1, A_2,…, A_n}，每个属性A_i的值域记为 Value Type(A_i)，值域可是离散的，也可是连续的。

- 把分类结果组成分类结果表Class={C_1, C_2,…, C_m}，一般应有n>=1,m>=2 。训练实例集T={<X,C>}中的一个元素就是一个训练实例<X,C>，实例<X,C>的特征向量X=(a_1, a_2,…, a_n)，其中为a_i这个实例的第i个属性的取值，C为这个实例的分类结果，C∈Class。可递归地描述下述决策树构造算法CLS。

决策树构造算法CLS

(1)如果训练实例集T中所有实例的分类结果均为C，则返回C。

(2)从属性表AttList中任选一个属性作为检测属性。

(3)若属性A_i的值域ValueType(A_i)中有s个不同的取值，则将T分成s个子集$T_1, T_2, \ldots, T_s$，每个子集的所有元素的属性A_i的取值相同。

(4)更新属性表，从属性表Attrlist中删除检测属性A_i。

(5)对每个子集T_k，若子集T_k中的所有实例的分类结果均为C_j，则生成叶节点C_j；若子集中T_k所有实例的分类结果有两个或两个以上，则对子集T_k和更新后的属性表转到步骤2），递归调用CLS构造算法，生成T_k的子树。

[例3] 假设需要根据人员的外貌特征对人员进行分类，用于人员分类的外貌特征有3个，它们组成人员分类属性表为：

AttrList={height, hair, eyes}

个属性值域分别为：

ValueType(height)={short,tall}

ValueType(hair)={blond,red,dark}

ValueType(eyes)={blue,brown}

（接下页）

若对人员进行分类的结果有两种，分别用“+”和“-”表示，组成分类结果表为：

Class={+,-}

若提供学习的训练实例集为：

T={<(short,blond,blue),+>, <(tall,blond,brown) ,- >,
<(tall,red,blue),+>, <(short,dark,blue), ->,
<(tall,dark,blue),->, <(tall,blond,blue),+>,
<(tall,dark,brown),->,(short,blond,brown),->}

应用CLS构造算法构造决策树，说明构造过程，画出构造的决策树。

解：执行CLS算法构造决策树的构造过程如下：

1)若选取属性表AttrList={height,hair,eyes}中的属性height作为第一个检测属性，则将T分为2个子集T_1和T_2，其中，T_1是属性height取值为short的训练实例子集，T_2是属性height取值为tall的训练实例子集。故有：

T_1={<(short,blond,blue),+>,<(short,dark,blue),->
<(short,blond,brown),->}

T_2={<(tall,blond,brown),- >,<(tall,red,blue),+>,
<(tall,dark,blue),- >,<(tall,blond,blue),+>,
<(tall,dark,brown),->}

从属性表中删除检测属性height ,故新的属性表为：

AttrList={hair,eyes}

生产决策树如图6(a)所示

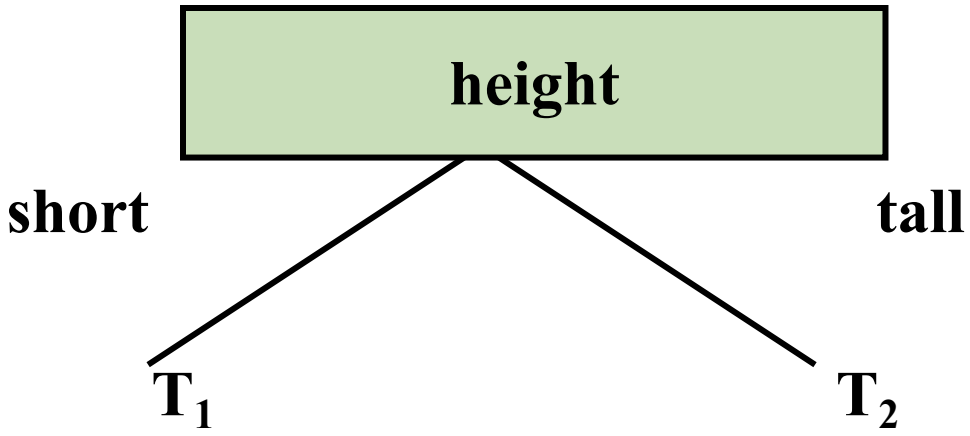


图6　生成的决策树(a)

2) 由于T_1和T_2中的实例的分类结果仍有两种，故需按新的属性表递归构造T_1和T_2的子树。若选取新属性hair作为第2个检测属性，且由于hair的值域ValueType(hair)={blond,red,dark}有3个取值，故将T_1分为3个子集T_{11}、T_{12}、T_{13}，将T_2也分为3个子集T_{21}、T_{22}和T_{23}，它们分别为：

T_{11}={<(short,blond,blue),+>,<short,blond,brown),->}

T_{12}={ }

T_{13}={<short,dark,blue>,->}

T_{21}={<tall,blond,brown),->,<(tall,blond,blue),->}

T_{22}={<(tall,red,blue),+>}

T_{23}={<(tall,dark,blue),->,<(tall,dark,brown),->}

从属性表中删除检测属性hair，故新属性表为：

AttrList={eyes}

至此，生产决策树如图6(b)所示。

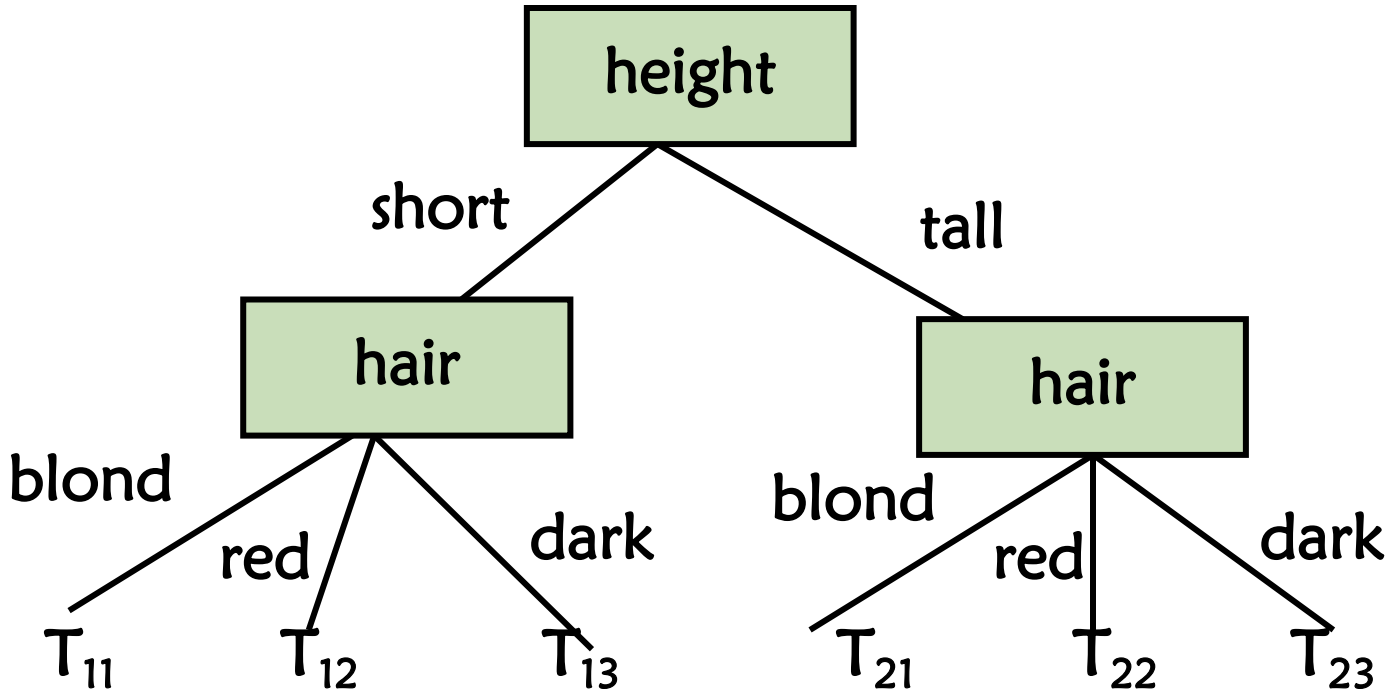

height
short
tall
hair
hair
blond
red
dark
blond
red
dark
T11
T12
T13
T21
T22
T23

图6　　生成的决策树(b)

3) 由于T_{13}、T_{22}和T_{23}中的实例的分类结果只有一种，故无需对T_{13}、T_{22}和T_{23}中的实例划分子集，并用T_{13}的实例分类结果“-”作为叶节点代替图6(b)所示决策树中节点的T_{13}，同样，用T_{22}的实例分类结果“+”作为叶节点代替T_{22}，用T_{23}的实例分类结果“-”为叶节点代替T_{23}。

T_{12}是一个空集，故从决策树中删除节点T_{12}。

由于T_{11}和T_{12}中的实例的分类结果仍有两种，故还需按新的属性表递归构造T_{11}和T_{21}的子树。

以新属性表中的属性eyes作为第3个检测属性，且eyes的值域ValueType(eyes)={blue,blown}有2个取值，故将T_{11}分为2个子集T_{111}和T_{112}，将T_{21}也分为2个子集T_{211}和T_{212}，它们分别为：

T_{111}={<(short,blond,blue),+>}

T_{112}={<(short,blond,brown),->}

T_{211}={<(tall,blond,blue),+>}

T_{212}={<(tall,blond,brown),->}

从属性表中删除检测属性eyes，故新属性表已成为一个空表。

至此生产决策树如图6(c)所示。

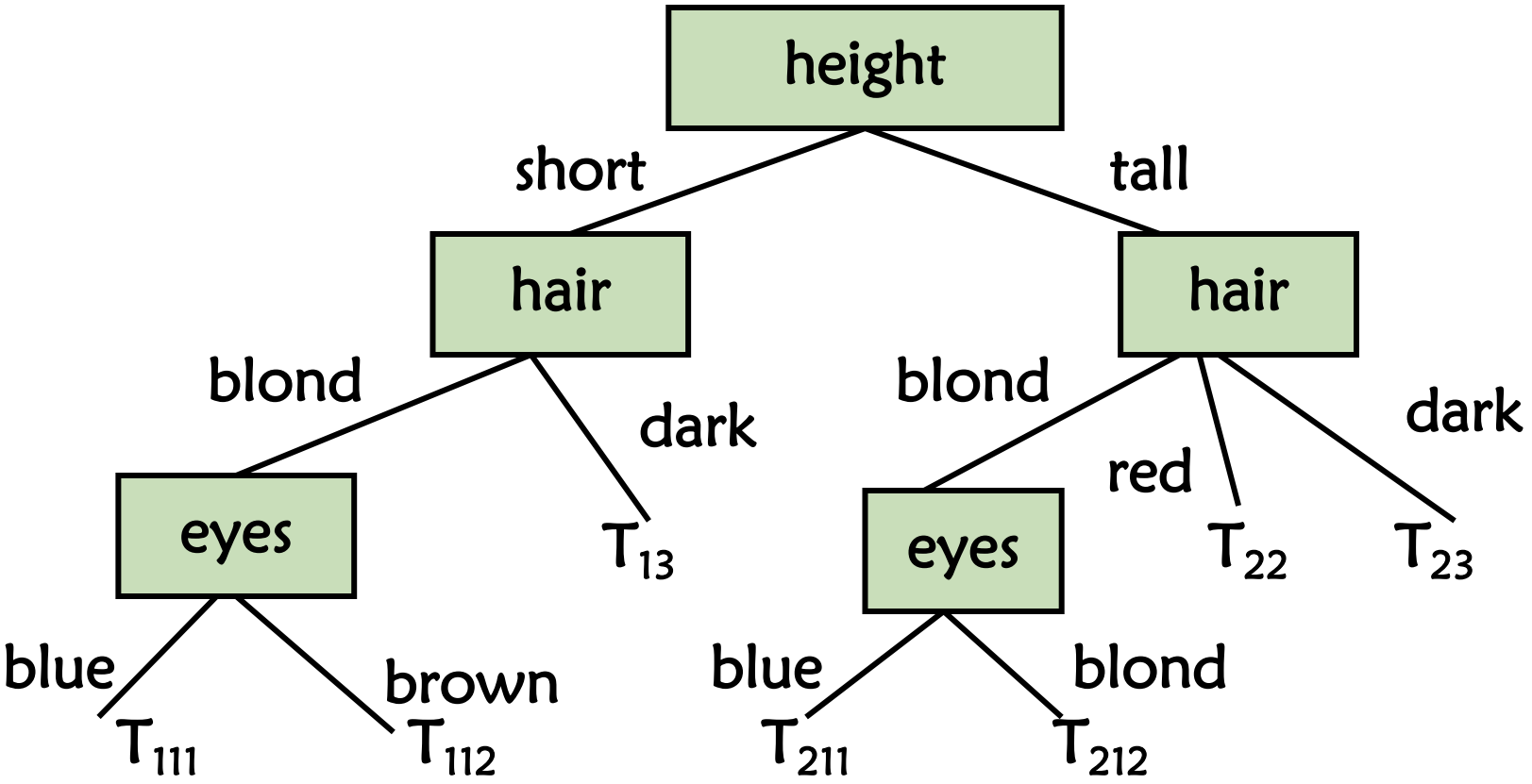

height
short
tall
hair
hair
blond
dark
blond
red
dark
eyes
eyes
T_{13}
T_{22}
T_{23}
blue
brown
blue
blond
T_{111}
T_{112}
T_{211}
T_{212}

图6　生成的决策树(c)

4) 由于T_{111}、T_{112}、T_{211}和T_{212}中的实例的分类结果只有一种，则用它们的实例分类结果作为叶节点来分别代替它们，得到最后完整的决策树如图6(d)所示。

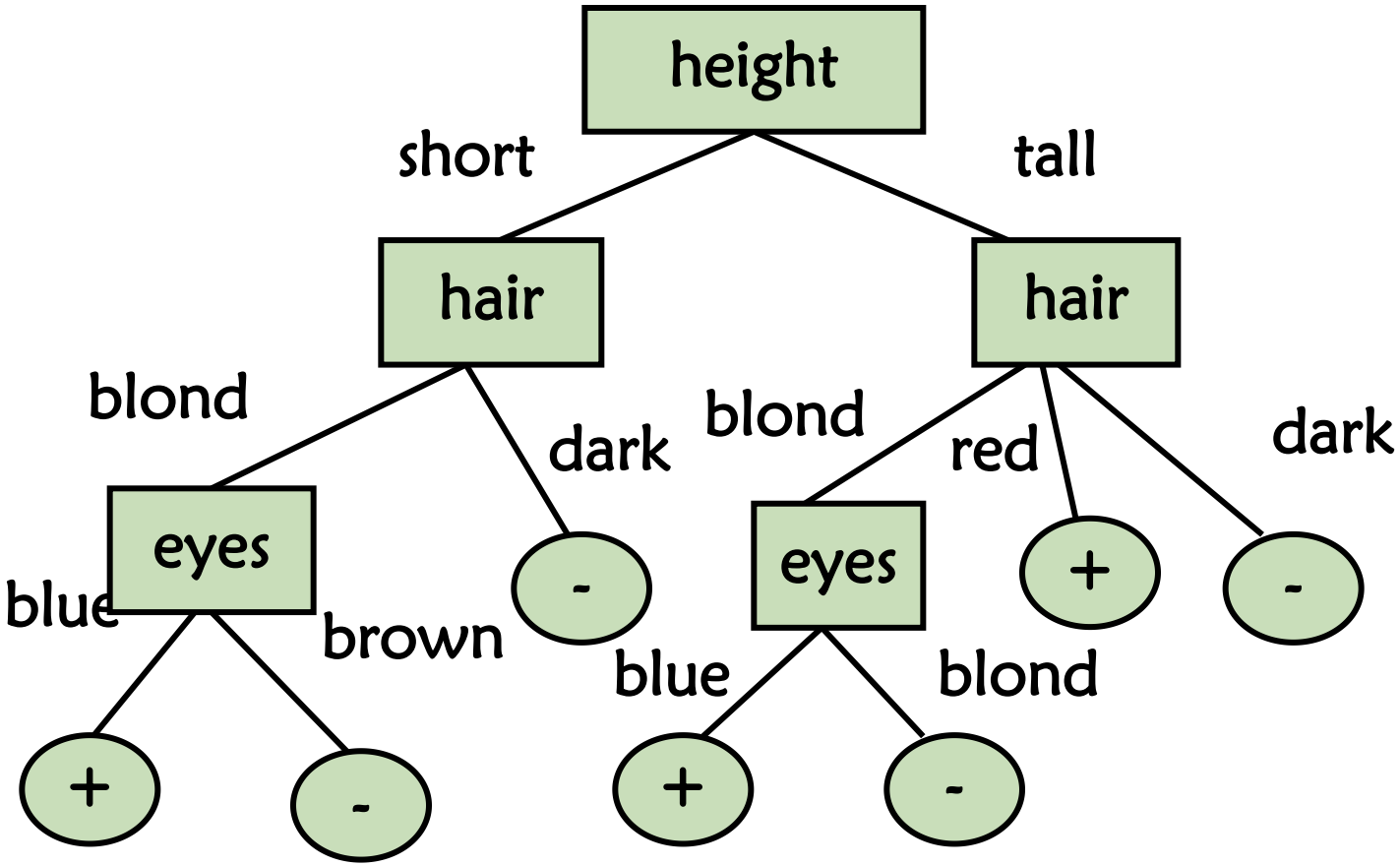

height
short
tall
hair
hair
blond
dark
blond
red
dark
eyes
eyes
blue
brown
blue
blond
-
+
-
+
-
+
-

图6　生成的决策树(d)

基于决策树的归纳学习

假设需要根据人员的外貌特征对人员进行分类，用于人员分类的外貌特征有3个，它们组成人员分类属性表为：AttrList={height, hair, eyes}

各属性的值域分别为：

ValueType(height)={short, tall},

ValueType(hair)={blond, red, dark},

ValueType(eyes)={blue, brown}。

对人员进行分类的结果有两种，分别用“+”和“—”表示，组成分类结果表为：

Class = { +, — }，提供学习的训练实例集为：

T = { < short, blond, blue, + >, < (tall, red, brown), — > ,

< (tall, red, blue), + >, < (short, dark, blue), + >,

< (tall, dark, blue), — >, < (tall, blond, blue), — > }

应用CLS算法构造决策树，画出中间结果图和最终构造的决策树。

10.3.2 Information Theoretic Test Selection

We may think of each **property** of an instance as contributing a certain **amount of information** to its **classification.**

Information theory provides a mathematical basis for **measuring** the information content of a message(消息）.

（1）The number of possible messages is important.

（2）The influence of the probability of each message on the amount of information is important.

Shannon, definition：$-\log_2 P$

Given:$T=\{m_1,m_2,...,m_m\}$,the expected information content of a message is given by:

$$I(T) = -\sum_{j=1}^{m} P_j \log_2 P_j = -\sum_{j=1}^{m} \frac{e_j}{|T|} \log_2 \frac{e_j}{|T|} = -(\sum_{j=1}^{m} e_j \log_2 \frac{e_j}{|T|}) / |T|$$

The information in a message is measured in **bits**. For example,the information content of a message telling the outcome of the flip of an honest coin is:

$$I[\text{Coin toss}]=-p(\text{heads})\log_2(p(\text{heads}))-p(\text{tails})\log_2(p(\text{tails}))$$

$$=-1/2\log_2(1/2)-1/2\log_2(1/2)$$

$$=1 \text{ bit}$$

- If the coin has been rigged to come up heads 75 per cent of the time ,then the information concent of a message is:

$$I[\text{Coin toss}]=-3/4\log_2(3/4)-1/4\log_2(1/4)$$

$$= -3/4*(-0.145)-1/4*(-2)$$

$$=0.811 \text{ bits}$$

Example label10-1:

P(risk is high)=6/14,p(risk is moderate)=3/14,p(risk is low)=5/14

It follows that the distribution described in Table10.1,D9.1

$I[D_{9.1}]$=-6/14$\log_2$(6/14)-3/14$\log_2$(3/14)-5/14$\log_2$(5/14)

=-6/14*(-1.222)-3/14*(-2.222)-5/14*(-1.485)

=1.531bits

$$E[p]=\sum_{i=1}^{n}\frac{|C_i|}{|C|}I[C_i]$$

gain(P)=I[C]-E[p]

Example:we make income the property tested at the root of the tree,this partitions the table of example into the partitions C1={1,4,7,11},C2={2,3,12,14},and C3={5,6,8,9,10,13}The expected information needed to complete the tree is:

E[income]=4/14*I[C1]+4/14*I[C2]+6/14*I[C3]

=4/14*0.0+4/14*1.0+6/14*0.650

=0.564 bits

The information gain for the distribution of Table 10.1 is :

gain(income)=|[D9.1]-E[income]

=1.531-0.564

=0.967 bits

gain(credit history)=0.266

gain(debt)=0.063

gain(collateral)=0.206

Because income provides the greatest information gain,ID3 will select it as the root of the tree.

ID3算法

- 由于CLS决策树构造算法每次从属性表AttrList中任选一个属性Ai作为检测属性来扩展生成决策树，按照训练集T中所有实例的属性Ai的取值来划分实例子集，因此，CLS算法的开销较大，效率较低。

- Quinlan提出了ID3算法，对检测属性的选择给出一种启发式规则，这个规则选择平均信息量（熵）最小的属性检测Ai，因此又称为最小熵原理。

若选择属性表AttrList={A_1,A_2,...,A_n}中的属性任选一个属性Ai为检测属性，则由A_i的值域ValueType(A_i)={V_1,...,V_s}的s个取值把训练实例集T分为s个子集：

$$T = \bigcup_{K=1}^{S} T_K^{(i)}$$

子集$T_K^{(i)}$中的所有实例的属性A_i的取值为V_k。

T中的实例的分类结果组成了分类结果表Class={C_1,C_2,...,C_j,...,C_m}，若分类结果为C_j的**实例数**为e_j，1<=j<=m,且

$$\sum_{j=1}^{m} e_j = |T|$$

|T|表示训练实例集T中的实例总数，则实例分类结果为C_j的概率为:$P_j=e_j/|T|$

定义训练实例集T的**平均信息量**为：

$$I(T) = -\sum_{j=1}^{m} P_j \log_2 P_j = -\sum_{j=1}^{m} \frac{e_j}{|T|} \log_2 \frac{e_j}{|T|} = -(\sum_{j=1}^{m} e_j \log_2 \frac{e_j}{|T|}) / |T|$$

类似可定义$T_K^{(i)}$的实例**平均信息量**为：

$$I\left(T_k^{(i)}\right) = -(\sum_{j=1}^{m} e_j^{(i)} \log_2 \frac{e_j^{(i)}}{|T_K^{(i)}|}) / |T_K^{(i)}|$$

其中，$e_j^{(i)}$为子集$T_K^{(i)}$中分类结果Cj的实例数，$T_K^{(i)}$为子集中实例总数，

$$\sum_{j=1}^{m} e_j^{(i)} = |T_k^{(i)}|$$

•若选择属性A_i作为检测属性将训练集T分为s个子集后，可由各实例子集的实例总信息量$|T_{K(i)}| \cdot I(T_{K(i)})$之和对实例集$|T|$的实例总数T的平均值来表示为实例集T的平均信息量。

即：

$$I(T, A_i) = \left(\sum_{K=1}^{s} |T_K^{(i)}| \cdot I(T_K^{(i)})\right) / |T|$$

选择属性A_i作为检测属性来将实例集T划分子集的启发式规则

选择属性作为检测属性，应由属性A_i的不同值把实例集划分为若干子集之前与之后的实例平均信息量的**差最大**，即使得$GI(T, A_i)=I(T)-I(T, A_i)$ 的值最大。$GI(T, A_i)$可认为是属性A_i对训练实例集T的信息量，因此，启发式规则实际是选择**信息量最小**的属性作为检测属性A_i来划分实例集。

例4：

应用ID3算法构造例3 要求构造的决策树，

说明构造过程决策树。

解：应用ID3算法构造决策树的构造过程如下：

1）实例集T分类结果为“+”的实例数为3，分类结果为“-”的实例数为5，故T的实例平均信息量为：

$I(T)=-(5\log_2 5/8+3\log_2 3/8)/8=0.954$ (bit)

若选属性表AttrList={height,hair,eyes}中的第1个属性height为检测属性，其值域ValueType(height)={short,tall}有2个值，则将T分为2个子集$T_1^{(1)}$和$T_2^{(1)}$：

$T_1^{(1)}$ ={<(short,blond,blue),+>,<short,dark,blue),->,
<short,blond,brown),->}

$T_2^{(1)}$ ={<(tall,blond,brown),->,<(tall,red,blue),+>,
<(tall,dark,blue),->,<(tall,blond,blue),+>,
<(tall,dark,brown),->}

$T_1^{(1)}$中的分类结果为“+”的实例数为1，分类结果为“-”的实例数为2，可得到$T_1^{(1)}$的实例平均信息量为：

$$I(T_1^{(1)})=-(\log_2 1/3+2\log_2 2/3)/3=0.918 \text{ (bit)}$$

$T_2^{(1)}$中的分类结果为“+”的实例数为2，分类结果为“-”的实例数为3，可得到$T_2^{(1)}$的实例平均信息量为

$$I(T_2^{(1)})=-(2\log_2 2/5+3\log_2 3/5)/5=0.971 \text{ (bit)}$$

故而，若以属性height为检测属性扩展后的T的实例平均信息量为：

$$I(T,\text{height})=(3*I(T_1^{(1)})+5*I(T_2^{(1)}))/8=0.951 \text{ (bit)}$$

由此可得属性height对实例T信息量为：

GI(T,height)=I(T)-I(T,height)=0.003 (bit)

GI(T,hair)=0.5 (bit)

GI(T,eyes)=0.347 (bit)

由于GI(T,hair)>GI(T,eyes)>GI(T,height)，故选取第2个属性 hair作为检测属性，并从属性表删除属性hair，更新属性表为：

AttrList={height,eyes}

至此生成的决策树如图7(a)所示

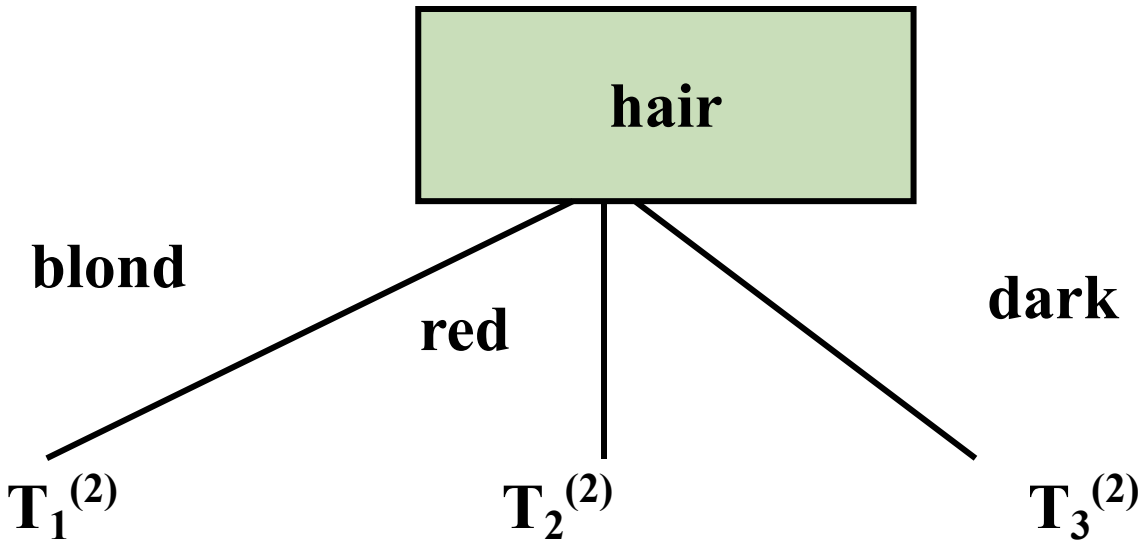


图7 生成的决策树(a)

图中的3个子节点$T_1^{(2)}$、$T_2^{(2)}$、$T_3^{(2)}$是由检测属性hair的值域 ValueType(hair)={blond,red,dark}的3个取值将T划分成3个子集，分别是：

$T_1^{(2)}$ ={<(short,blond,blue),+>,<(tall,blond,brown),- >,

<(tall,blond,blue),->,<(short,blond,brown),->}

$T_2^{(2)}$ ={<(tall,red,blue),+>}

$T_3^{(2)}$ ={<(short,dark,blue),->,<(tall,dark,blue),- >,

<(tall,dark,brown),->}

2）由于$T_2^{(2)}$中的决策树分类结果只有“+”，故用叶节点“+”代替节点$T_2^{(2)}$；$T_3^{(2)}$中的实例的分类结果都是“-”，故用叶节点“-”代替节点$T_3^{(2)}$。由于$T_1^{(2)}$中的分类结果仍有两种，故需按新属性表启发式选择检测属性来构造子树。

在实例集$T_1^{(2)}$中，分类结果为“+”的实例数为2，分类结果为“-”的实例数也为2，可得$T_1^{(2)}$的实例平均信息量为：

$$I(T_1^{(2)})=-(2\log_2 2/4+2\log_2 2/4)/4=1 \text{ (bit)}$$

若选新属性表AttrList={height,eyes}中的第1个属性height为检测属性，则由值域ValueType(height)={short,tall} 的2个取值将$T_1^{(2)}$分为2个子集$T_{11}^{(21)}$和$T_{12}^{(21)}$：

$T_{11}^{(21)}$ ={<(short,blond,blue),+>, <(short,blond,brown),->}

$T_{12}^{(21)}$ ={<(tall,blond,brown),>,<(tall,blond,brown),+>}

$T_{11}^{(21)}$和$T_{12}^{(21)}$中的分类结果为“+”的实例数都为1，分类结果为“-”的实例数也都为1，可得$T_{11}^{(21)}$和$T_{12}^{(21)}$的实例平均信息量分别为：

$I(T_{11}^{(21)}) = -(\log_2 1/2+\log_2 1/2)/2=1$ (bit)

$I(T_{12}^{(21)}) = -(\log_2 1/2+\log_2 1/2)/2=1$ (bit)

故而，若以属性height为检测属性扩展后的实例平均信息量为：

$$I(T_1^{(2)},\text{height})=(2*I(T_{11}^{(21)})+2*I(T_{12}^{(21)}))/4=1\ (\text{bit})$$

由此可得属性height对$T_1^{(2)}$实例集的信息量为：

$$GI(T_1^{(2)},\text{height})=I(T_1^{(2)})-I(T_1^{(2)},\text{height})=0$$

若选新属性表AttrList={height,eyes}中的第2个属性eyes为检测属性，由其值域ValueType(eyes)={blue,brown}的2个取值将$T_1^{(2)}$分为2个子集$T_{11}^{(22)}$和$T_{12}^{(22)}$：

$T_{11}^{(22)}$ ={<(short,blond,blue),+>, <(tall,blond,blue),->}

$T_{12}^{(22)}$ ={<(tall,blond,brown),>,<(short,blond,brown),+>}

由于height对实例集$T_1^{(2)}$的信息量GI($T_1^{(2)}$, height)=0,因此选择属性eyes作为检测属性，由eyes的2个取值扩展生成2个子集节点$T_{11}^{(22)}$和$T_{12}^{(22)}$。

至此，生成决策树如图7(b)。

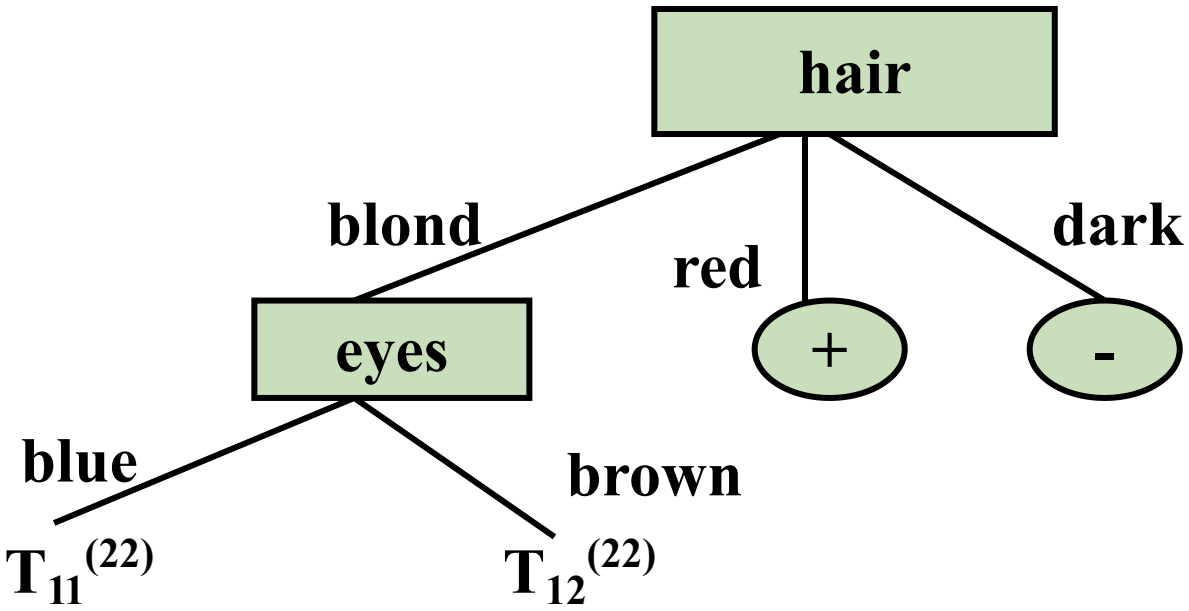


图7生成的决策树(b)

3）由于$T_{11}^{(22)}$中的决策树分类结果只有“+”，故用叶节点“+”代替节点$T_{11}^{(22)}$；$T_{12}^{(22)}$中的实例的分类结果都是“-”，故用叶节点“-”代替节点$T_{12}^{(22)}$。

至此，得到最后的完整决策树如图7(c)。

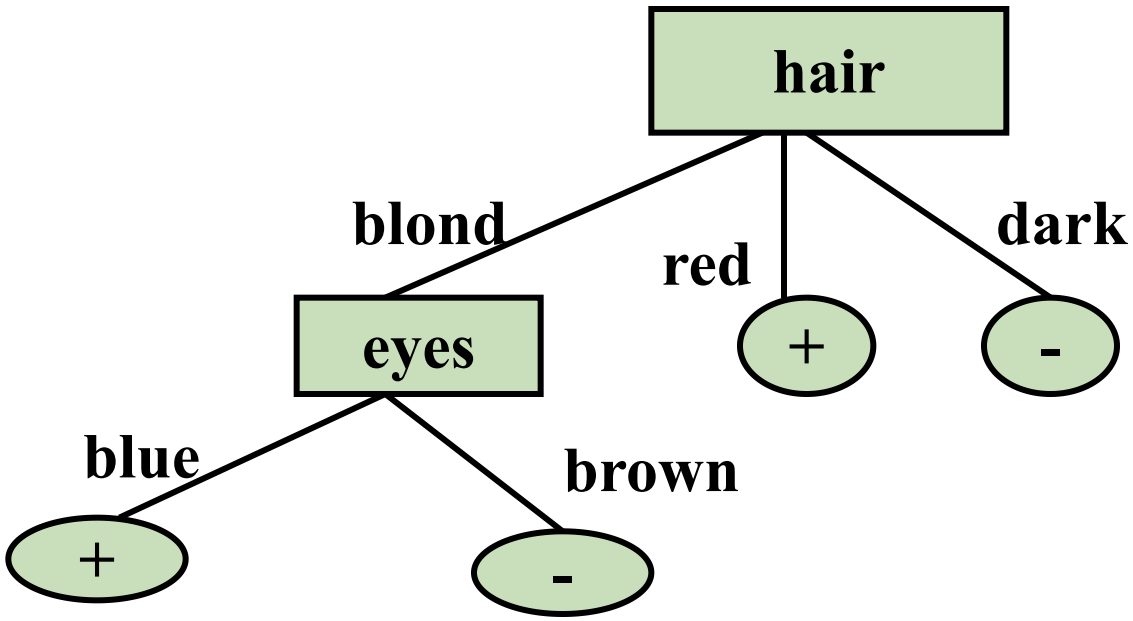


图7　生成的决策树(c)

练习

假设需要根据人员的外貌特征对人员进行分类，用于人员分类的外貌特征有3个，它们组成人员分类属性表为：AttrList={height，hair，eyes}

各属性的值域分别为：

ValueType(height)={short，tall}，

ValueType(hair)={blond，red，dark}，

ValueType(eyes)={blue，brown}。

对人员进行分类的结果有两种，分别用“+”和“—”表示，组成分类结果表为：

Class = { +，— }，提供学习的训练实例集为：

T = { < short，blond，blue，+ >，< (tall，red，brown)，— > ，

< (tall，red，blue)，+ >，　< (short，dark，blue)，+ >，

< (tall，dark，blue)，— >，< (tall，blond，blue)，— >　}

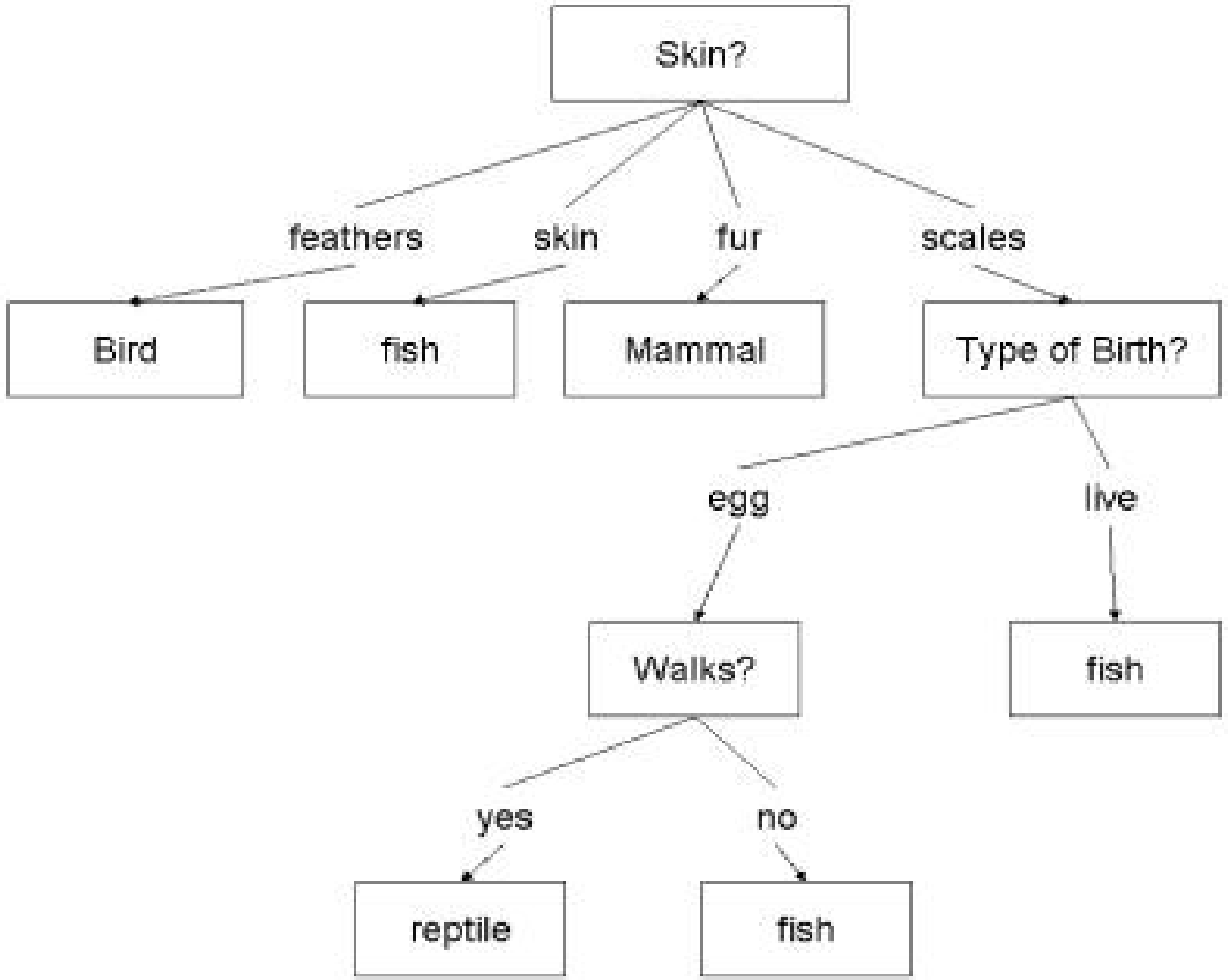
- 应用ID3算法构造决策树，请回答下面有关问题：

1. 计算划分前 T 的实例平均信息量I(T)
2. 若选第3个属性eyes为检测属性，则将T划分为哪几个子集？
3. 每个子集的的实例平均信息量是多少？
4. 每个子集的的实例总信息量是多少？
5. 划分后 T 的实例平均信息量 I(T，eyes)是多少？

Develop a simple table of examples to classify an animal as a fish, a bird, a reptile, or a mammal. Trace the construction of a decision tree by the ID3 algorithm.

Class	No.	Skin	Birth	Cares for young	Walks	Swims	Flies
Bird	1	Feathers	Egg	Yes	Yes	No	Yes
Bird	2	Feathers	Egg	Yes	Yes	No	No
Mammal	3	Fur	Live	Yes	Yes	Yes	No
Mammal	4	Fur	Live	Yes	Yes	Yes	No
Reptile	5	Scales	Egg	No	Yes	Yes	No
Fish	6	Skin	Live	No	No	Yes	No
Bird	7	Feathers	Egg	Yes	Yes	Yes	Yes
Mammal	8	Fur	Live	Yes	Yes	Yes	No
Fish	9	Skin	Egg	No	No	Yes	No
Mammal	10	Fur	Live	Yes	No	No	Yes
Fish	11	Scales	Egg	No	No	Yes	No
Bird	12	Feathers	Egg	Yes	Yes	No	No
Fish	13	Scales	Live	Yes	No	Yes	No
Mammal	14	Fur	Egg	Yes	Yes	Yes	No

the resulting decision tree:



5. Using Shannon's formula, show whether or not a message about the outcome of a fair six-sided die has more information than one about the outcome of a coin toss. What if the die message is "not three"?

Solution:

In Section 10.3.2 the information gain from a toss of a coin is shown to be one bit.

$$I[M] = \left(\sum_{i=1}^{n} - p(m_i) \log_2 p(m_i) \right)$$

$$I[M] = \left(\sum_{i=1}^{6} - \frac{1}{6} \log_2 \frac{1}{6} \right)$$

$$= 6 * \left(- \frac{1}{6} \log_2 \frac{1}{6} \right)$$

$$= 6 * \left(- \frac{1}{6} \log_2 \frac{1}{6} \right)$$

$$= \left(- \log_2 \frac{1}{6} \right)$$

= 2.59

Thus, information about the roll of a six-sided fair die contains more information than the toss of a coin.

If we know the die cannot roll a three, then Shannon's formula gives us:

$$I[M] = \left(\sum_{i=1}^{5} - \frac{1}{5} \log_2 \frac{1}{5} \right)$$

$$= \left(- \log_2 \frac{1}{5} \right)$$

= 2.32

which is consistent with the intuition that if something is less likely, it contains more information.

作业

- 2
- 4
- 6