Ch09_지식 공학과 데이터 마이닝

이 장에서 다룰 내용

- 01_지식 공학이란?
- 02_전문가 시스템은 어떤 문제를 해결할 수 있을까?
- 03_퍼지 전문가 시스템은 어떤 문제를 해결할 수 있을까?
- 04_인공 신경망은 어떤 문제를 해결할 수 있을까?
- 05_유전 알고리즘은 어떤 문제를 해결할 수 있을까?
- 06_하이브리드 지능 시스템은 어떤 문제를 해결할 수 있을까?
- 07_데이터 마이닝과 지식 발견
- 08_요약

01_지식 공학이란?

❖ 지식 공학

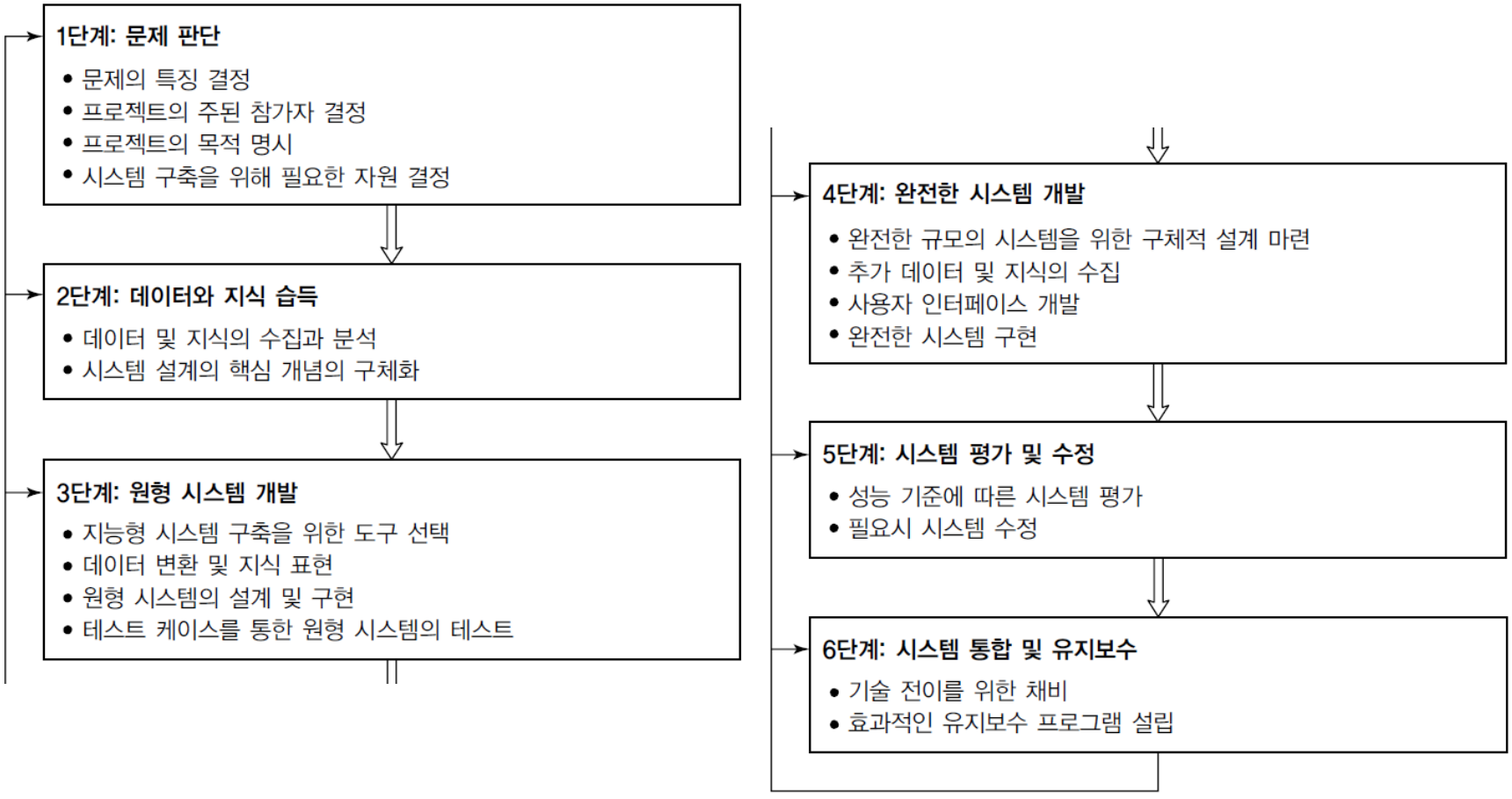
- 지능형 지식기반 시스템을 구축하는 과정을 지식 공학이라 한다.

- 지식형 시스템의 구축 과정
  - 지능형 시스템을 구축하는 과정은 해당 문제를 이해하는 데서 시작된다. 먼저 문제를 판단하고, 어떤 데이터를 사용할 수 있을지, 문제를 해결하는 데 필요한 것을 결정해야 한다.
  - 일단 문제를 이해하면 적당한 도구를 선택할 수 있고, 선택한 도구를 이용해 시스템을 개발할 수 있다.

- 지식 공학의 6단계
  - 1. 문제판단
  - 2. 데이터와 지식 습득
  - 3. 원형(prototype) 시스템 개발
  - 4. 완전한 시스템 개발
  - 5. 시스템 평가 및 수정
  - 6. 시스템 통합 및 유지보수
  - [그림9-1]은 지식공학의 과정을 설명한다. '지식공학'은 '공학'이라는 용어가 들어 있긴 하지만, 공학보다 인문학에 가깝다.

01_지식 공학이란?

- **지식 공학의 6단계 : [그림 9-1]**



[그림 9-1] 지식 공학의 과정

01_지식 공학이란?

- **지식 공학의 6단계 : 문제 판단**
  - **문제 판단 단계에서는 문제의 특징과 프로젝트 참여자를 결정하고 프로젝트의 목적을 규정하며, 어떤 자원이 시스템 구축에 필요한지 결정한다.**
  - **문제의 특성을 묘사하기 위해 문제 유형, 입출력 변수 및 그들 사이의 관계, 해의 형태와 내용을 결정해야 한다.**
  - **첫 번째 단계는 문제의 유형을 결정하는 것이다. [표9-1]은 지능형 시스템에서 종종 발생하는 전형적인 문제다. 여기에는 진단(diagnosis), 선택(selection), 예측(prediction), 분류(classification), 군집화(clustering), 최적화(optimization), 제어(control)가 있다.**

[표 9-1] 지능형 시스템에서 발생하는 전형적인 문제

문제 유형	설명
진단	사물의 행동을 바탕으로 기능 불량을 추론하고 해법을 추천함
선택	가능한 대안을 바탕으로 가장 좋은 것을 추천함
예측	사물의 과거 행동을 바탕으로 앞으로의 행동을 예측함
분류	정의된 종류 중 하나에 사물을 할당함
군집화	이종의 사물로 이루어진 그룹 하나를 동종의 그룹으로 나눔
최적화	최적해를 찾을 때까지 해의 품질을 개선함
제어	실시간으로 요구조건을 만족하도록 사물의 행동을 다스림

  - **해의 형태와 내용도 구축 도구를 선택하는 데 영향을 준다.**
  - **문제 유형은 지능형 시스템을 구축하기 위한 도구를 선택할 때 영향을 미친다.**

01_지식 공학이란?

- 지식 공학의 6단계 : 문제 판단
  - 두 번째 단계에서는 프로젝트 참여자를 결정한다. 지식 공학 프로젝트에서 가장 중요한 참여자는 지식공학자(지능형 시스템을 설계, 구축 및 테스트할 수 있는 사람)와 주제 전문가(특정 영역에서 문제를 풀 수 있는 학식 있는 사람)다.
  - 마지막 세 번째 단계는 어떤 자원이 시스템 구축에 필요한지 결정한다. 여기에는 컴퓨터 자원, 개발 소프트웨어, 지식 및 데이터 소스(인간 전문가, 책, 매뉴얼, 웹 사이트, 데이터베이스, 사례)가 포함된다. 물론 돈도 포함된다.

- 지식 공학의 6단계 : 데이터와 지식 습득
  - 지식 공학 단계에서는 데이터와 지식을 모으고 분석하여 문제를 조금 더 깊이 이해함으로써 시스템 설계의 핵심 개념을 구체화한다.
  - 지능형 시스템 구축에 사용하는 특정 도구에는 특정 형태의 데이터가 필요하다. 몇몇 도구는 연속된 변수를 다루는 반면 어떤 도구에는 여러 영역으로 나뉜 변수가 필요하거나 0~1 사이의 고정 범위로 정규화시킬 필요가 있다.
  - 특정 도구에서 사용할 수 있도록 데이터를 변환해야 한다(즉 정규화1해야 한다).
  - 어떤 도구를 선택하든 데이터를 정규화하기 전에 다음 세 가지 문제를 해결해야 한다.

[세 가지 문제]

  - 문제 1. 호환되지 않는 데이터 : 지능형 시스템을 구축하는 데 사용하는 도구와 데이터가 호환되지 않는 경우가 종종 있다. 이 문제는 보통 요구된 데이터 형태로 변환하는 코드를 자동으로 만들어 주는 데이터 변환도구로 해결한다.

01_지식 공학이란?

- 지식 공학의 6단계 : 데이터와 지식 습득

  [세 가지 문제]

  - 문제 2. 모순된 데이터 : 같은 사실을 데이터베이스에 따라 다르게 표현하는 일이 많다. 이런 차이를 제때 인식하고 해결하지 못하면, 잘못된 분석 결과를 얻게 된다.
  - 문제 3. 분실한 데이터 : 가끔 실제 데이터 레코드에 비어있는 필드가 있다. 그와 같은 불완전한 데이터를 버리기도 하지만, 보통 비어 있는 필드에서 유용한 정보를 추론하려 한다. 대부분의 경우에는 비어 있는 필드를 가장 보편적이거나 평균값으로 채운다. 어떤 경우에는 특정 필드가 채워지지 않았다는 사실 자체를 매우 유용한 정보로 제공한다.

  - 시스템 구축 도구는 습득한 데이터에 따라 달라진다.
  - 데이터 습득 작업은 지식 습득 작업과 관련 있다. 실제로 데이터를 수집하는 동안 문제 영역에 관한 일부 지식을 얻을 수 있다.

  [지식 습득의 과정]

  - 보통은 문서를 재검토하고 문제와 관련된 책, 논문, 매뉴얼을 읽는 것으로 시작한다. 일단 문제에 익숙해지면 주제 전문가를 인터뷰하면서 더 많은 지식을 모을 수 있다. 그런 후 습득한 지식을 공부하고 분석하며, 전 과정을 다시 반복한다.
  - 문제의 분야를 이해하는 과정은 지능형 시스템 구축에서 매우 중요하다.

  - 지식 공학의 두 번째 단계를 거치면서 얻은 데이터와 지식을 바탕으로 가장 추상적인 (개념적인) 수준에서 문제 해결 전략을 설명하고, 원형 시스템을 구축할 도구를 선택할 수 있다.

01_지식 공학이란?

- 지식 공학의 6단계 : 원형 시스템 개발
  - 원형 시스템 개발은 실제로 지능형 시스템(더 정확히 말하면, 시스템보다 작은 버전)을 만들고 수 많은 테스트 케이스로 시스템을 검사하는 것을 포함한다.

[원형 시스템]

- 원형 시스템은 최종 시스템의 기능을 간략하게 구현한 시스템으로, 문제를 얼마나 잘 이해하고 있는지 검사하기 위해 설계한다.
- 문제 풀이 전략과 시스템 구축을 위해 선택된 도구 그리고 습득한 데이터와 지식을 표현하는 기법이 현 작업에 적절한지 확인하기 위한 것이다.
- 도구를 선택하고 데이터를 정규화하여 도구에 적합한 형태로 지식을 표현한 후, 시스템의 원형버전을 설계하고 이를 구현한다.
- 일단 원형 시스템을 구축했으면 다양한 테스트 케이스로 시스템을 검사하여(보통은 주제 전문가와 함께) 성능을 살펴본다.

[테스트 케이스]

- 테스트 케이스(test case)란 과거에 성공적으로 해결하여 입력 데이터와 출력 해를 이미 알고 있는 문제를 말한다.
- 검사를 진행하는 동안 시스템은 성공적으로 해결한 문제와 같은 입력 데이터를 이용하여 출력 해를 구하고, 이를 성공적으로 해결한 문제의 출력 해와 비교한다.

01_지식 공학이란?

- 지식 공학의 6단계 : 원형 시스템 개발

  [시스템 구축 도구를 잘 못 선택 시]

  - 현재의 원형 시스템을 버리고 원형화 단계를 다시 시작해야 한다.
  - 잘못 선택한 도구를 적합하지 않은 문제에 끼워 맞추려는 시도를 할수록 시스템 개발은 지연되기만 할 뿐이다.
  - 원형화 단계의 핵심 목표는 문제를 더 잘 이해하게 만드는 것이므로 새로운 도구로 이 단계를 시작함으로써 시간과 돈을 낭비하지 않게 된다.

- 지식 공학의 6단계 : 완전한 시스템 개발
  - 원형 시스템이 만족스럽게 작동하기 시작하면 완전한 시스템을 개발하는 데 관련된 실제 요소를 판단할 수 있다. 완전한 시스템을 위한 계획, 스케줄 및 예산을 알 수 있고, 시스템의 성능 기준도 명확히 정의할 수 있다.
  - 이 단계에서 주된 일은 시스템에 데이터와 지식을 추가하는 일과 종종 관련 있다. 예를 들어, 진단 시스템을 개발한다면 특정 경우를 다루기 위해 더 많은 규칙을 제공해야 한다.
  - 지능형 시스템의 개발은 진화적인 과정이다. 프로젝트를 진행하면서 새로운 데이터와 지식을 모으고 시스템에 추가함으로써 능력을 향상시킨다. 이런 과정을 통해 원형 시스템은 점차 최종 시스템으로 진화한다.

01_지식 공학이란?

- 지식 공학의 6단계 : 시스템 평가 및 수정
  - 보통의 컴퓨터 프로그램과 달리 지능형 시스템은 '올바른' 그리고 '틀린' 해를 명확히 정의하지 못하는 문제를 풀기 위해 설계된다.
  - 시스템 성능이 사용자가 만족할만한 정도임을 확실히 해두기 위해 지능형 시스템을 평가한다. 시스템의 공식 평가는 보통 사용자가 선택한 테스트 케이스로 진행한다. 시스템 성능은 원형 단계의 마지막에 얻은 성능 기준과 비교한다.
  - 평가에서 시스템의 한계와 약점이 종종 발견된다. 따라서 해당 시스템을 재검토하고 관련 개발 단계를 반복하여 수정한다.

- 지식 공학의 6단계 : 시스템 통합 및 유지보수
  - 시스템 통합 및 유지 보수 단계에서는 시스템을 동작 환경에 통합하고 효과적인 유지보수 프로그램을 만드는 것을 포함한다.
  - '통합'은 한 체제 내에서 새로운 지능형 시스템을 기존 시스템에 조화시키고 기술 전이를 준비하는 것을 뜻한다. 사용자가 시스템의 사용 및 유지 방법을 아는지 확인해야 한다.
  - 지능형 시스템은 지식기반 시스템이고, 지식은 시간에 따라 바뀌기 때문에 시스템을 변화시켜야 한다.

02_전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 전문가 시스템의 문제 해결

- '전화 걸기 규칙'
  - 오래되긴 했지만, 전문가 시스템을 사용하기에 적합한지 알아보는 유용한 테스트가 있다. 이는 전화 걸기 규칙(Phone Call Rule)이다
  - 전문가와 10~30분 정도의 통화로 해결할 수 있는 문제는 전문가 시스템으로 개발할 수 있다(Firebaugh, 1988).

- 진단과 고장 원일을 찾는 문제
  - 진단과 고장 원인을 찾는 문제는 전문가 시스템 기술을 사용하기에 매우 적합한 후보다. 물론 컴퓨터 진단도 그 중 하나다. 의학 진단은 전문가 시스템을 적용한 첫 번째 분야였고, 그 이후에는 공학과 제조분야에 자주 적용되었다.
  - 진단 전문가 시스템은 개발하기가 상대적으로 쉽다. 대부분의 진단 문제는 해의 개수가 유한하고, 제한된 양의 잘 형식화된 지식을 포함하며, 인간 전문가가 문제를 해결할 경우 한 시간 정도의 짧은 시간이 걸린다.

- 전문가 시스템 개발 도구 선택
  - 일반적으로 문제의 특징에 맞는 도구를 찾아내야 한다. 도구에는 LISP, PROLOG, OPS, C, Java와 같은 고급 프로그래밍 언어와 전문가 시스템 틀이 있다.
  - 고급 프로그래밍 언어는 훨씬 유연하고 다양한 프로젝트의 요구 사항을 맞춰줄 수 있지만, 고급 프로그래밍 기술을 갖추고 있어야 한다.

02_전문가 시스템은 어떤 문제를 해결할 수 있을까?

- 전문가 시스템 개발 도구 선택
  - 전문가 시스템 틀에는 프로그래밍 언어의 유연성은 없지만, 이미 짜여있는 추론 엔진, 설명 설비 및 사용자 인터페이스를 제공한다. 이틀을 사용할 때는 프로그래밍 기술이 필요 없다. 다만 틀의 기반지식에 영어로 규칙을 써넣으면 된다. 이런 특성 때문에 전문가 시스템 틀은 원형 시스템을 빠르게 구축하는 데 특히 유용하다.

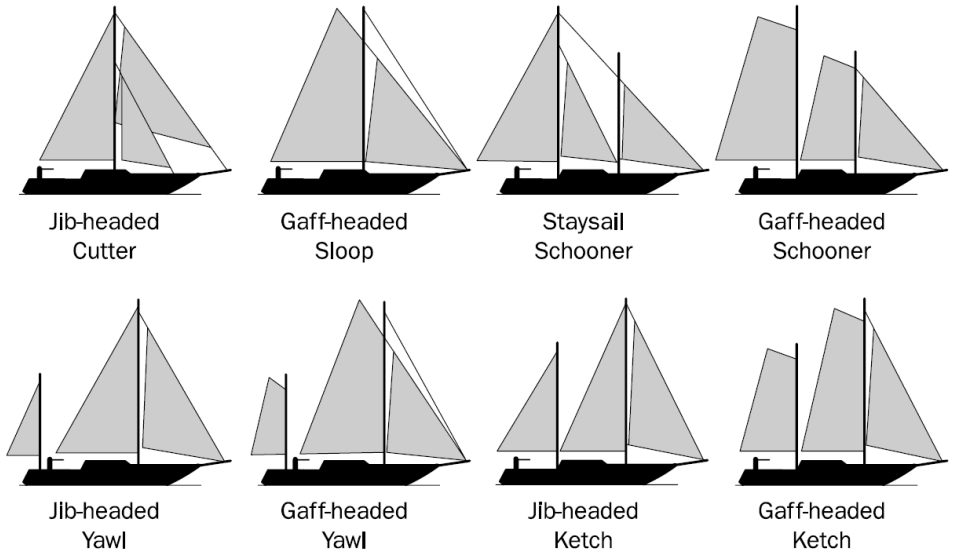
- 틀 선택
  - 일반적으로 전문가 시스템 틀을 고를 때는 그 틀이 지식(규칙이나 프레임)을 어떻게 표현하고, 어떤 추론 메커니즘(순방향 또는 역방향 추론)을 사용하는지, 부정확한 추론을 지원하는지, 지원한다면 어떤 기법(베이즈 추론, 확신도, 혹은 퍼지 논리)을 사용하는지 고려해야 한다.
  - 외부 데이터 파일에 접근을 허용하는 '열린' 구조인지, 사용자가 전문가 시스템과 어떻게 상호작용할 것인지(그래픽 사용자 인터페이스나 하이퍼텍스트)를 고려해야 한다.

02_전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 : 분류 전문가 시스템

- 돗단배의 종류를 구분하도록 도와주는 전문가 시스템을 개발하고 싶다. 이 문제에 전문가 시스템을 적용할 수 있을까?

  - 이는 전형적인 분류 문제다. 배를 구분한다는 것은 정의한 종류 중 하나로 지정하는 것과 같다. 그리고 앞서 언급했듯이 이런 문제는 전문가 시스템과 신경망으로 잘 해결할 수 있다. 만약 전문가 시스템을 구축해야 한다면 각 돛단배의 돛대 구조와 돛의 설계도에 관한 정보를 수집하는 것부터 시작해야 한다.
  - [그림 9-4]에서 돗단배의 종류 8가지 보여준다. 돛의 설계도로 각각의 돛단배를 구별할 수 있다.



[그림 9-4] 8가지 종류의 돛단배

02_전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 : 분류 전문가 시스템

- [그림 9-5]는 돛단배를 분류하는 일련의 규칙을 레오나르도 코드로 보여준다. 사용자와 대화하는 동안 시스템은 알려지지 않은 돛단배의 주 돛의 모양뿐만 아니라 돛대의 개수 및 위치 정보를 얻는다.
- 그런 후 [그림 9-4]에서 제시한 8종류의 돛단배를 구별한다.

```
/* 돛단배 분류 전문가 시스템: Mark 1
규칙 1:  if    ('돛대의 수'가 하나다)
         and   ('주 돛의 모양'이 삼각형이다)
         then  (배는 'Jib-headed Cutter'이다)

규칙 2:  if    ('돛대의 수'가 하나다)
         and   ('주 돛의 모양'이 사각형이다)
         then  (배는 'Gaff-headed Sloop'이다)

규칙 3:  if    ('돛대의 수'가 둘이다)
         and   ('주 돛대의 위치'는 '짧은 돛대의 앞쪽'이다)
         and   ('짧은 돛대의 위치'는 '키의 앞쪽'이다)
         and   ('주 돛의 모양'이 삼각형이다)
         then  (배는 'Jib-headed Ketch'이다)

규칙 4:  if    ('돛대의 수'가 둘이다)
         and   ('주 돛대의 위치'는 '짧은 돛대의 앞쪽'이다)
         and   ('짧은 돛대의 위치'는 '키의 앞쪽'이다)
         and   (주 돛의 모양'이 사각형이다)
         then  (배는 'Gaff-headed Ketch'이다)

규칙 5:  if    ('돛대의 수'가 둘이다)
         and   ('주 돛대의 위치'는 '짧은 돛대의 앞쪽'이다)
         and   ('짧은 돛대의 위치'는 '키의 뒤쪽'이다)
         and   ('주 돛의 모양'이 삼각형이다)
         then  (배는 'Jib-headed Yawl'이다)

규칙 6:  if    ('돛대의 수'가 둘이다)
         and   ('주 돛대의 위치'는 '짧은 돛대의 앞쪽'이다)
         and   ('짧은 돛대의 위치'는 '키의 뒤쪽' 이다)
         and   ('주 돛의 모양'이 사각형이다)
         then  (배는 'Gaff-headed Yawl'이다)

규칙 7:  if    ('돛대의 수'가 둘이다)
         and   ('주 돛대의 위치'는 '짧은 돛대의 뒤쪽'이다)
         and   ('주 돛의 모양'이 사각형이다)
         then  (배는 'Gaff-headed Schooner'이다)

규칙 8:  if    ('돛대의 수'가 둘이다)
         and   ('주 돛대의 위치'는 '짧은 돛대의 뒤쪽'이다)
         and   ('주 돛의 모양'이 '두 개의 앞 돛이 달린 삼각형'이다)
         then  (배는 'Staysail Schooner'이다)
/*************************************************
/* SEEK 명령어가 결론을 정한다.
seek 배
```

02_전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 : 분류 전문가 시스템

- [그림 9-6]은 확신도로 돛단배 분류 문제를 풀기 위해 총망라된 규칙을 보여준다.

```
/* 돛단배 분류 전문가 시스템: Mark 2
control cf
규칙 1:  if    ('돛대의 수' 가 하나다)
         then  (배는 'Jib-headed Cutter' 이다)                {cf 0.4}
               (배는 'Gaff-headed Sloop' 이다)                {cf 0.4}

규칙 2:  if    ('돛대의 수' 가 하나다) and ('주 돛의 모양' 이 삼각형이다)
         then  (배는 'Jib-headed Cutter' 이다)                {cf 1.0}

규칙 3:  if    ('돛대의 수' 가 하나다) and ('주 돛의 모양' 이 사각형이다)
         then  (배는 'Gaff-headed Sloop' 이다)                {cf 1.0}

규칙 4:  if    ('돛대의 수' 가 둘이다)
         then  (배는 'Jib-headed Ketch' 이다)                 {cf 0.1}
               (배는 'Gaff-headed Ketch' 이다)                {cf 0.1}
               (배는 'Jib-headed Yawl' 이다)                  {cf 0.1}
               (배는 'Gaff-headed Yawl' 이다)                 {cf 0.1}
               (배는 'Gaff-headed Schooner' 이다)             {cf 0.1}
               (배는 'Staysail Schooner' 이다)                {cf 0.1}

규칙 5:  if    ('돛대의 수' 가 둘이다) and ('주 돛대의 위치' 는 '짧은 돛대의 앞쪽' 이다)
         then  (배는 'Jib-headed Ketch' 이다)                 {cf 0.2}
               (배는 'Gaff-headed Ketch' 이다)                {cf 0.2}
               (배는 'Jib-headed Yawl' 이다)                  {cf 0.2}
               (배는 'Gaff-headed Yawl' 이다)                 {cf 0.2}

규칙 6:  if    ('돛대의 수' 가 둘이다) and ('주 돛대의 위치' 는 '짧은 돛대의 뒤쪽' 이다)
         then  (배는 'Gaff-headed Schooner' 이다)             {cf 0.4}
               (배는 'Staysail Schooner' 이다)                {cf 0.4}

규칙 7:  if    ('돛대의 수' 가 둘이다) and ('짧은 돛대의 위치' 는 '키의 앞쪽' 이다)
         then  (배는 'Jib-headed Ketch' 이다)                 {cf 0.4}
               (배는 'Gaff-headed Ketch' 이다)                {cf 0.4}

규칙 8:  if    ('돛대의 수' 가 둘이다)
         and   ('짧은 돛대의 위치' 는 '키의 뒤쪽' 이다)
         then  (배는 'Jib-headed Yawl' 이다)                  {cf 0.2}
               (배는 'Gaff-headed Yawl' 이다)                 {cf 0.2}
               (배는 'Gaff-headed Schooner' 이다)             {cf 0.2}
               (배는 'Staysail Schooner' 이다)                {cf 0.2}

규칙 9:  if    ('돛대의 수' 가 둘이다) and ('주 돛의 모양' 이 삼각형이다)
         then  (배는 'Jib-headed Ketch' 이다)                 {cf 0.4}
               (배는 'Jib-headed Yawl' 이다)                  {cf 0.4}

규칙 10: if    ('돛대의 수' 가 둘이다) and ('주 돛의 모양' 이 사각형이다)
         then  (배는 'Gaff-headed Ketch' 이다)                {cf 0.3}
               (배는 'Gaff-headed Yawl' 이다)                 {cf 0.3}
               (배는 'Gaff-headed Schooner' 이다)             {cf 0.3}

규칙 11: if    ('돛대의 수' 가 둘이다) and ('주 돛의 모양' 이 '두 개의 앞 돛이 있는 삼각형' 이다)
         then  (배는 'Staysail Schooner' 이다)                {cf 1.0}

         seek  배
```

[그림 9-6] 배 분류 전문가 시스템에서 불확실성 다루기

03_퍼지 전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 퍼지 전문가 시스템의 문제 해결

- 적합한 후보 결정
  - 어느 문제가 퍼지 기술에 적합한 후보인지 정해야 한다. 간단하고 기본적인 방법은 가능한 상황에 대한 정확한 규칙 집합을 일일이 정의할 수 없을 때 퍼지 논리를 사용하는 것이다.
  - 확신도와 베이즈 확률은 잘 정의된 사건에서 나오는 결과의 부정확성과 관계가 있는 반면, 퍼지 논리는 사건자체의 부정확성에 집중한다. 다시 말해서, 문제에 내재된 부정확한 특성이 있으면 그 문제는 퍼지 기술을 적용하기에 적합한 후보가 된다.

- 퍼지 전문가 시스템의 문제 해결
  - 퍼지 시스템은 특히 인간의 의사 결정을 모델링하는 데 적합하다. 우리는 중요한 결정을 내릴 때 흔히 상식에 의존하고, 막연하거나 모호한 용어를 사용한다.
  - 예를 들어, 의사가 수술 후 회복실에있는 환자를 일반 병실로 옮길 때, 획일적으로 적용할 수 있는 기준은 없다. 의사는 체온 같은 인수 하나의 정확성에 의존하지 않고, 환자가 수술 후 회복실을 떠나려는 의지처럼 모호한 표현도 포함한 여러 인수를 평가하여 정확한 판단을 한다.

- 퍼지 전문가 시스템의 응용분야
  - 퍼지 기술은 여전히 제어나 공학 분야에서 주로 응용되고 있지만, 사업과 금융 분야에서 더 큰 잠재력을 보인다

03_퍼지 전문가 시스템은 어떤 문제를 해결할 수 있을까?

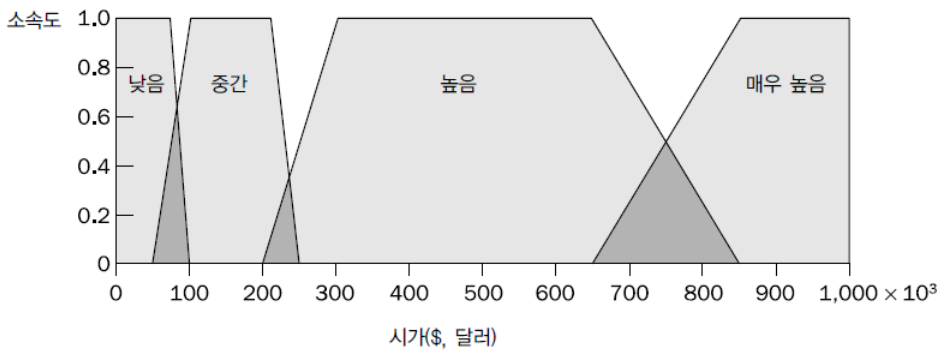
❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템

- 담보 대출 신청을 평가하는 지능 시스템을 개발하고 싶다. 이 문제에 퍼지 전문가 시스템을 적용할 수 있을까?

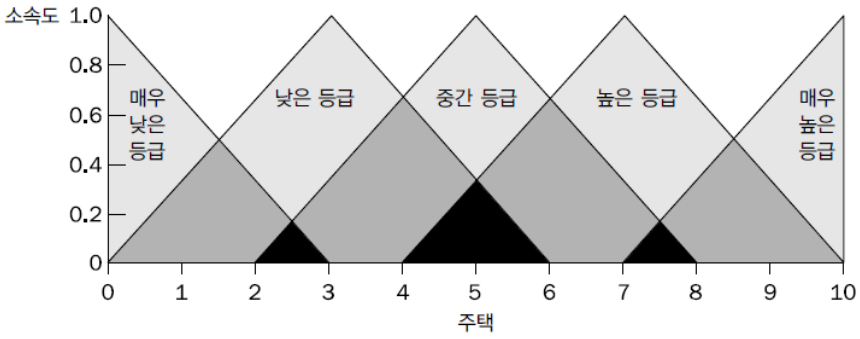
  - 담보 대출 신청 평가는 의사 결정 지원 퍼지 시스템(decision-support fuzzy system)을 성공적으로 적용한 전형적인 문제다(von Altrock, 1997).
  - 이 문제를 다루는 의사 결정 지원 퍼지 시스템을 개발하려면, 먼저 담보 대출 신청 평가의 기본 개념을 퍼지 용어로 나타내야 한다. 그런 다음 적절한 퍼지 도구를 사용하여 이 개념을 원형 시스템으로 구현하고, 선택된 시험 사례에 대해 시스템을 테스트하고 최적화한다.
  - 담보 대출 신청은 보통 주택의 시가, 위치, 신청자의 자산과 소득, 상환 계획에 근거하여 평가한다. 상환 계획은 신청자의 소득과 은행의 이자율에 따라 결정된다.

- 담보 대출 신청 평가에 대한 소속 함수와 규칙
  - 소속 함수를 정의하고 퍼지 규칙을 세우려면 보통 경험 많은 담보 대출 전문가와 담보 대출 승인정책을 개발한 은행 지점장의 도움이 필요하다. [그림 9-7]~[그림 9-14]는 이 문제에서 사용한 언어 변수에 대한 퍼지 집합을 보여준다.
  - 삼각형과 사다리꼴 소속 함수는 담보 대출 전문가의 지식을 적절히 나타낼 수 있다.

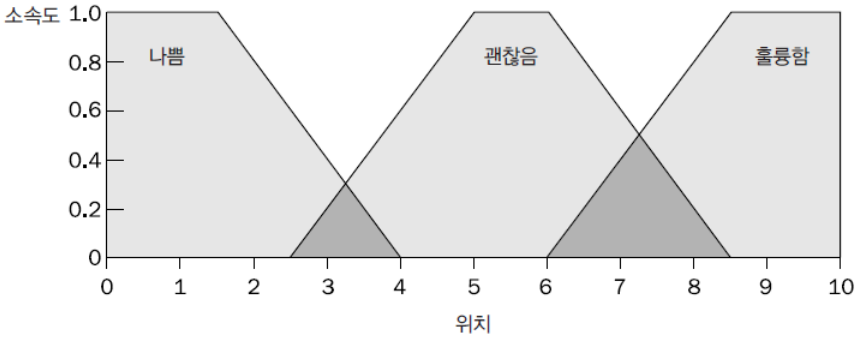
❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템



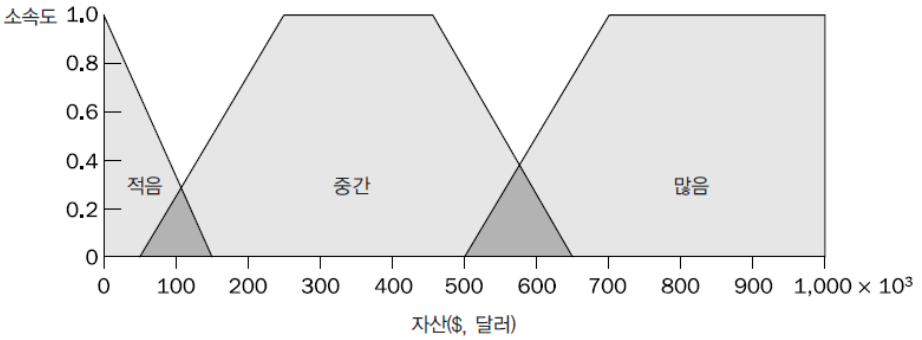
[그림 9-7] 언어 변수 **시가**에 대한 퍼지 집합



[그림 9-9] 언어 변수 **주택**에 대한 퍼지 집합

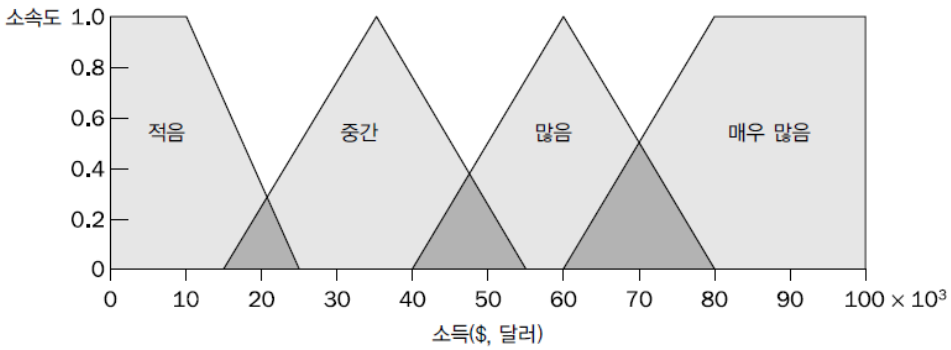


[그림 9-8] 언어 변수 **위치**에 대한 퍼지 집합

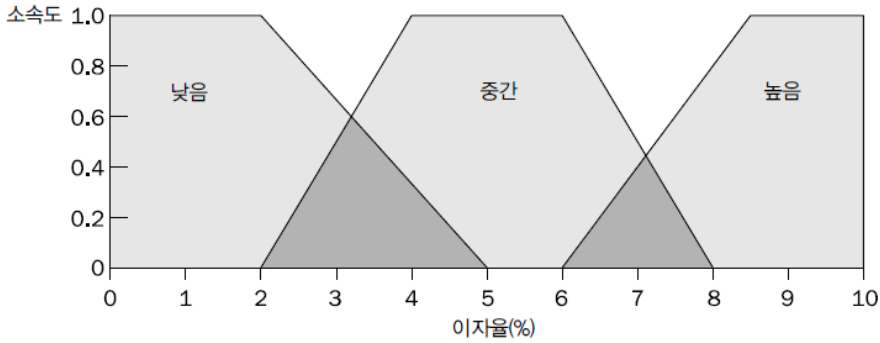


[그림 9-10] 언어 변수 **자산**에 대한 퍼지 집합

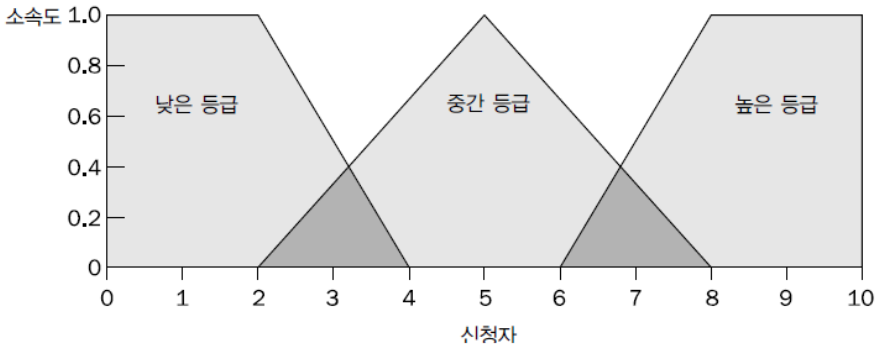
❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템



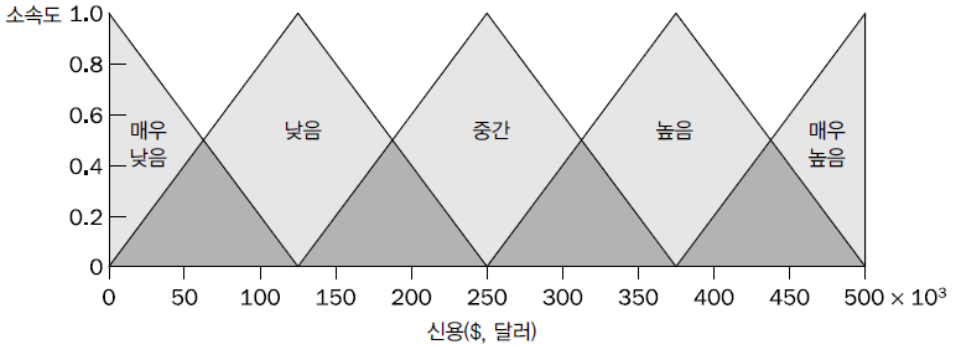
[그림 9-11] 언어 변수 **소득**에 대한 퍼지 집합



[그림 9-13] 언어 변수 **이자율**에 대한 퍼지 집합



[그림 9-12] 언어 변수 **신청자**에 대한 퍼지 집합



[그림 9-14] 언어 변수 **신용**에 대한 퍼지 집합

03_퍼지 전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템

- **퍼지 규칙을 구한다. 이 경우에는 폰 알트락이 담보 대출 평가에 대한 퍼지 모델에서 사용한 기본 규칙의 일부를 개조하면 된다. 이 규칙은 [그림 9-15]와 같다.**

[기반 규칙 1: 주택 평가]

1. If (시가가 낮다) then (주택은 낮은 등급)
2. If (위치가 나쁘다) then (주택은 낮은 등급)
3. If (위치가 나쁘다) and (시가가 낮다) then (주택은 매우 낮은 등급)
4 If (위치가 나쁘다) and (시가가 중간이다) then (주택은 낮은 등급)
5. If (위치가 나쁘다) and (시가가 높다) then (주택은 중간 등급)
6. If (위치가 나쁘다) and (시가가 매우 높다) then (주택은 높은 등급)
7. If (위치가 괜찮다) and (시가가 낮다) then (주택은 낮은 등급)
8 If (위치가 괜찮다) and (시가가 중간이다) then (주택은 중간 등급)
9. If (위치가 괜찮다) and (시가가 높다) then (주택은 높은 등급)
10. If (위치가 괜찮다) and (시가가 매우 높다) then (주택은 매우 높은 등급)
11. If (위치가 훌륭하다) and (시가가 낮다) then (주택은 중간 등급)
12. If (위치가 훌륭하다) and (시가가 중간이다) then (주택은 높은 등급)
13. If (위치가 훌륭하다) and (시가가 높다) then (주택은 매우 높은 등급)
14. If (위치가 훌륭하다) and (시가가 매우 높다) then (주택은 매우 높은 등급)

❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템

- **퍼지 규칙** : [그림 9-15]

[기반 규칙 2: 신청자 평가]

1. If (자산이 적다) and (수입이 적다) then (신청자는 낮은 등급)
2. If (자산이 적다) and (수입이 중간이다) then (신청자는 낮은 등급)
3. If (자산이 적다) and (수입이 많다) then (신청자는 중간 등급)
4 If (자산이 적다) and (수입이 매우 많다) then (신청자는 높은 등급)
5. If (자산이 중간이다) and (수입이 적다) then (신청자는 낮은 등급)
6 If (자산이 중간이다) and (수입이 중간이다) then (신청자는 중간 등급)
7. If (자산이 중간이다) and (수입이 많다) then (신청자는 높은 등급)
8. If (자산이 중간이다) and (수입이 매우 많다) then (신청자는 높은 등급)
9. If (자산이 많다) and (수입이 적다) then (신청자는 중간 등급)
10. If (자산이 많다) and (수입이 중간이다) then (신청자는 중간 등급)
11. If (자산이 많다) and (수입이 많다) then (신청자는 높은 등급)
12. If (자산이 많다) and (수입이 매우 많다) then (신청자는 높은 등급)

❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템

- **퍼지 규칙** : [그림 9-15]

[기반 규칙 3: 신용 평가]

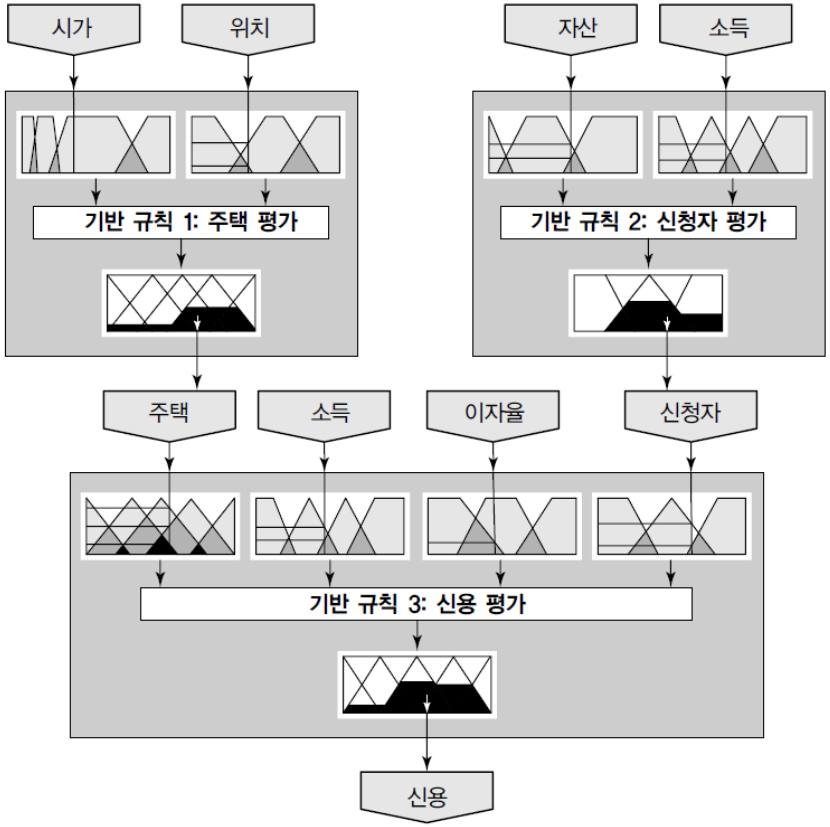
1. If (수입이 적다) and (이자율이 중간이다) then (신용은 매우 낮다)
2. If (수입이 적다) and (이자율이 높다) then (신용은 매우 낮다)
3. If (수입이 중간이다) and (이자율이 높다) then (신용은 낮다)
4. If (신청자가 낮은 등급) then (신용은 매우 낮다)
5. If (주택이 낮은 등급) then (신용은 매우 낮다)
6. If (신청자가 중간 등급) and (주택이 매우 낮은 등급) then (신용은 낮다)
7. If (신청자가 중간 등급) and (주택이 낮은 등급) then (신용은 낮다)
8. If (신청자가 중간 등급) and (주택이 중간 등급) then (신용은 중간이다)
9. If (신청자가 중간 등급) and (주택이 높은 등급) then (신용은 높다)
10. If (신청자가 중간 등급) and (주택이 매우 높은 등급) then (신용은 높다)
11. If (신청자가 높은 등급) and (주택이 매우 낮은 등급) then (신용은 낮다)
12. If (신청자가 높은 등급) and (주택이 낮은 등급) then (신용은 중간이다)
13. If (신청자가 높은 등급) and (주택이 중간 등급) then (신용은 높다)
14. If (신청자가 높은 등급) and (주택이 높은 등급) then (신용은 높다)
15. If (신청자가 높은 등급) and (주택이 매우 높은 등급) then (신용은 매우 높다)

[그림 9-15] 담보 대출 평가를 위한 규칙들

03_퍼지 전문가 시스템은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템

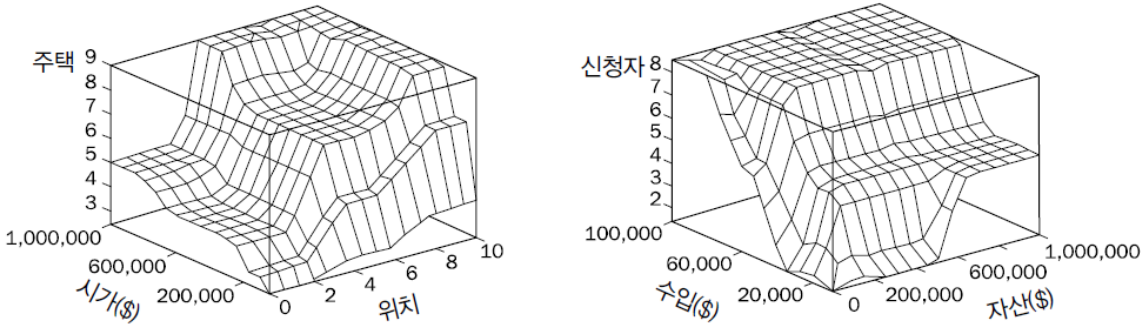
- 퍼지 시스템에서 사용된 모든 변수 사이의 복잡한 관계는 [그림 9-16]의 계층적 구조로 가장 잘 나타낼 수 있다. 시스템을 구축하기 위해 MATLAB Fuzzy Logic Toolbox를 사용한다. 이것은 현재 업계에서 가장 인기 있는 퍼지 도구다.



[그림 9-16] 담보 대출 평가에 대한 계층적 퍼지 모델

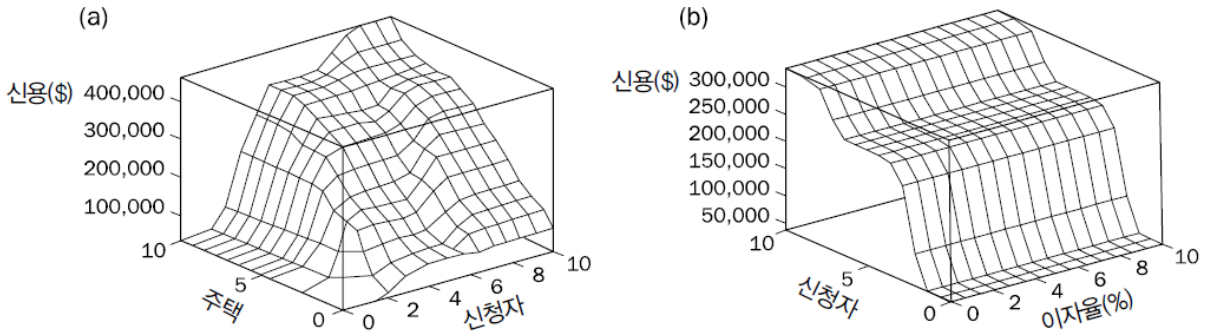
❖ 사례 연구 3: 의사 결정 지원 퍼지 시스템

- 원형 시스템 개발의 마지막 단계는 평가와 테스트다. 퍼지 시스템의 성능을 평가하고 분석하기 위해 Fuzzy Logic Toolbox에서 제공하는 출력 도면 뷰어를 사용할 수 있다. [그림 9-17]과 [그림 9-18]은 담보 대출 평가를 위한 퍼지 시스템의 3차원 도면을 나타낸다. 담보 대출 전문가는 마지막으로 몇몇 사례로 시스템을 테스트할 것이다.



(a) 기반 규칙 1에 대한 3차원 도면 (b) 기반 규칙 2에 대한 3차원 도면

[그림 9-17] 기반 규칙 1과 기반 규칙 2에 대한 3차원 도면



[그림 9-18] 기반 규칙 3에 대한 3차원 도면

04_인공 신경망은 어떤 문제를 해결할 수 있을까?

❖ 인공 신경망의 문제 해결

- 인공 신경망의 응용 분야.
  - 신경망은 예측, 분류, 군집화 문제에 성공적으로 적용했던 매우 강력한 범용 도구다.
  - 신경망은 언어 및 문자 인식, 사기 거래 탐지, 심장병 진단, 프로세스 제어, 로보틱스, 환율 예측, 레이더 표적탐지 및 구별 등 다양한 분야에서 사용된다. 신경망의 응용 분야는 빠르게 확산되고 있다.

- 인공 신경망의 특징
  - 다재다능한 신경망은 이진 데이터와 연속 데이터를 다룰 수 있고, 복잡한 분야에서 좋은 결과를 만들어내기 때문에 인기가 많다.
  - 신경망은 출력이 연속일 때 예측 문제를 다룰 수 있지만, 출력이 이진 값일 때는 분류자로 동작한다.

04_인공 신경망은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 4: 문자 인식 신경망

- 문자 인식 시스템을 개발하고 싶다. 이 문제에 신경망을 적용할 수 있을까?

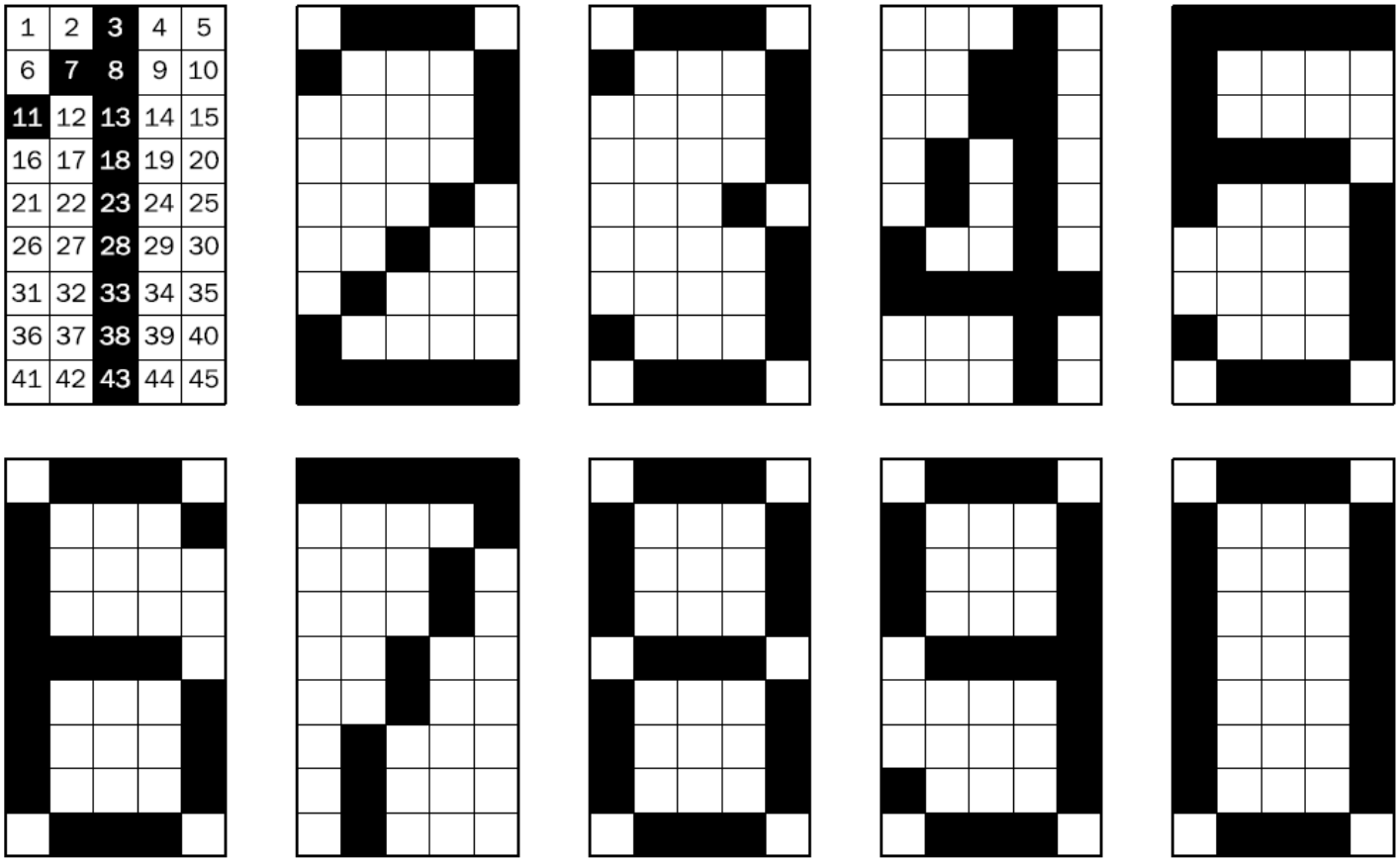
  - 인식(optical character recognition) 시스템은 신경망을 상업적으로 응용한 예 중 하나다.

- 광학 문자 인식
  - 특별한 소프트웨어를 사용하여 문자 이미지를 텍스트 파일로 바꾸는 것은 컴퓨터의 능력이다. 이는 인쇄된 문서를 다시 타이핑하지 않고도 컴퓨터에서 편집할 수 있는 형태로 넣을 수 있게 해준다.
  - 문자 이미지를 캡처하려면 스캐너를 사용한다. 스캐너는 인쇄물의 채색된 면 위로 빛에 민감한 센서를 통과시키거나 인쇄물이 센서를 지나가게 한다. 스캐너는 이미지를 인치당 수백 개의 픽셀로 나누고 각 박스를 (박스가 채워져 있다면) 1 또는 (박스가 비어 있다면) 0으로 표현하면서 처리한다.
  - 점으로 된 결과 행렬을 비트맵(bit map)이라 한다. 컴퓨터는 비트맵을 저장하고, 화면에 표시, 출력할 수 있으나 텍스트를 편집하려고 워드프로세서를 사용할 수는 없다. 러려면 점들의 패턴을 컴퓨터에서 문자로 인식해야 한다. 이는 신경망이 해야 하는 작업이다.
  - 인쇄된 문자를 인식하도록 다층 순방향 신경망을 응용해보자. 숫자 0~9를 인식하는 것으로 일을 제한한다. [그림 9-19]에서 볼 수 있듯이 각 숫자를 5 × 9 비트맵으로 표현한다. 해상도가 더 좋아야 하는 상업적 응용 프로그램에서는 최하 16 × 16 비트맵을 사용한다

04_인공 신경망은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 4: 문자 인식 신경망

- **광학 문자 인식**



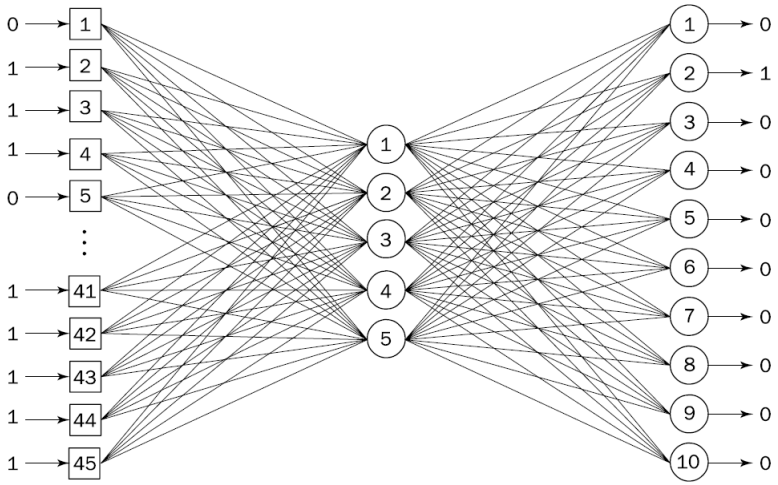
[그림 9-19] 숫자 인식을 위한 비트맵

04_인공 신경망은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 4: 문자 인식 신경망

- 문자 인식을 위한 신경망의 구조
  - 신경망의 구조와 크기는 문제의 복잡도에 따라 달라진다. 예를 들어, 손으로 쓴 문자를 인식하려면 은닉층 3~4개와 수백 개의 뉴런을 포함하는 무척 복잡한 다층 신경망이 있어야 한다.
  - 인쇄된 숫자 인식 문제에 대해서는 은닉층이 1개인 3층 신경망으로도 충분히 정확한 결과를 얻을 수 있을 것이다.

- 최적의 은닉 뉴런 개수
  - [그림 9-20]은 문자 인식 문제를 위한(은닉층에 5개의 뉴런이 있는) 신경망 구조를 보여준다. 은닉층과 출력층에 있는 뉴런은 시그모이드 활성화 함수를 사용한다. 신경망은 모멘텀이 있는 역전파 알고리즘으로 학습된다. 여기에서 모멘텀 상수는 0.95로 정했다.



[그림 9-20] 출력된 숫자를 인식하기 위한 신경망

❖ 사례 연구 4: 문자 인식 신경망

- **최적의 은닉 뉴런 개수**
  - **[표9-2]는 입출력 훈련 패턴을 보여준다. 각 숫자의 비트맵을 표현한 이진 입력 벡터는 신경망에 직접 입력된다.**

[표 9-2] 숫자 인식 신경망을 위한 입력과 바람직한 출력 패턴

숫자	입력 패턴 픽셀 행렬에서의 행									바람직한 출력패턴
	1	2	3	4	5	6	7	8	9	
1	00100	01100	10100	00100	00100	00100	00100	00100	00100	1000000000
2	01110	10001	00001	00001	00010	00100	01000	10000	11111	0100000000
3	01110	10001	00001	00001	00010	00001	00001	10001	01110	0010000000
4	00010	00110	00110	01010	01010	10010	11111	00010	00010	0001000000
5	11111	10000	10000	11110	10001	00001	00001	10001	01110	0000100000
6	01110	10001	10000	10000	11110	10001	10001	10001	01110	0000010000
7	11111	00001	00010	00010	00100	00100	01000	01000	01000	0000001000
8	01110	10001	10001	10001	01110	10001	10001	10001	01110	0000000100
9	01110	10001	10001	10001	01111	00001	00001	10001	01110	0000000010
0	01110	10001	10001	10001	10001	10001	10001	10001	01110	0000000001

05_유전 알고리즘은 어떤 문제를 해결할 수 있을까?

❖ 유전 알고리즘의 문제 해결

- 유전 알고리즘의 특징
  - 유전 알고리즘은 여러 가지 최적화 문제에 적용할 수 있다(Haupt와Haupt,1998).
  - 최적화란 본질적으로 어떤 문제에 대해 좀 더 만족스러운 해를 찾아내는 과정이다. 이는 문제의 해가 하나 이상이고, 그 품질이 모두 다르다는 뜻이다.
  - 유전 알고리즘은 경쟁하는 후보 해집단을 생성하고, 자연 도태 과정을 통해 해집단을 진화하게 한다. 품질이 나쁜 해는 멸종하기 쉬운 반면, 품질이 좋은 해는 살아남아 재생산된다. 유전 알고리즘은 이 과정을 몇 번이고 반복함으로써 최적 해를 번식시킨다.

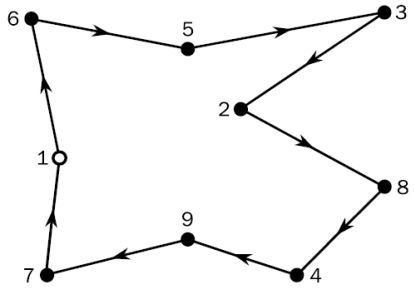
05_유전 알고리즘은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 7: 순회 판매원 문제

- 최적의 여행 계획을 세울 수 있는 지능 시스템을 개발하고 싶다. 자동차로 서유럽과 중부유럽에 있는 주요 도시를 방문한 후 집으로 돌아오려고 한다. 이 문제에 유전 알고리즘을 적용할 수 있을까?

  - 이 문제는 순회 판매원 문제(TSP, Travelling Salesman Problem)로 잘 알려져 있다.
  - 도시N개가있고, 도시 쌍 사이를 여행하는 비용(또는 거리)을 알고 있을 때, 도시를 각각 정확히 한 번씩 방문하고 시작점으로 돌아오는 가장 경제적인 방법(또는 가장 짧은 경로)을 찾으려고 한다.

- 유전알고리즘으로 TSP 풀기
  - 먼저 판매원이 지나는 길을 어떻게 나타낼지 결정해야 한다. 길을 표시하는 가장 자연스러운 방법은 경로 표현이다.
  - 도시에는 문자나 숫자로 된 이름이 붙어 있고, 각 도시를 지나는 경로는 염색체로 표현되며, 적절한 유전 연산자를 사용하여 새로운 경로를 만든다.



[그림 9-28] 판매원의 경로 예시

❖ 사례 연구 7: 순회 판매원 문제

- 유전알고리즘으로 TSP 풀기
  - 1~9 사이의 숫자로 나타내는 도시가 9개 있다고 하자. 염색체에서 정수의 순서는 판매원이 방문하는 도시의 순서를 나타낸다. 이를테면, 다음 염색체는 [그림9-28]과 같은 경로를 나타낸다

1	6	5	3	2	8	4	9	7

  - 판매원은 도시1에서 출발하여 다른 모든 도시를 한 번씩 방문하고 시작점으로 돌아온다.

- TSP에서의 교차 연산자 작동
  - 교차 연산자는 고전적인 형태로는 TSP에 직접 적용하지 못 한다. 단순히 부모 둘의 일부를 교환하기만 하면 중복이나 누락이 포함된 잘못된 경로가 나올 수 있다. 즉 같은 도시를 두 번 방문하거나 한번도 방문하지 않을 때가 발생한다. 예를 들어, 두 부모 염색체의 일부를 다음과 같이 교환해 보자.

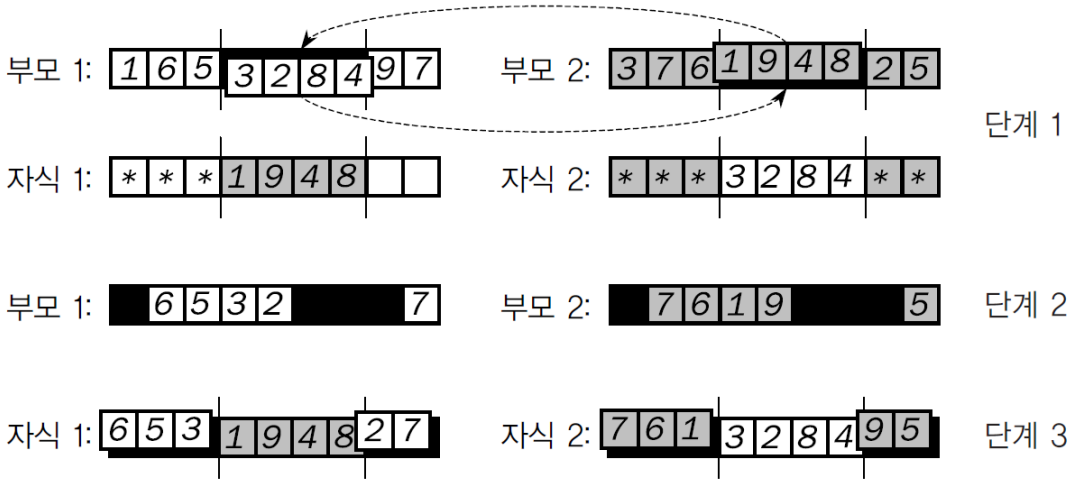
부모 1: | 1 | 6 | 5 | 3 | 2 | 8 | 4 | 9 | 7 | 부모 2: | 3 | 7 | 6 | 1 | 9 | 4 | 8 | 2 | 5 |

  - 도시 5를 두 번 방문하고 도시 7을 빠뜨리는 경로가 만들어진다. 그리고 다른 경로는 도시 7을 두 번 방문하고, 도시5를 빠뜨린다.

자식 1: | 1 | 6 | 5 | 3 | 9 | 4 | 8 | 2 | 5 | 자식 2: | 3 | 7 | 6 | 1 | 2 | 8 | 4 | 9 | 7 |

❖ 사례 연구 7: 순회 판매원 문제

- 유전알고리즘으로 TSP 풀기
  - 연산자는 대체로 한쪽 부모에서 경로의 일부를 고르고, 다른쪽 부모에서는 도시의 순서를 보존하여 자식을 만드는 방법에 바탕을 둔다. [그림 9-29]는 교차 연산자가 작동하는 방법을 보여준다.



[그림 9-29] TSP를 위한 교차 연산자

  - 먼저 부모 각각의 염색체 문자열에서 임의로 두 교차점(문자 ' | '로 표시함)을 균등하게 고른다. 교차점 사이의 유전 정보는 교환 구역을 정의한다. 부모 간에 교환 구역을 교환함으로써 자식 둘의 염색체가 만들어진다. [그림 9-29]에서 별표(*)는 아직 정해지지 않은 도시를 나타낸다.

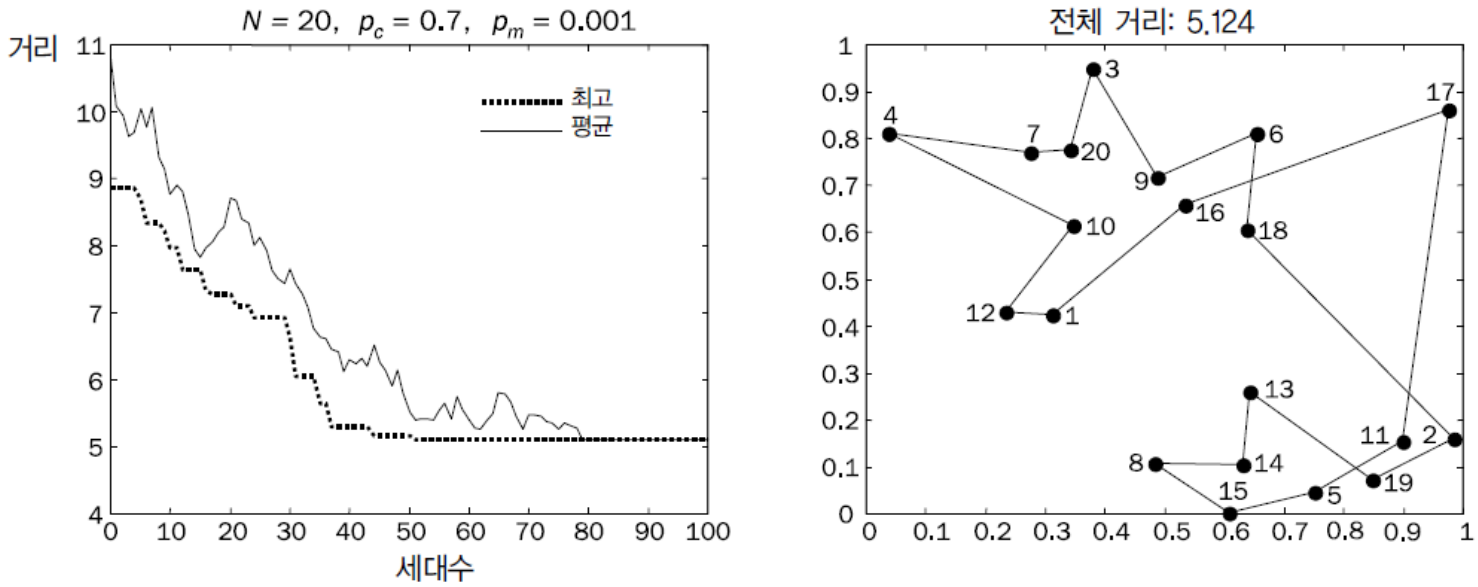
❖ 사례 연구 7: 순회 판매원 문제

- 유전알고리즘으로 TSP 풀기
  - 그 다음으로 각 부모에 있던 원래 도시를 다른 부모의 교환 구역에 있는 것은 제외하고 원래 순서대로 놓는다. 이를테면, 첫째 부모는 도시 3, 2, 8, 4를 둘째 부모의 교환 구역에 있는 도시 1, 9, 4, 8과 교환한다. 그리고 남은 도시들은 원래 순서를 보존하면서 자식으로 들어간다. 그 결과 자식은 부모 각각에 따라 부분적으로 정해진 경로를 나타낸다..

- TSP에서 적합도 함수 정의
  - TSP에 대한 유전 연산자를 만드는 것은 간단한 문제가 아니지만, 적합도 함수를 설계하는 일은 간단하다. 우리가 할 일은 경로의 전체 길이를 재는 것뿐이다. 개별 염색체의 적합도를 경로 길이의 역수로 결정한다. 다시 말해 경로가 짧을수록 염색체의 적합도가 높다.
  - 적합도 함수를 정의하고 유전 연산자를 만들면 유전 알고리즘을 구현하고 실행할 수 있다. 예를 들어, 1 × 1 정사각형에 놓인 20개 도시에 대한 TSP를 생각해보자. 먼저 해집단의 크기와 실행할 세대수를 정한다.
  - 다소 작은 해집단에서 시작하여 몇 세대 뒤에 구한 해를 조사한다. [그림 9-31]은 100세대 뒤에 염색체 20개로 만든 가장 좋은 경로를 보여준다. 여기서 볼 수 있듯, 이 경로는 최적이 아니며 분명히 개선될 수 있다.

❖ 사례 연구 7: 순회 판매원 문제

- TSP에서 적합도 함수 정의

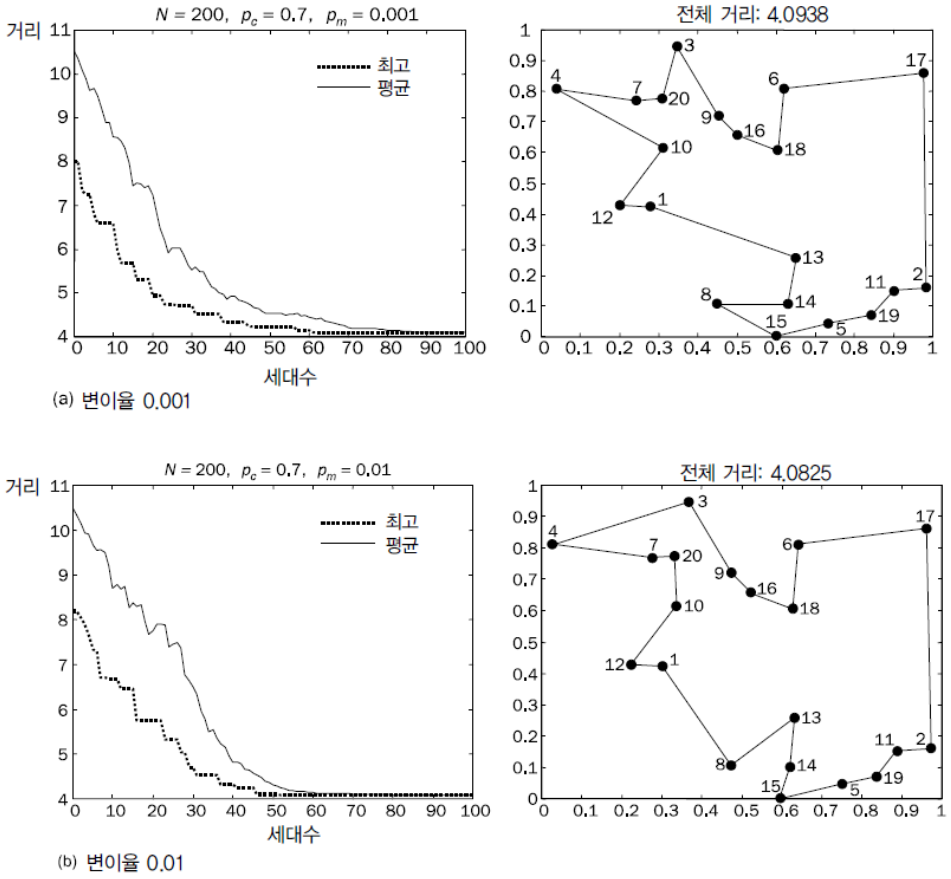


[그림 9-31] 100세대 후 염색체 20개로 이루어진 해집단의 성능 그래프와 여기서 만들어진 가장 좋은 판매원의 경로

- 해집단의 크기를 늘리고 *GA*를 다시 실행해보자. [그림9-32]의 (a)를 보면 경로의 전체 길이가 무려20% 줄어들었다. 결과가 매우 향상되었음을 볼 수 있다.

❖ 사례 연구 7: 순회 판매원 문제

- TSP에서 적합도 함수 정의
  - 해집단의 크기를 늘리고 GA를 다시 실행해보자. [그림9-32]의 (a)를 보면 경로의 전체 길이가 무려20% 줄어들었다. 결과가 매우 향상되었음을 볼 수 있다.



[그림 9-32] 염색체 200개로 이루어진 해집단의 성능 그래프와 여기서 만들어진 가장 좋은 경로

06_하이브리드 지능 시스템은 어떤 문제를 해결할 수 있을까?

❖ 하이브리드 지능 시스템의 문제 해결

- 하이브리드 지능 시스템의 특징
  - 복잡한 현실 문제를 풀 때는 전문가 시스템, 퍼지 논리, 인공 신경망, 진화 연산의 장점을 결합한 복합 지능 시스템을 이용해야 한다.
  - 여러 장점을 결합한 하이브리드 지능 시스템은 특정 분야에 대한 인간적인 전문 지식과 급변하는 환경에서 학습하고 적응하는 능력을 통합할 수 있다.

- 하이브리드 지능 시스템의 장점
  - 비록 하이브리드 지능 시스템 분야는 여전히 진화하는 중이고, 아직까지 하이브리드 도구 대부분이 특별히 유효하지는 않지만, 하이브리드 지능 시스템인 뉴로-퍼지 시스템은 이미 성공적인 많은 응용사례가 있으며 고급 기술로 성숙했다.
  - 인공 신경망은 데이터를 통해 학습할 수 있고, 퍼지 논리는 인간의 의사 결정을 모델링하는 능력이 주된 장점이다.

06_하이브리드 지능 시스템은 어떤 문제를 해결할 수 있을까?

❖ 사례 연구 8: 뉴로-퍼지 의사 결정 지원 시스템

- 심장 사진에서 심근 관류를 진단하는 지능 시스템을 개발하고 싶다. 현재 심장 사진과 임상 기록, 의사의 해석까지 갖추고 있다. 이 문제에 하이브리드 시스템을 적용할 수 있을까?

  - 현대 심장 의학에서 진단은 SPECT(Single Proton Emission Computed Tomography) 영상 분석을 기초로 한다. 환자에게 방사성 추적자를 투여하여 SPECT 영상 두 장을 얻는다. 투여하고10~15분 뒤에 부하가 최대가 될 때 한 장찍고(부하기 영상), 여하고 2~5시간 뒤에 한 장 더 찍는다(안정기 영상).
  - 심근에서 방사성 추적자는 근육의 관류에 비례하여 분포한다. 심장병 전문의는 부하기 영상과 안정기영상을 비교함으로써 심장 기능의 이상을 발견할 수 있다.
  - SPECT 영상은 보통256단계 음영 표현이 가능한 고해상도의 2차원 흑백 사진으로 보여준다. 영상에서 밝은 부분은 심근층에서 관류가 잘 되는 부분이고, 어두운 부분은 허혈이 있을 수 있다는 뜻이다.
  - 이 연구에서는 심장 진단 사례 267개를 사용한다. 각 사례에는 SPECT 영상 2개(부하기 영상과안정기 영상)가 덧붙는다. 그리고 영상은 각각 22구역으로 나뉜다. 구역의 밝기는 그 구역 안의 관류 상태를 반영하며, 0~100 사이의 정수로 나타낸다(Kurgan 외, 2001). 따라서 심장 진단 사례는 각각 연속된 특성 44개와 종합 진단(정상 또는 비정상)을 나타내는 특성 1개로 표현된다.
  - 전체 SPECT 데이터 집합은 정상으로 분류된 사례55개(긍정적인 예)와 비정상으로 분류된 사례 212개(부정적인 예)로 구성된다.

06_하이브리드 지능 시스템은 어떤 문제를 해결할 수 있을까?

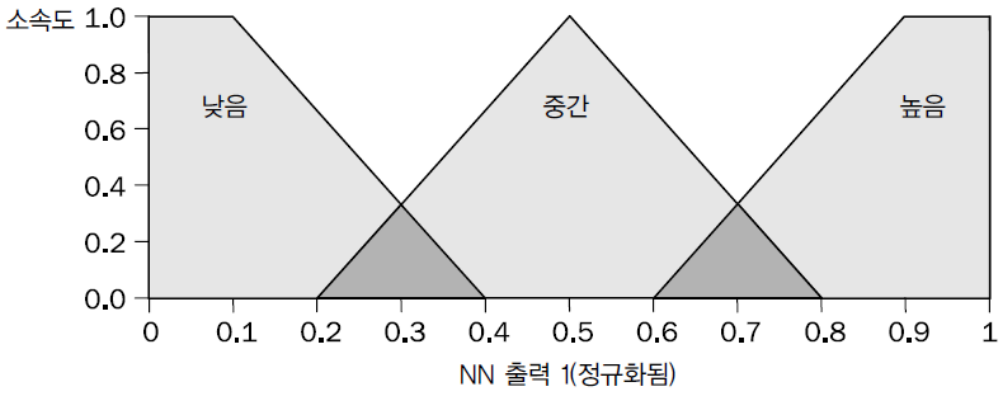
❖ 사례 연구 8: 뉴로-퍼지 의사 결정 지원 시스템

- 이 집합을 훈련 집합과 테스트 집합으로 나눈다. 훈련 집합에는 긍정적인 예 40개와 부정적인 예 40개가 있고, 테스트 집합에는 긍정적인 예 15개와 부정적인 예172개가 있다.

- SPECT 영상을 정상과 비정상으로 분류하기 위한 역전파 신경망의 훈련
  - 실제로 역전파 신경망은 SPECT 영상 분류 문제를 다룰 수 있다. 훈련 집합의 크기는 충분히 큰 것으로 보이고, 신경망은 분류자로서 작동할 수 있다.
  - 입력층에 있는 뉴런의 수는 부하기 영상과 안정기 영상에 있는 구역의 총 수로 결정된다. 이 예에서는 영상이 각각 22구역으로 나뉘므로 입력 뉴런 44개가 필요하다. SPECT 영상은 정상 또는 비정상으로 분류되므로 출력 뉴런을 2개 써야 한다. 실험해 보면 은닉층에 뉴런이 5~7개 정도로 적을 때 만족스럽게 일반화된다. 역전파 신경망은 비교적 빠르게 학습하여 특정한 해에 수렴한다.
  - 인공 신경망의 출력은2개다. 첫 번째 출력은 SPECT 영상이 정상 유형에 속할 가능성, 두 번째는 영상이 비정상 유형에 속할 가능성을 의미한다. 예를 들어, 만약 첫째(정상) 출력이0.92, 둘째(비정상)가 0.16이라면 SPECT 영상은 정상으로 분류되며, 이 경우에는 심장병이 발병할 위험이 낮다고 결론지을 수 있다.

❖ 사례 연구 8: 뉴로-퍼지 의사 결정 지원 시스템

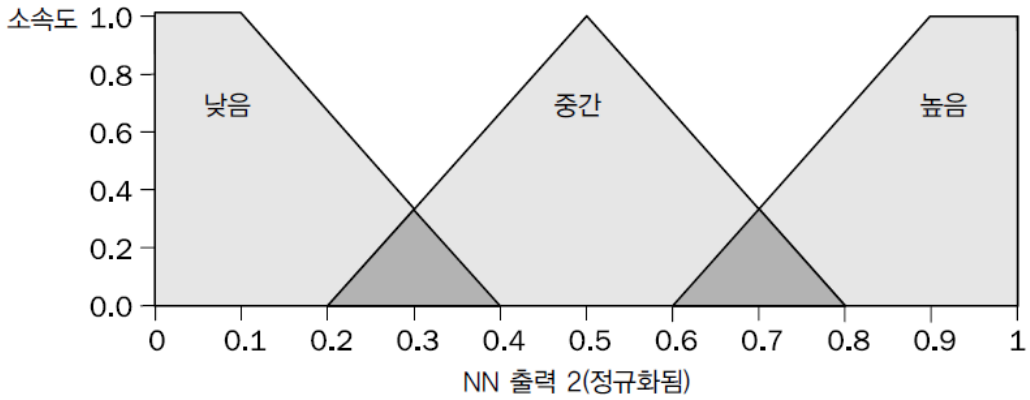
- 퍼지 논리를 이용한 의료 진단
  - 의사들이 SPECT 영상을 분류할 때 정확한 기준이 있는 것은 아니다. 심장병 전문의는 진단 영상에 있는 모든 관류를 조사하고, 부하기 영상과 안정기 영상별로 해당 심근 구역의 밝기도 비교한다. 사실 의사는 심근의 이상을 발견할 때 흔히 경험과 직관에 따른다.
  - 퍼지 논리는 심장병 전문의가 심장병이 발병할 위험을 추정하는 과정을 모델링할 수단을 제공한다
  - [그림 9-33]~[그림 9-35]는 퍼지 시스템에서 사용하는 언어변수의 퍼지집합을 보여준다.



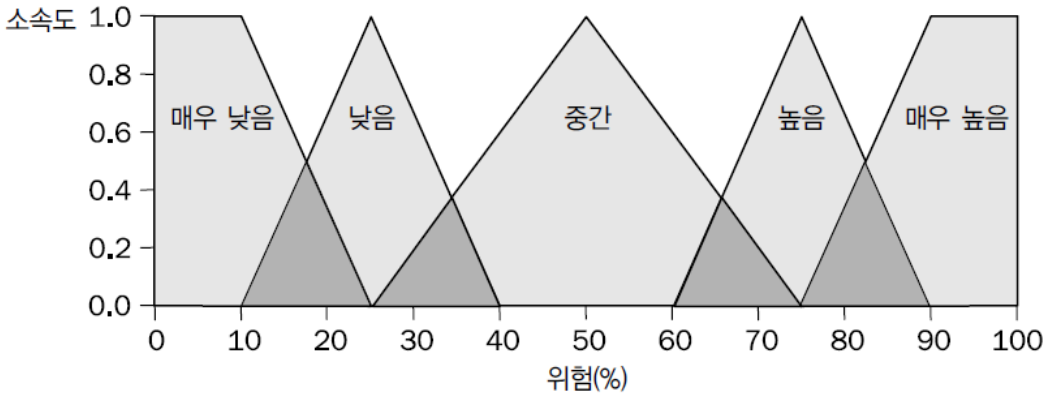
[그림 9-33] 인공 신경망 출력 **정상**에 대한 퍼지 집합

❖ 사례 연구 8: 뉴로-퍼지 의사 결정 지원 시스템

- 퍼지 논리를 이용한 의료 진단
  - [그림 9-33]~[그림 9-35]는 퍼지 시스템에서 사용하는 언어변수의 퍼지집합을 보여준다.



[그림 9-34] 인공 신경망 출력 **비정상**에 대한 퍼지 집합



[그림 9-35] 언어 변수 **위험**에 대한 퍼지 집합

❖ 사례 연구 8: 뉴로-퍼지 의사 결정 지원 시스템

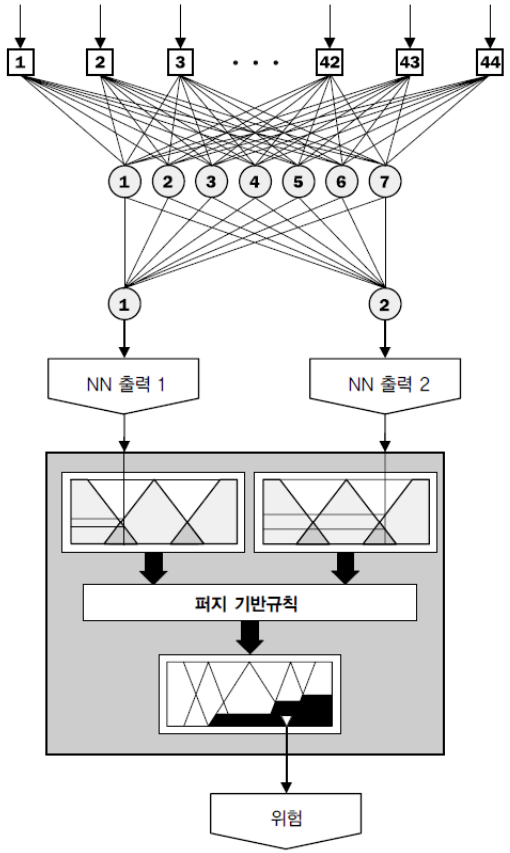
- **퍼지 논리를 이용한 의료 진단**
  - **퍼지규칙** : [그림9-36]

1. If (NN 출력 1이 낮다) and (NN 출력 2가 낮다) then (위험은 중간이다)
2. If (NN 출력 1이 낮다) and (NN 출력 2가 중간이다) then (위험은 높다)
3. If (NN 출력 1이 낮다) and (NN 출력 2가 높다) then (위험은 매우 높다)
4. If (NN 출력 1이 중간이다) and (NN 출력 2가 낮다) then (위험은 낮다)
5. If (NN 출력 1이 중간이다) and (NN 출력 2가 중간이다) then (위험은 중간이다)
6. If (NN 출력 1이 중간이다) and (NN 출력 2가 높다) then (위험은 높다)
7. If (NN 출력 1이 높다) and (NN 출력 2가 낮다) then (위험은 매우 낮다)
8. If (NN 출력 1이 높다) and (NN 출력 2가 중간이다) then (위험은 낮다)
9. If (NN 출력 1이 높다) and (NN 출력 2가 높다) then (위험은 중간이다)

[그림 9-36] 심장병이 발병할 위험을 추정하는 퍼지 규칙

❖ 사례 연구 8: 뉴로-퍼지 의사 결정 지원 시스템

- **퍼지 논리를 이용한 의료 진단**
  - [그림 9-37]은 심장병이 발병할 위험을 추정하는 뉴로-퍼지 의사 결정 지원 시스템의 완전한 구조를 보여준다. 이 시스템을 구축하기 위해 MATLAB Neural Network Toolbox와 MATLAB Fuzzy Logic Toolbox를 사용할 수 있다.



[그림 9-37] 심장병이 발병할 위험을 추정하는 뉴로-퍼지 시스템의 계층 구조

07_데이터 마이닝과 지식 발견

❖ 데이터 마이닝

- 데이터는 모아서 저장해 놓은 것을 말하며, 지식은 정보에 근거한 결정을 내리도록 도와주는 것이다.
- 데이터로부터 지식을 추출하는 것을 데이터 마이닝이라고 한다.
- 데이터 마이닝은 의미 있는 패턴과 규칙을 찾기 위해 엄청난 양의 데이터를 뒤지고 분석하는 것으로 정의할 수도 있다.
- 데이터 마이닝의 궁극의 목적은 지식을 발견하는 것이다.

▪ 데이터 마이닝의 필요성

- 현대에는 데이터의 양이 빠르게 증가한다. 현재 데이터의 양이 매년 두 배씩 증가하고, 종종 엄청난 양의 데이터에서 필요한 정보를 찾느라 어려움을 겪는다.
- 현재 매일 수백 메가바이트의 정보가 인터넷을 통해 돌아다닌다.
- 정보에서 의미 있는 정보와 지식을 추출하는 방법 이 필요한데, 이 방법이 데이터 마이닝이다.

▪ 질의 도구와 데이터 마이닝의 차이점

- 전통적인 질의 도구(query tool)는 가정 기반(assumption-based) 방식이다. 사용자는 올바른 질문만 해야 한다.

07_데이터 마이닝과 지식 발견

- 질의 도구와 데이터 마이닝의 차이점
  - 데이터 집합에 있는 다른 변수 간의 어떤 관계를 가정하는 대신, 데이터 마이닝 도구를 사용해 결과에 영향을 주는 가장 의미 있는 요소를 정할 수 있다.
  - 데이터 마이닝은 어떤 가설도 필요 없으며, 숨겨진 관계와 패턴을 자동으로 발견한다.

- 데이터 마이닝의 실생활 적용
  - 데이터 마이닝은 은행, 금융, 마케팅, 통신 등에서 수없이 응용되고 있다.
  - 특정 상품이나 서비스를 가장 사고자 하는 사람을 정하는 경우, 주식 시장을 모델링하기 위해 시장의 경향을 찾는 경우, 사기 가능성이 높은 보험 청구, 휴대폰 전화, 신용카드 거래를 찾는 경우 등에 데이터 마이닝을 활용한다.

07_요약

- 지식 공학
  - 지식 공학은 지능형 지식기반 시스템을 구축하는 과정이다. 문제 판단, 데이터와 지식 습득, 원형 시스템 개발, 완전한 시스템 개발, 시스템 평가 및 수정, 시스템의 통합 및 유지보수 등 총6단계로 진행된다.

- 지능형 시스템
  - 지능형 시스템은 전형적으로 진단, 선택, 예측, 분류, 군집화, 최적화 및 제어에 사용된다. 문제의 유형, 데이터 및 전문가 의견을 이용할 가능성, 요구된 해법의 형태 및 내용은 지능형 시스템을 구축할 도구를 선택하는 데 영향을 준다.
  - 문제 영역에 관한 이해는 지능형 시스템 구축에서 결정적이다. 원형 시스템의 개발은 문제를 얼마나 잘 이해하고 있는지 검사하는 데 도움이 되고, 문제 풀이 전략, 시스템 구축을 위해 선택된 도구, 습득된 데이터 및 지식의 표현 기법이 현 작업에 적절한지 확인하는 데 도움이 된다.
  - 보통 컴퓨터 프로그램과 달리 지능형 시스템은 '올바른', 그리고 '잘못된' 해를 명확히 정의하기가 어려운 문제를 풀기 위해 설계된다. 따라서 시스템은 보통 사용자가 고른 테스트 케이스로 평가된다.

- 전문가 시스템
  - 진단 문제와 고장 원인을 찾는 문제는 전문가 시스템이 제격이다. 대부분의 진단 문제에는 가능한 해의 수가 한정되어 있고, 잘 형식화된 지식의 양이 제한되어 있다. 또한 인간 전문가가 문제를 푸는 데 시간이 많이 걸리지 않기 때문에 진단 전문가 시스템을 개발하기 쉽다.

07_요약

- 전문가 시스템
  - 실세계 분류 문제를 풀다 보면 종종 부정확하고 불완전한 데이터가 나타난다. 전문가 시스템은 신뢰도가 다른 정보뿐만 아니라 점차 축적되는 증거를 처리함으로써 이런 데이터를 다룰 수 있다.

- 퍼지 시스템
  - 퍼지 시스템은 인간의 의사 결정을 모델링하는 데 적합하다. 중요한 결정을 내릴 때는 보통 자료의 이용 가능성이나 정확성보다 인간의 직관, 상식, 경험에 기반을 둔다. 퍼지 기술은 '연성 기준(soft criteria)'과 '모호한 자료(fuzzy data)'에 대처할 수단을 제공한다.
  - 의사 결정 지원 퍼지 시스템에는 규칙이 수십~수백 가지 포함될 수 있지만, 이들은 비교적 빠르게 개발 되고, 테스트할 수 있다.

- 신경망
  - 신경망은 예측, 분류, 군집화 문제에 성공적으로 적용한 범용 도구 중 하나다. 신경망은 언어 및 문자 인식, 의학 진단, 프로세스 제어 및 로보틱스, 레이더 목표 식별, 환율 예측 및 사기 거래 탐지와 같은 분야에 사용한다. 신경망 응용 분야는 하루가 다르게 확대되고 있다.

07_요약

- 데이터 마이닝
  - 데이터 마이닝은 데이터에서 지식을 이끌어내는 것이다. 또한 의미있는 패턴과 규칙을 발견하기 위해 거대한 규모의 데이터를 탐험하고 분석하는 것으로 정의할 수 있다. 데이터 마이닝의 궁극 적인 목표는 지식 발견이다.
  - 데이터 마이닝은 여전히 새롭고 발전하고 있는 분야지만, 이를 응용한 사례는 상당히 많다. 일대일 마케팅에서는 특정 상품과 서비스를 제일 잘 구매할 것 같은 사람들을 선정하고, 경향 분석에서는 주식 시장을 모델링함으로써 시장에서의 경향성을 찾는다. 사기 탐지에서는 사기일 것 같은 보험 청구, 휴대폰 전화, 신용카드 거래를 찾는 데 데이터 마이닝을 이용한다.
  - 데이터 마이닝에 사용하는 가장 유명한 도구는 트리 구조로 데이터 집합을 설명하는 결정 트리다. 결정 트리는 분류 문제를 푸는 데 특히 적합하다. 데이터 마이닝에 결정 트리 방법을 사용하면 해를 시각화하여 트리를 통해 임의의 경로를 따라가기가 쉽다는 장점이 있다. 깔끔한 규칙을 만들어내는 결정 트리의 능력 때문에 사업 전문가들이 매력을 느낀다.

Thank You !