Lecture 13

Knowledge engineering: Building expert and fuzzy systems

- Introduction, or what is knowledge engineering?
- Will an expert system work for my problem?
- Will a fuzzy expert system work for my problem?
- Summary

What is knowledge engineering?

Davis' law: ***"For every tool there is a task perfectly suited to it"***.

But…

It would be too optimistic to assume that for every task there is a tool perfectly suited to it.

The process of building intelligent knowledge-based systems is called ***knowledge engineering***.

Knowledge engineering has six basic phases:

***Phase* 1:** Problem assessment.

***Phase* 2:** Data and knowledge acquisition.

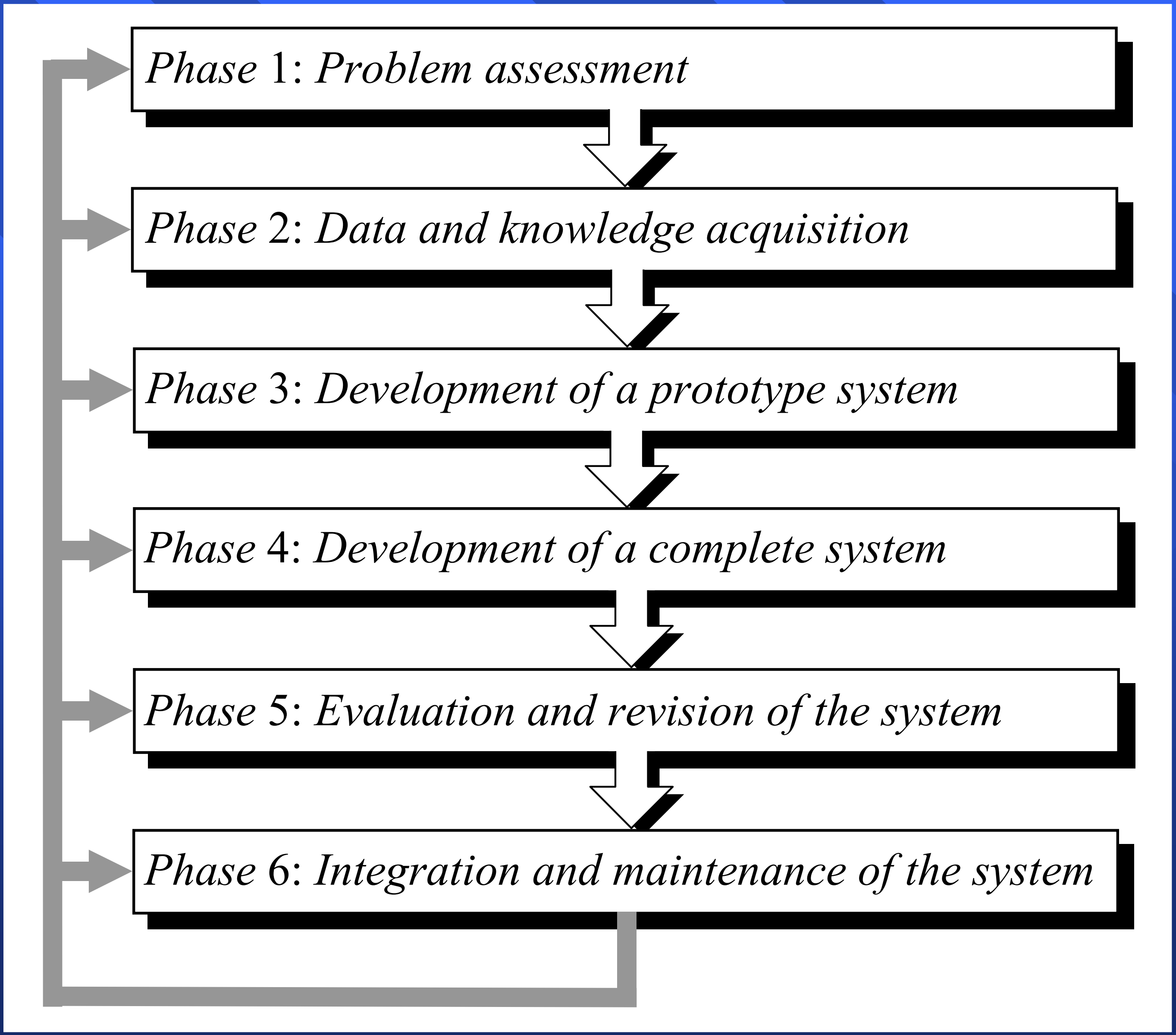
***Phase* 3:** Development of a prototype system.

***Phase* 4:** Development of a complete system.

***Phase* 5:** Evaluation and revision of the system.

***Phase* 6:** Integration and maintenance of the system.

The process of knowledge engineering


Phase 1: Problem assessment
Phase 2: Data and knowledge acquisition
Phase 3: Development of a prototype system
Phase 4: Development of a complete system
Phase 5: Evaluation and revision of the system
Phase 6: Integration and maintenance of the system

Phase 1: Problem assessment

- Determine the problem's characteristics.
- Identify the main participants in the project.
- Specify the project's objectives.
- Determine the resources needed for building the system.

Typical problems addressed by intelligent systems

Problem type	Description
Diagnosis	Inferring malfunctions of an object from its behaviour and recommending solutions.
Selection	Recommending the best option from a list of possible alternatives.
Prediction	Predicting the future behaviour of an object from its behaviour in the past.
Classification	Assigning an object to one of the defined classes.
Clustering	Dividing a heterogeneous group of objects into homogeneous subgroups.
Optimisation	Improving the quality of solutions until an optimal one is found.
Control	Governing the behaviour of an object to meet specified requirements in real-time.

Phase 2: Data and knowledge acquisition

- Collect and analyse data and knowledge.
- Make key concepts of the system design more explicit.

The first issue is ***incompatible*** **data**.

Often the data we want to analyse store text in EBCDIC coding and numbers in packed decimal format, while the tools we want to use for building intelligent systems store text in the ASCII code and numbers as integers with a single- or double-precision floating point.

This issue is normally resolved with data transport tools that automatically produce the code for the required data transformation.

The second issue is ***inconsistent* data**.

Often the same facts are represented differently in different data bases. If these differences are not spotted and resolved in time, we might find ourselves, for example, analysing consumption patterns of carbonated drinks using data that does not include Coca-Cola just because it was stored in a separate database.

The third issue is ***missing* data**.

Actual data records often contain blank fields. Normally we would attempt to infer some useful information from them. In many cases, we can simply fill the blank fields in with the most common or average values. In other cases, the fact that a particular field has not been filled in might itself provide us with very useful information. For example, in a job application form, a blank field for a business phone number might suggest that an applicant is currently unemployed.

How do we approach knowledge acquisition?

- Usually we start with reviewing documents and reading books, papers and manuals related to the problem domain.
- Once we become familiar with the problem, we can collect further knowledge through interviewing the domain expert.
- Then we study and analyse the acquired knowledge, and repeat the entire process again. **Knowledge acquisition is an inherently iterative process.**

Understanding the problem domain is critical for building intelligent system.

A classical example is given by Donald Michie.

A cheese factory had an experienced cheese-tester who was approaching retirement age. The factory manager decided to replace him with an “intelligent machine”. The human tester tested the cheese by sticking his finger into a sample and deciding if it “felt right”. So it was assumed the machine had to do the same – test for the right surface tension. But the machine was useless. Eventually, it turned out that the human tester subconsciously relied on the cheese’s smell rather than on its surface tension and used his finger just to break the crust and let the aroma out.

Phase 3: Development of a prototype system

- Choose a tool for building an intelligent system.
- Transform data and represent knowledge.
- Design and implement a prototype system.
- Test the prototype with test cases.

What is a prototype?

- A prototype system is defined as a small version of the final system.
- It is designed to test how well we understand the problem – to make sure that the problem-solving strategy, the tool selected for building a system, and techniques for representing acquired data and knowledge are adequate to the task.
- It also provides us with an opportunity to persuade the sceptics and, in many cases, to actively engage the domain expert in the system's development.

What is a test case?

- A test case is a problem successfully solved in the past for which input data and an output solution are known.
- During testing, the system is presented with the same input data and its solution is compared with the original solution.

Phase 4: Development of a complete system

- Prepare a detailed design for a full-scale system.
- Collect additional data and knowledge.
- Develop the user interface.
- Implement the complete system.

The main work at this phase is often associated with **adding data and knowledge to the system**.

- If, for example, we develop a diagnostic system, we might need to provide it with more rules for handling specific cases.
- If we develop a prediction system, we might need to collect additional historical examples to make predictions more accurate.

Phase 5: Evaluation and revision of the system

- Evaluate the system against the performance criteria.
- Revise the system as necessary.

- Intelligent systems, unlike conventional computer programs, are designed to solve problems that quite often do not have clearly defined "right" and "wrong" solutions.
- To evaluate an intelligent system is , in fact, to assure that the system performs the intended task to the user's satisfaction.
- A formal evaluation of the system is normally accomplished with the test cases.
- The system's performance is compared against the performance criteria that were agreed upon at the end of the prototyping phase.

<u>Phase 6</u>: Integration and maintenance of the system

- Make arrangements for technology transfer.
- Establish an effective maintenance program.

Will an expert system work for my problem?

The **Phone Call Rule**: "Any problem that can be solved by your in-house expert in a 10-30 minute phone call can be developed as an expert system".

Case study 1
Diagnostic expert system

Diagnostic expert systems are relatively easy to develop:

- Most diagnostic problems have a finite list of possible solutions,
- Involve a rather limited amount of well-formalised knowledge, and
- Often take a human expert a short time (say, an hour) to solve.

Troubleshooting manual for the Macintosh computer

Section 1: System Start-up	
Problem	*Action*
1.1. System does not start	• Check power cords (both ends). • Check the power strip. • Check screen brightness. • Check phonet connectors. • Check keyboard connectors. • Check pins connectors.
1.2. System starts and then freezes	• Restart the Mac. • Unhook all SCSI devices and restart the Mac. • Restart with the shift key down (turns off extensions). • Remove all extensions and add them back one at a time. Restart the Mac after each addition. If using System 7.5 or higher, use Extensions Manager.
1.3. System starts with a Sad Mac	• Start with Disk Tools and do a clean reinstall of the system. • Run Disk First Aid, MacCheck, or Norton's Disk Doctor. • Start with Disk Tools, and if the hard drive icon does not show up, call AV Repair.
1.4. System starts with wrong "music"	• Check cables.

General rule structure

In each rule, we include a clause that identifies the current task:

```
Rule: 1
if     task is ‘system start-up’
then   ask problem

Rule: 2
if     task is ‘system start-up’
and    problem is ‘system does not start’
then   ask ‘test power cords’

Rule: 3
if     task is ‘system start-up’
and    problem is ‘system does not start’
and    ‘test power cords’ is ok
then   ask ‘test the power strip’
```

How do we choose an expert system development tool?

- Tools range from high-level programming languages such as LISP, PROLOG, OPS, C and Java, to expert system shells.
- High-level programming languages offer a greater flexibility, but they require high-level programming skills.
- Shells provide us with the built-in inference engine, explanation facilities and the user interface. We do not need any programming skills to use a shell – we enter rules in English in the shell’s knowledge base.

How do we choose an expert system shell?

When selecting an expert system shell, we consider

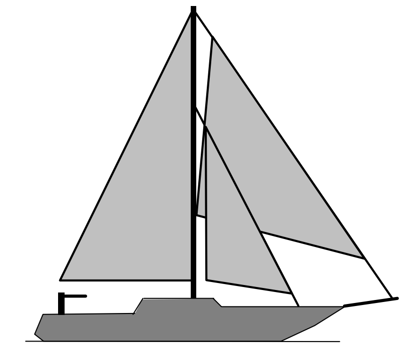
- how the shell represents knowledge (rules or frames);
- what inference mechanism it uses (forward or backward chaining);
- whether the shell supports inexact reasoning and if so what technique it uses (Bayesian reasoning, certainty factors or fuzzy logic);
- whether the shell has an "open" architecture allowing access to external data files and programs;
- how the user will interact with the expert system (graphical user interface, hypertext).

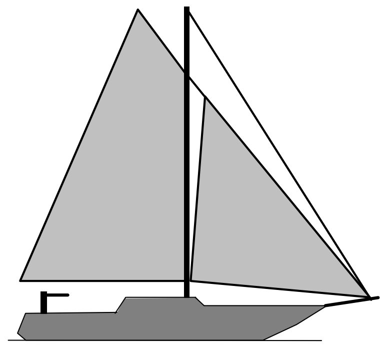
Case study 2
Classification expert system

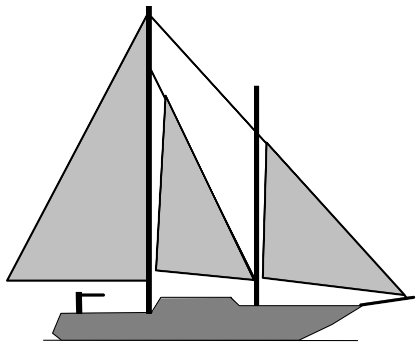
Classification problems can be handled well by both expert systems and neural networks.

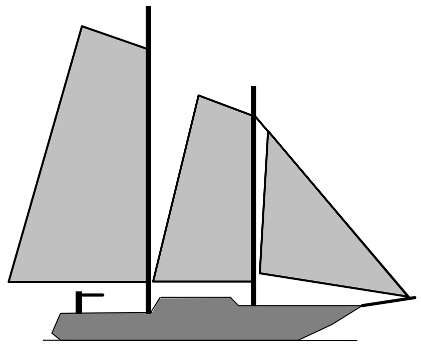
As an example, we will build an expert system to identify different classes of sail boats. We start with collecting some information about mast structures and sail plans of different sailing vessels. Each boat can be uniquely identified by its sail plans.

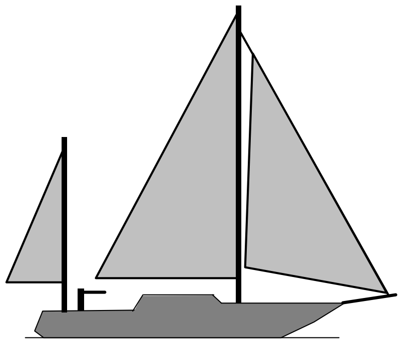
Eight classes of sailing vessels

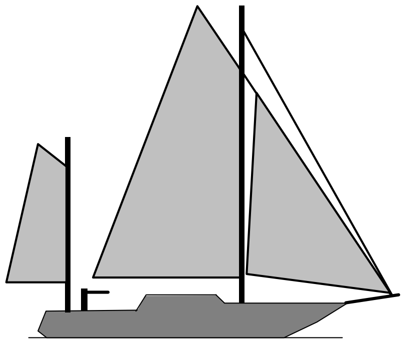

Jib-headed Cutter

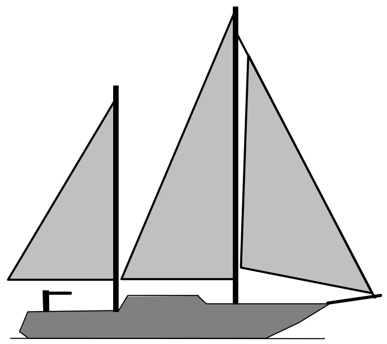

Gaff-headed Sloop

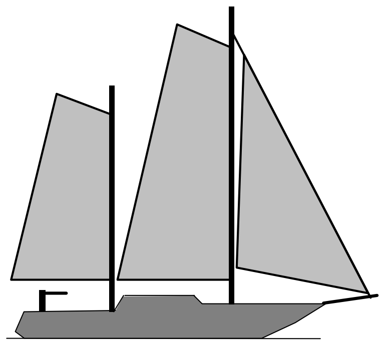

Staysail Schooner


Gaff-headed Schooner


Jib-headed Yawl


Gaff-headed Yawl


Jib-headed Ketch


Gaff-headed Ketch

Rules for the boat classification expert system

```
        /* Sailing Vessel Classification Expert System: Mark 1

Rule: 1 if   'the number of masts' is one
        and  'the shape of the mainsail' is triangular
        then boat is 'Jib-headed Cutter'

Rule: 2 if   'the number of masts' is one
        and  'the shape of the mainsail' is quadrilateral
        then boat is 'Gaff-headed Sloop'

Rule: 3 if   'the number of masts' is two
        and  'the main mast position' is 'forward of the short mast'
        and  'the short mast position' is 'forward of the helm'
        and  'the shape of the mainsail' is triangular
        then boat is 'Jib-headed Ketch'

Rule: 4 if   'the number of masts' is two
        and  'the main mast position' is 'forward of the short mast'
        and  'the short mast position' is 'forward of the helm'
        and  'the shape of the mainsail' is quadrilateral
        then boat is 'Gaff-headed Ketch'
```

Continued

```
Rule: 5  if    'the number of masts' is two
         and   'the main mast position' is 'forward of the short mast'
         and   'the short mast position' is 'aft the helm'
         and   'the shape of the mainsail' is triangular
         then  boat is 'Jib-headed Yawl'

Rule: 6  if    'the number of masts' is two
         and   'the main mast position' is 'forward of the short mast'
         and   'the short mast position' is 'aft the helm'
         and   'the shape of the mainsail' is quadrilateral
         then  boat is 'Gaff-headed Yawl'

Rule: 7  if    'the number of masts' is two
         and   'the main mast position' is 'aft the short mast'
         and   'the shape of the mainsail' is quadrilateral
         then  boat is 'Gaff-headed Schooner'

Rule: 8  if    'the number of masts' is two
         and   'the main mast position' is 'aft the short mast'
         and   'the shape of the mainsail' is 'triangular with two foresails'
         then  boat is 'Staysail Schooner'

         /*****************************************************************************
         /* The SEEK directive sets up the goal
         seek boat
```

Solving classification problems with certainty factors

Although solving real-world classification problems often involves inexact and incomplete data, we still can use the expert system approach.

However, we need to deal with **uncertainties**. The certainty factors theory can manage incrementally acquired evidence, as well as information with different degrees of belief.

Uncertainty management in the boat classification expert system

```
           /* Sailing Vessel Classification Expert System: Mark 2
           control cf
Rule: 1    if     'the number of masts' is one
           then   boat is 'Jib-headed Cutter'              {cf 0.4};
                  boat is 'Gaff-headed Sloop'              {cf 0.4}
Rule: 2    if     'the number of masts' is one
           and    'the shape of the mainsail' is triangular
           then   boat is 'Jib-headed Cutter'              {cf 1.0}
Rule: 3    if     'the number of masts' is one
           and    'the shape of the mainsail' is quadrilateral
           then   boat is 'Gaff-headed Sloop'              {cf 1.0}
Rule: 4    if     'the number of masts' is two
           then   boat is 'Jib-headed Ketch'               {cf 0.1};
                  boat is 'Gaff-headed Ketch'              {cf 0.1};
                  boat is 'Jib-headed Yawl'                {cf 0.1};
                  boat is 'Gaff-headed Yawl'               {cf 0.1};
                  boat is 'Gaff-headed Schooner'           {cf 0.1};
                  boat is 'Staysail Schooner'              {cf 0.1}
```

Continued

Rule: 5	if	‘number of masts’ is two	
	and	‘main mast position’ is ‘forward of the short mast’	
	then	boat is ‘Jib-headed Ketch’	{cf 0.2};
		boat is ‘Gaff-headed Ketch’	{cf 0.2};
		boat is ‘Jib-headed Yawl’	{cf 0.2};
		boat is ‘Gaff-headed Yawl’	{cf 0.2}
Rule: 6	if	‘the number of masts’ is two	
	and	‘the main mast position’ is ‘aft the short mast’	
	then	boat is ‘Gaff-headed Schooner’	{cf 0.4};
		boat is ‘Staysail Schooner’	{cf 0.4}
Rule: 7	if	‘the number of masts’ is two	
	and	‘the short mast position’ is ‘forward of the helm’	
	then	boat is ‘Jib-headed Ketch’	{cf 0.4};
		boat is ‘Gaff-headed Ketch’	{cf 0.4}
Rule: 8	if	‘the number of masts’ is two	
	and	‘the short mast position’ is ‘aft the helm’	
	then	boat is ‘Jib-headed Yawl’	{cf 0.2};
		boat is ‘Gaff-headed Yawl’	{cf 0.2};
		boat is ‘Gaff-headed Schooner’	{cf 0.2};
		boat is ‘Staysail Schooner’	{cf 0.2}

Continued

```
Rule: 9    if     'the number of masts' is two
           and    'the shape of the mainsail' is triangular
           then   boat is 'Jib-headed Ketch'              {cf 0.4};
                  boat is 'Jib-headed Yawl'               {cf 0.4}
Rule: 10   if     'the number of masts' is two
           and    'the shape of the mainsail' is quadrilateral
           then   boat is 'Gaff-headed Ketch'             {cf 0.3};
                  boat is 'Gaff-headed Yawl'              {cf 0.3};
                  boat is 'Gaff-headed Schooner'          {cf 0.3}
Rule: 11   if     'the number of masts' is two
           and    'the shape of the mainsail' is 'triangular with two foresails'
           then   boat is 'Staysail Schooner'             {cf 1.0}
           seek boat
```

Will a fuzzy expert system work for my problem?

If you cannot define a set of exact rules for each possible situation, then use fuzzy logic.

While certainty factors and Bayesian probabilities are concerned with the imprecision associated with the outcome of a well-defined event, fuzzy logic concentrates on the imprecision of the event itself.

Inherently imprecise properties of the problem make it a good candidate for fuzzy technology.

Case study 3
Decision-support fuzzy systems

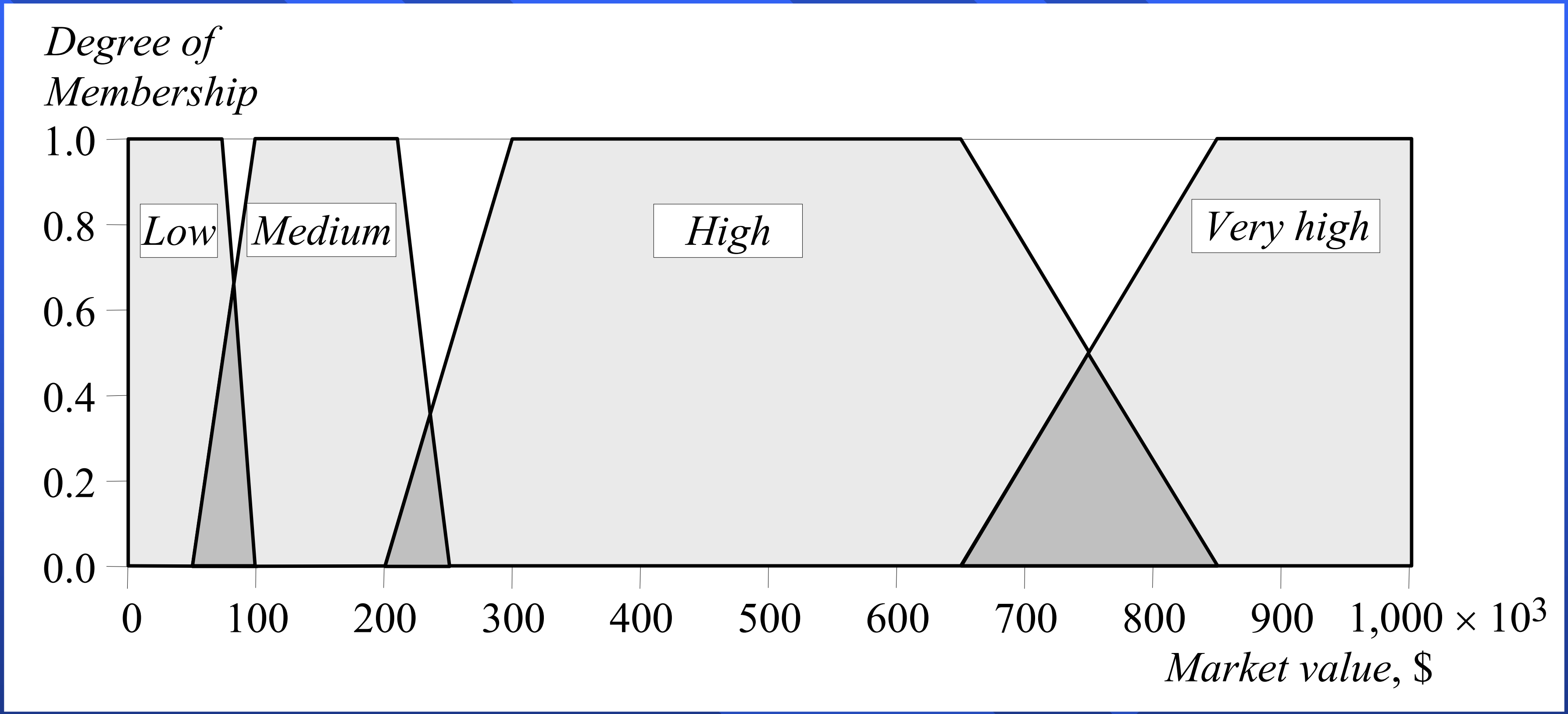
Although, most fuzzy technology applications are still reported in control and engineering, an even larger potential exists in business and finance. Decisions in these areas are often based on human intuition, common sense and experience, rather than on the availability and precision of data.

Fuzzy technology provides us with a means of coping with the "soft criteria" and "fuzzy data" that are often used in business and finance.

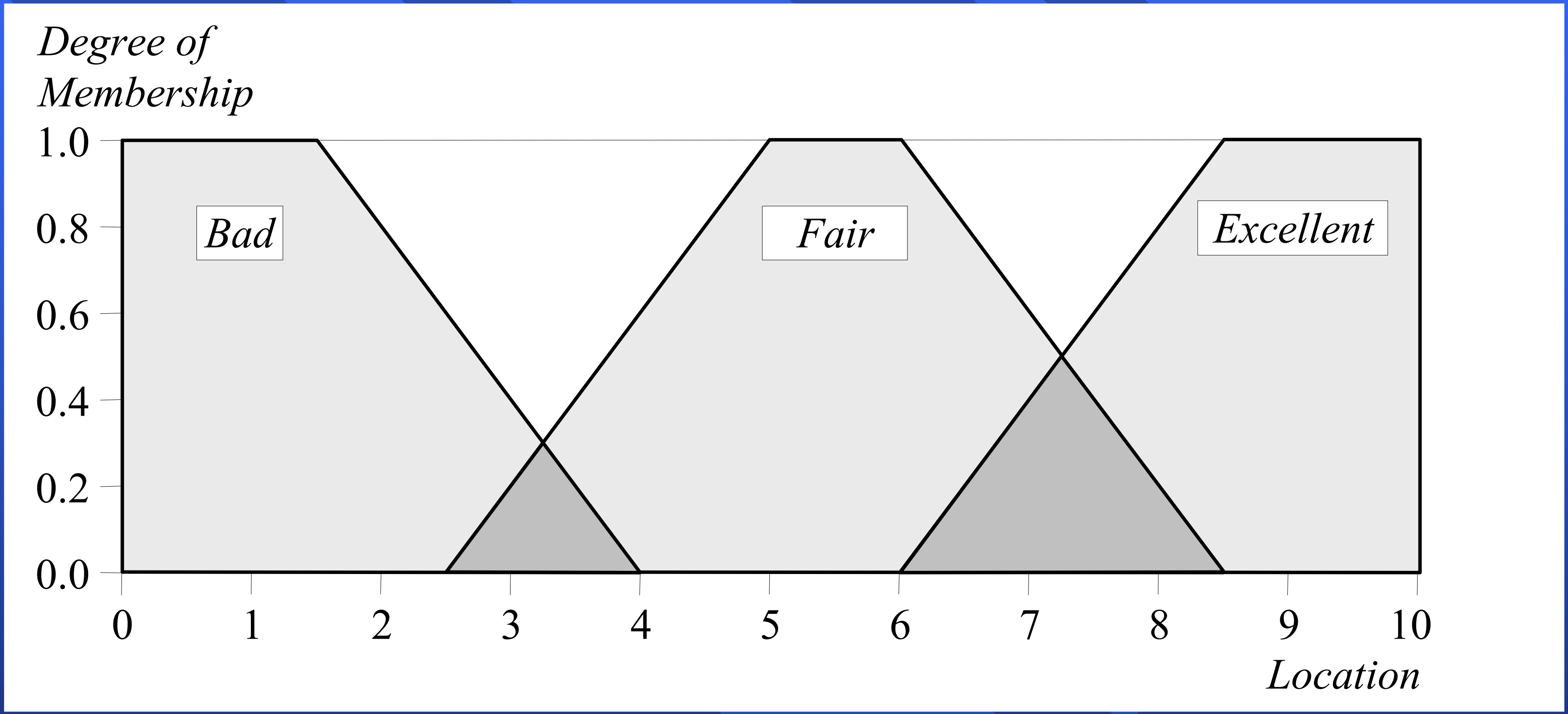
Mortgage application assessment is a typical problem to which decision-support fuzzy systems can be successfully applied.

Assessment of a mortgage application is normally based on evaluating the market value and location of the house, the applicant's assets and income, and the repayment plan, which is decided by the applicant's income and bank's interest charges.

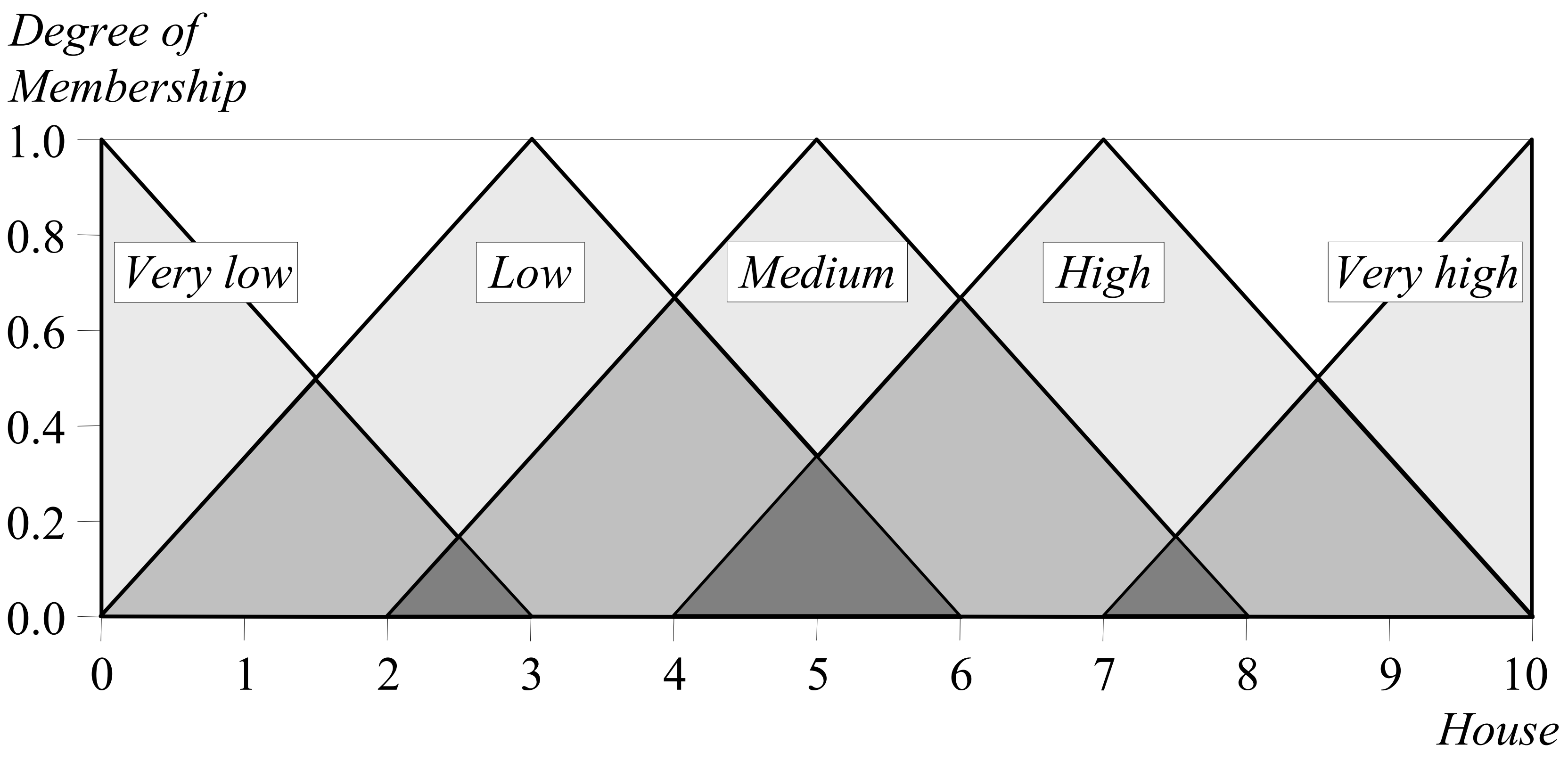
Fuzzy sets of the linguistic variable *Market value*



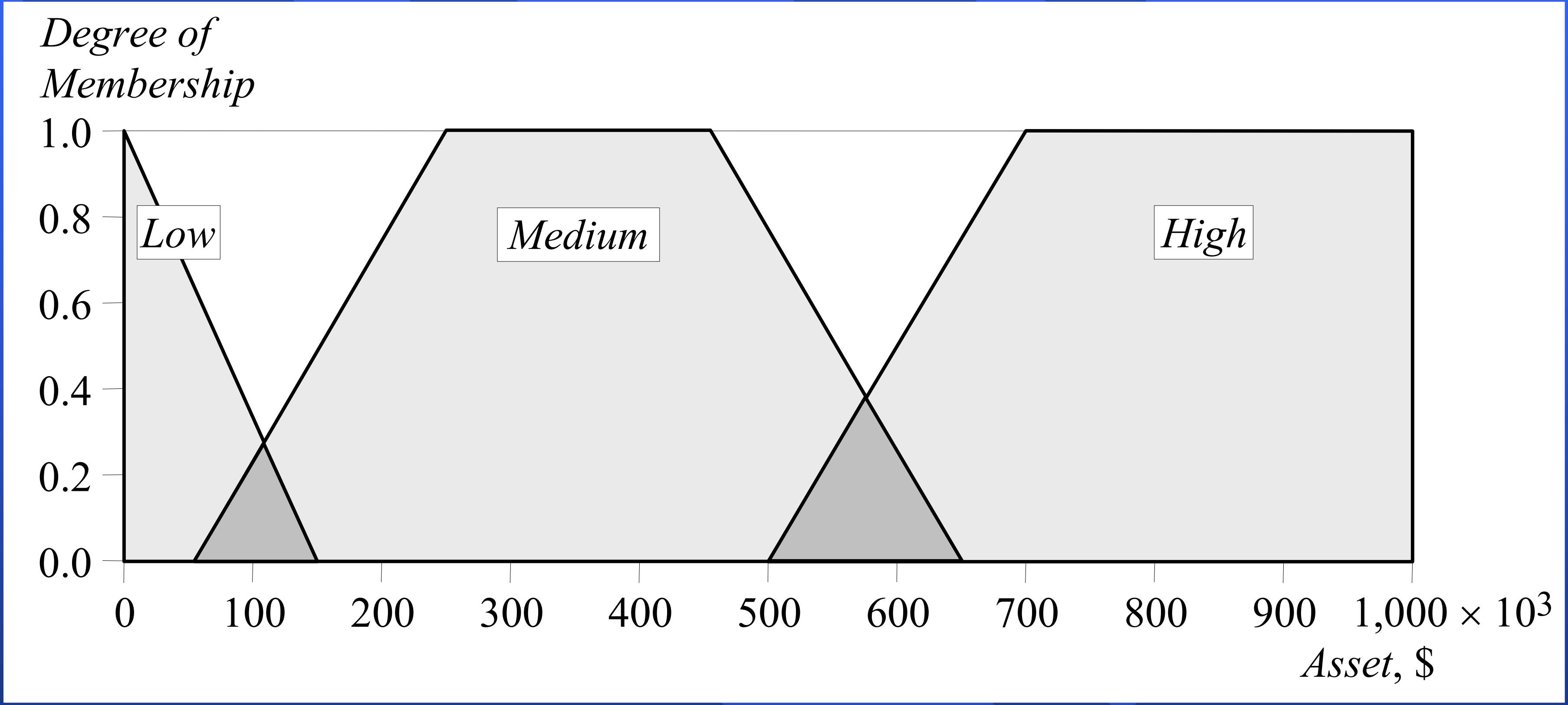
Fuzzy sets of the linguistic variable *Location*



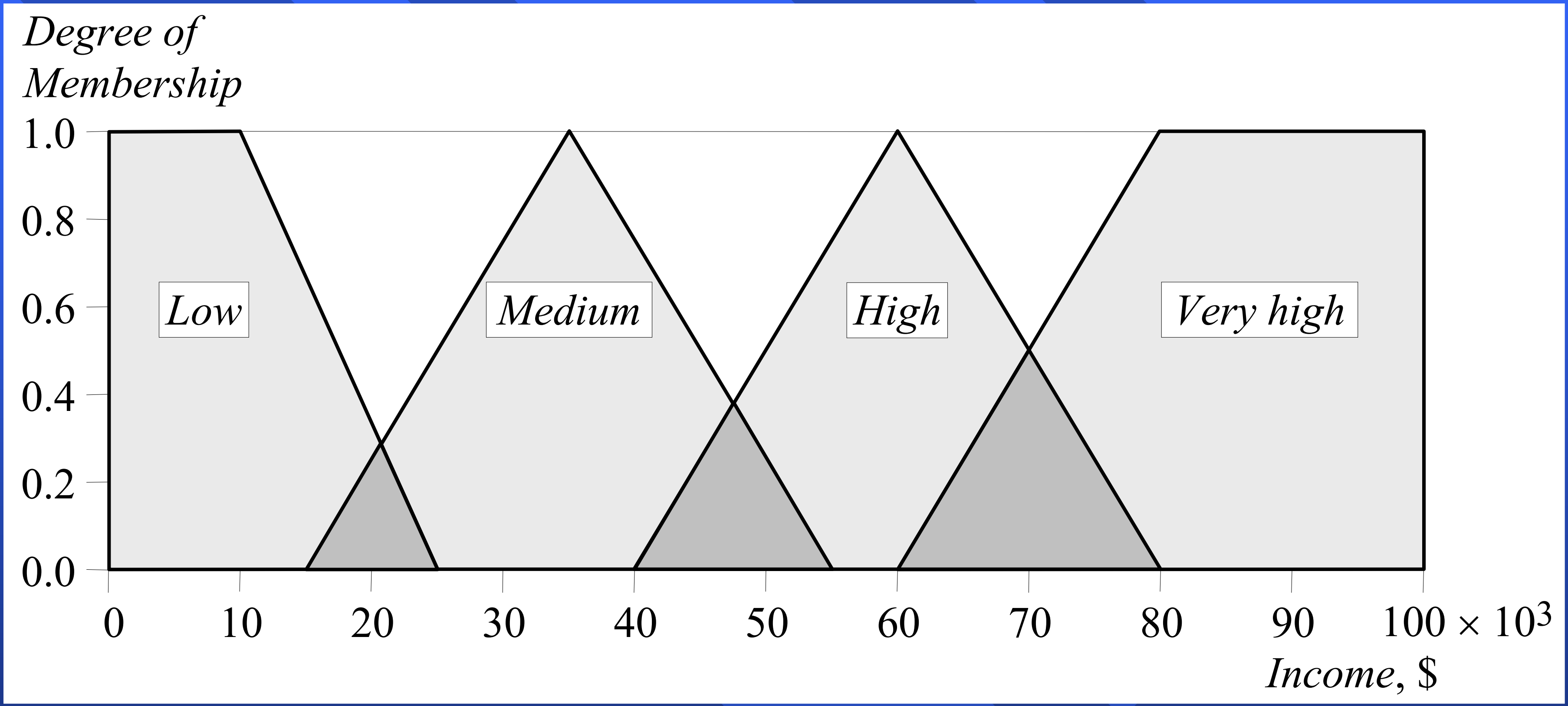
Fuzzy sets of the linguistic variable *House*



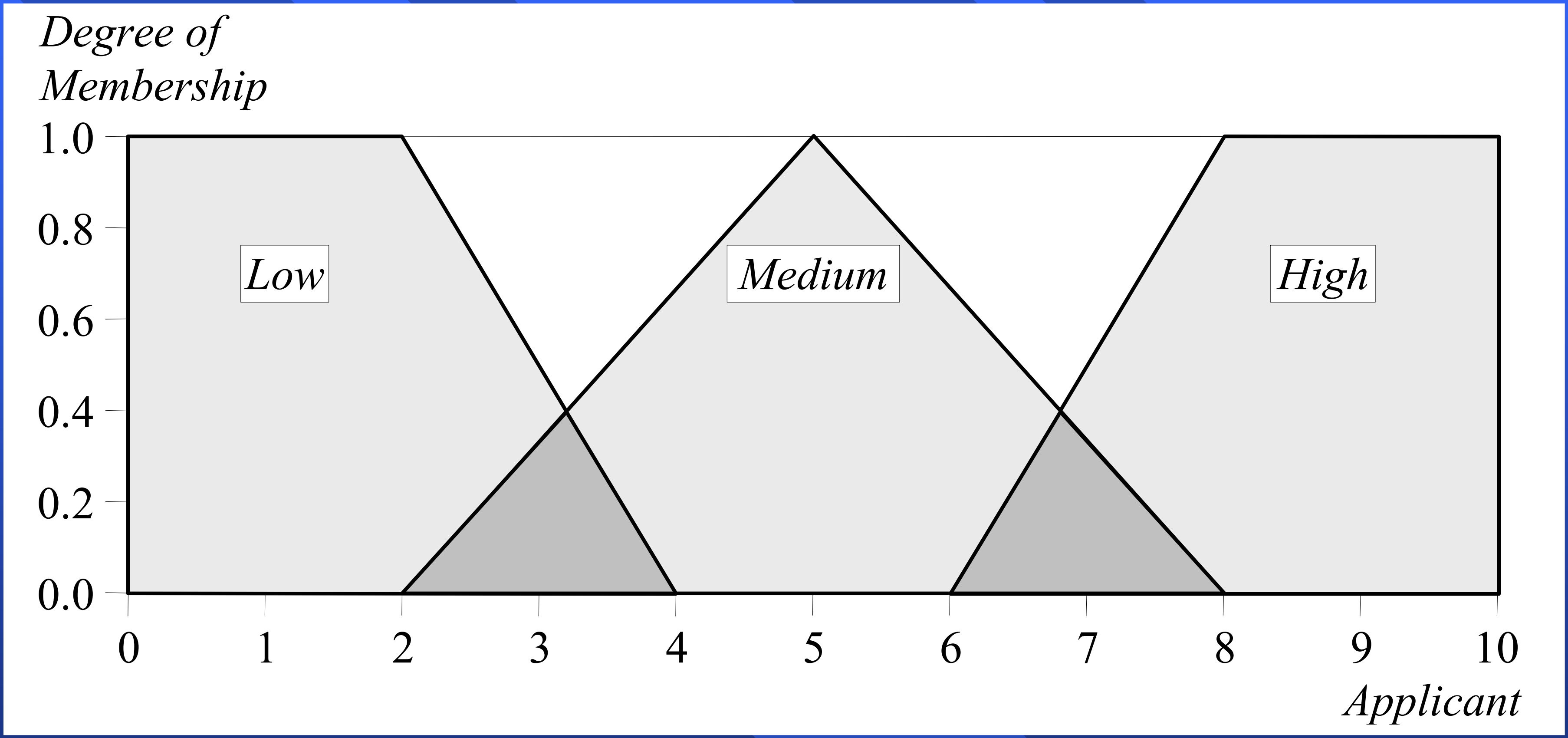
Fuzzy sets of the linguistic variable *Asset*



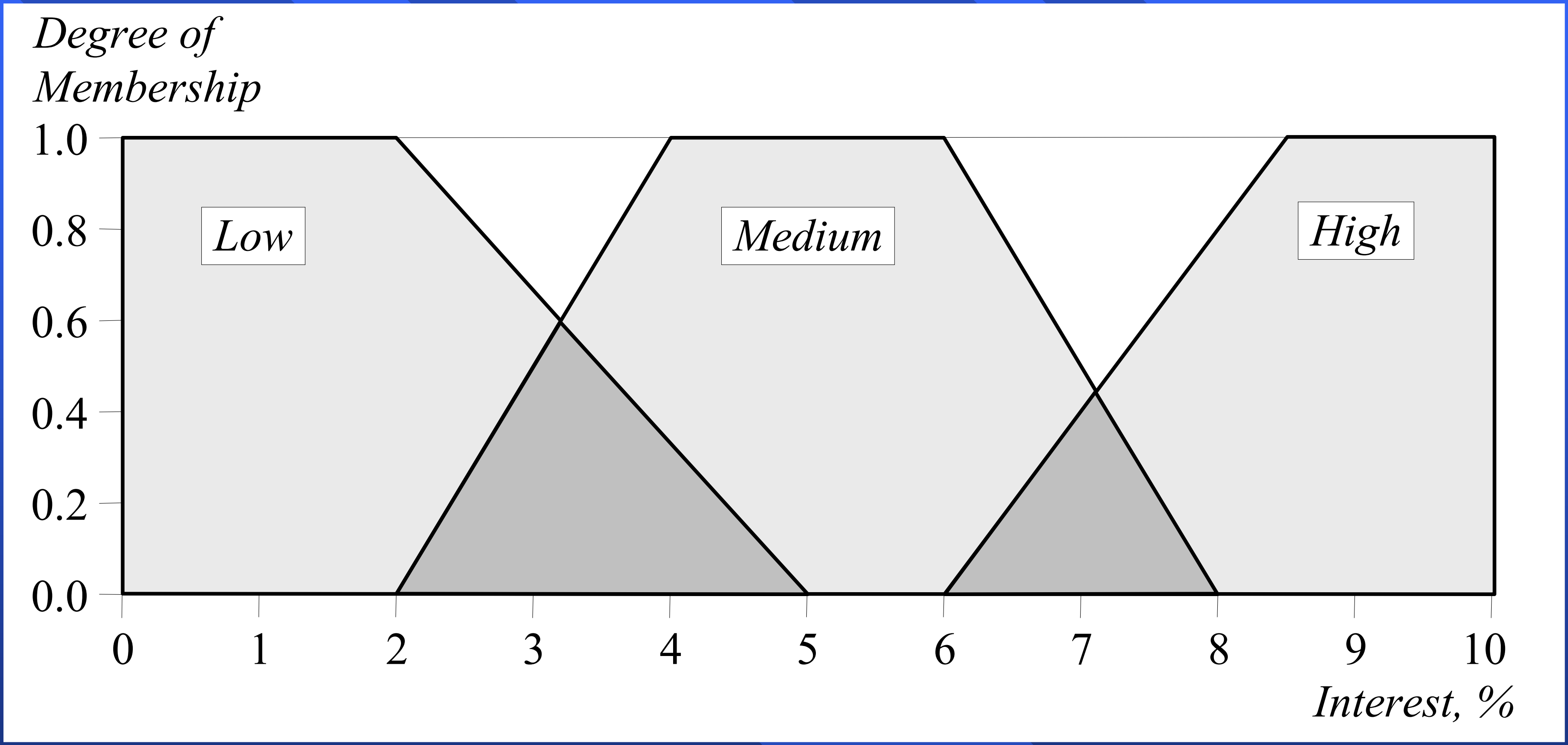
Fuzzy sets of the linguistic variable *Income*



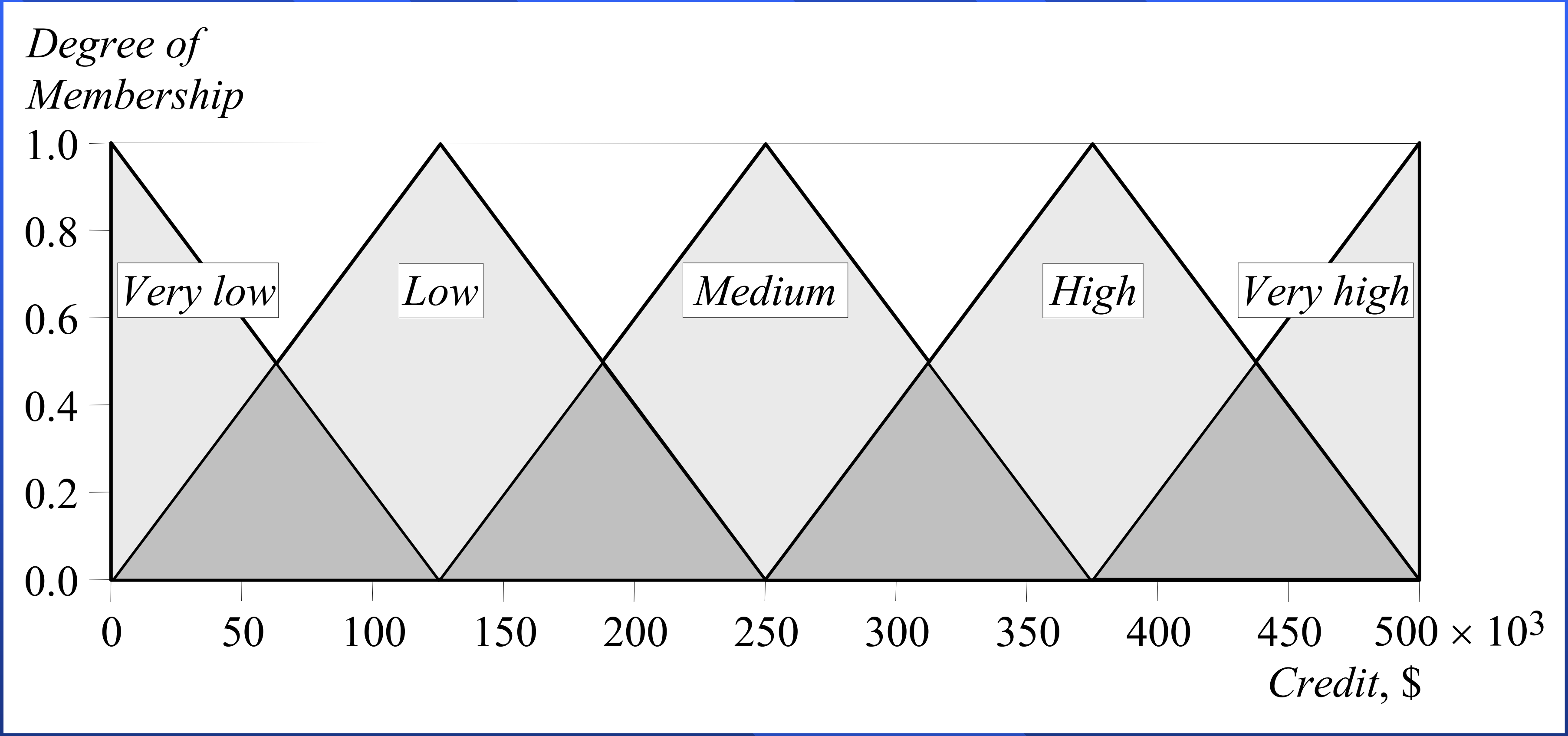
Fuzzy sets of the linguistic variable *Applicant*



Fuzzy sets of the linguistic variable *Interest*



Fuzzy sets of the linguistic variable *Credit*



Rules for mortgage loan assessment

Rule Base 1: Home Evaluation

1. If (Market_value is Low) then (House is Low)
2. If (Location is Bad) then (House is Low)
3. If (Location is Bad) and (Market_value is Low) then (House is Very_low)
4. If (Location is Bad) and (Market_value is Medium) then (House is Low)
5. If (Location is Bad) and (Market_value is High) then (House is Medium)
6. If (Location is Bad) and (Market_value is Very_high) then (House is High)
7. If (Location is Fair) and (Market_value is Low) then (House is Low)
8. If (Location is Fair) and (Market_value is Medium) then (House is Medium)
9. If (Location is Fair) and (Market_value is High) then (House is High)
10. If (Location is Fair) and (Market_value is Very_high) then (House is Very_high)
11. If (Location is Excellent) and (Market_value is Low) then (House is Medium)
12. If (Location is Excellent) and (Market_value is Medium) then (House is High)
13. If (Location is Excellent) and (Market_value is High) then (House is Very_high)
14. If (Location is Excellent) and (Market_value is Very_high) then (House is Very_high)

Rules for mortgage loan assessment

Rule Base 2: Applicant Evaluation

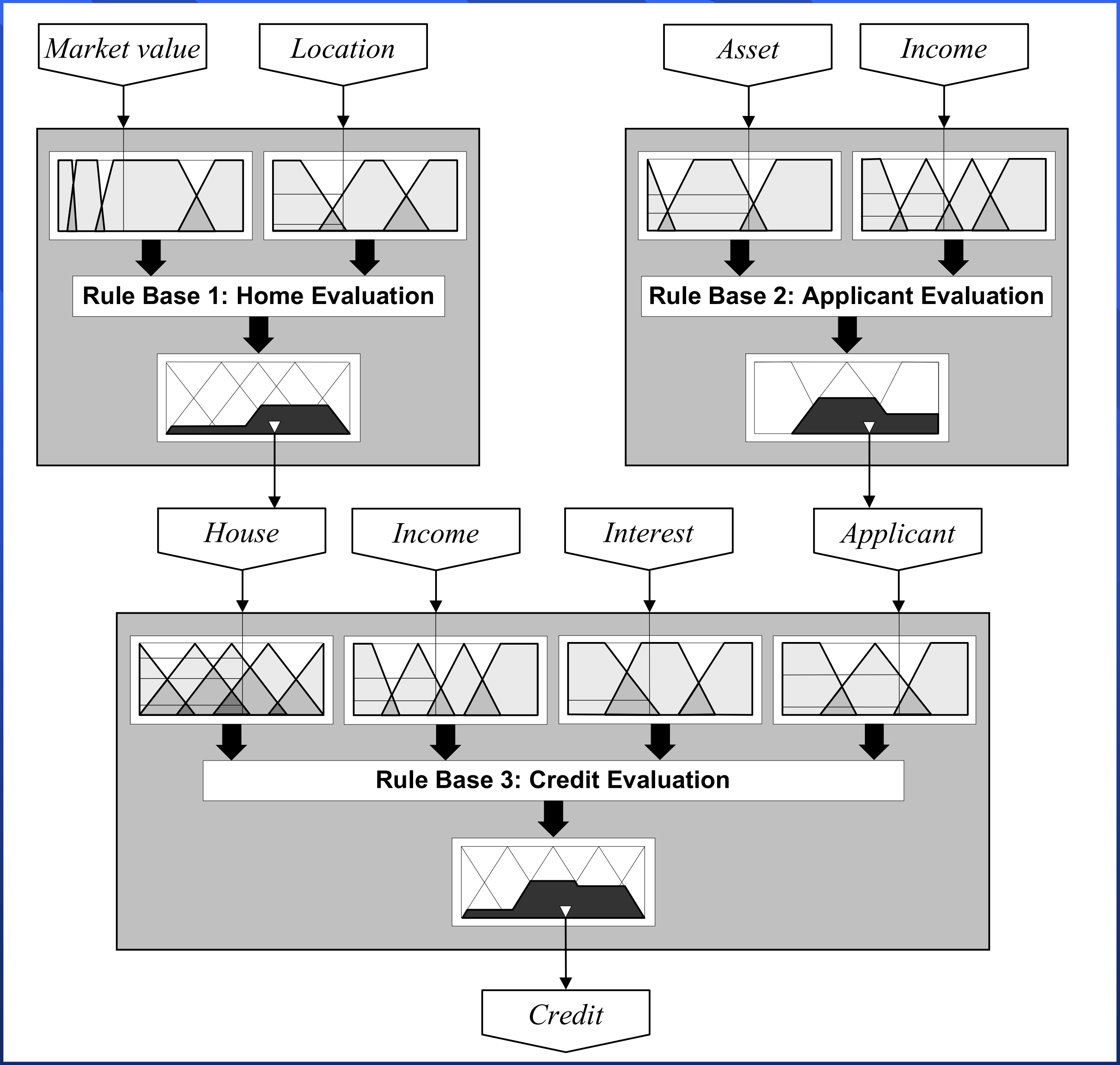
1. If (Asset is Low) and (Income is Low) then (Applicant is Low)
2. If (Asset is Low) and (Income is Medium) then (Applicant is Low)
3. If (Asset is Low) and (Income is High) then (Applicant is Medium)
4. If (Asset is Low) and (Income is Very_high) then (Applicant is High)
5. If (Asset is Medium) and (Income is Low) then (Applicant is Low)
6. If (Asset is Medium) and (Income is Medium) then (Applicant is Medium)
7. If (Asset is Medium) and (Income is High) then (Applicant is High)
8. If (Asset is Medium) and (Income is Very_high) then (Applicant is High)
9. If (Asset is High) and (Income is Low) then (Applicant is Medium)
10. If (Asset is High) and (Income is Medium) then (Applicant is Medium)
11. If (Asset is High) and (Income is High) then (Applicant is High)
12. If (Asset is High) and (Income is Very_high) then (Applicant is High)

Rules for mortgage loan assessment

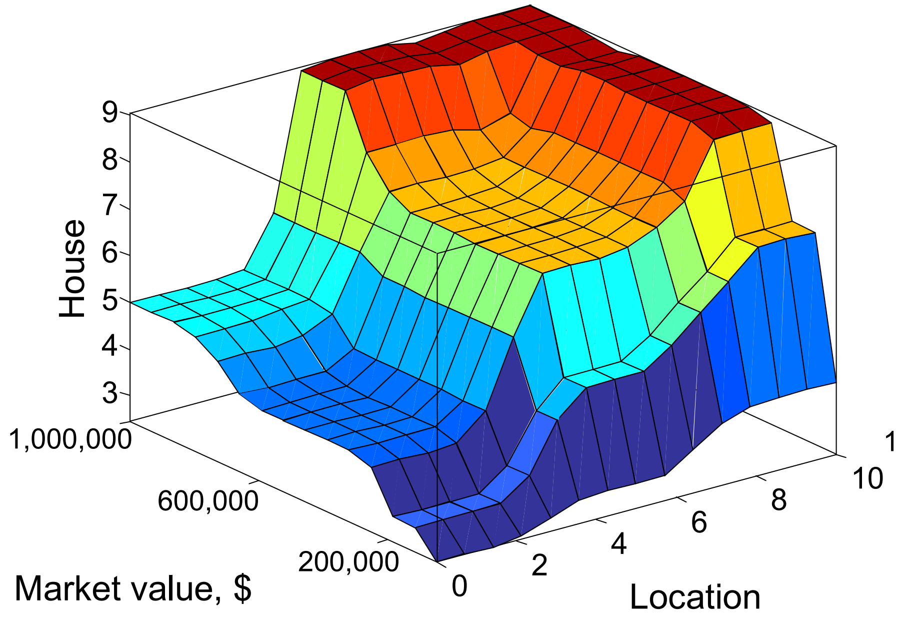
Rule Base 3: Credit Evaluation

1. If (Income is Low) and (Interest is Medium) then (Credit is Very_low)
2. If (Income is Low) and (Interest is High) then (Credit is Very_low)
2. If (Income is Medium) and (Interest is High) then (Credit is Low)
4. If (Applicant is Low) then (Credit is Very_low)
5. If (House is Very_low) then (Credit is Very_low)
6. If (Applicant is Medium) and (House is Very_low) then (Credit is Low)
7. If (Applicant is Medium) and (House is Low) then (Credit is Low)
8. If (Applicant is Medium) and (House is Medium) then (Credit is Medium)
9. If (Applicant is Medium) and (House is High) then (Credit is High)
10. If (Applicant is Medium) and (House is Very_high) then (Credit is High)
11. If (Applicant is High) and (House is Very_low) then (Credit is Low)
12. If (Applicant is High) and (House is Low) then (Credit is Medium)
13. If (Applicant is High) and (House is Medium) then (Credit is High)
14. If (Applicant is High) and (House is High) then (Credit is High)
15. If (Applicant is High) and (House is Very_high) then (Credit is Very_high)

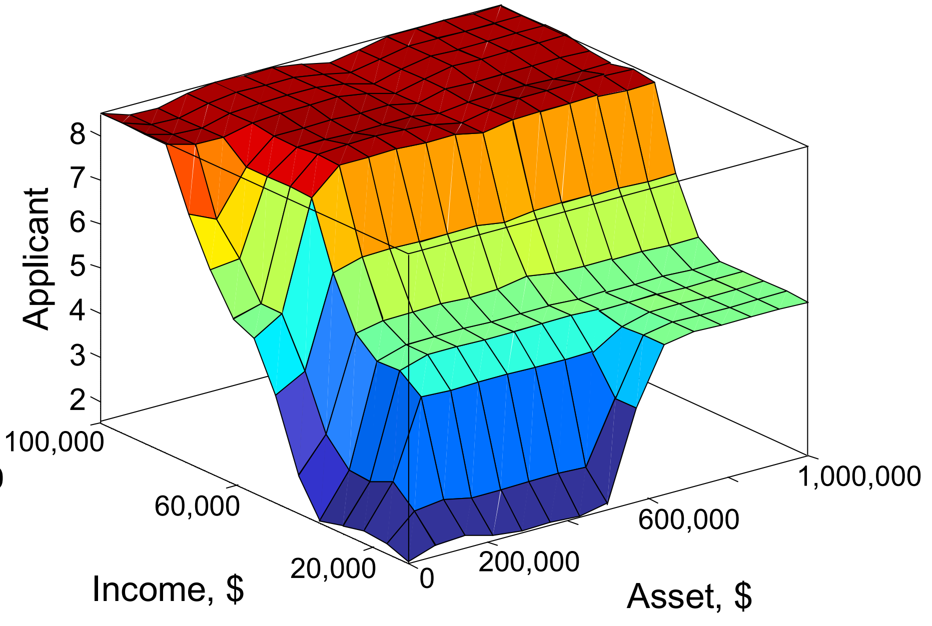
Hierarchical fuzzy model


Market value
Location
Asset
Income
Rule Base 1: Home Evaluation
Rule Base 2: Applicant Evaluation
House
Income
Interest
Applicant
Rule Base 3: Credit Evaluation
Credit

Three-dimensional plots for Rule Base 1 and Rule base 2

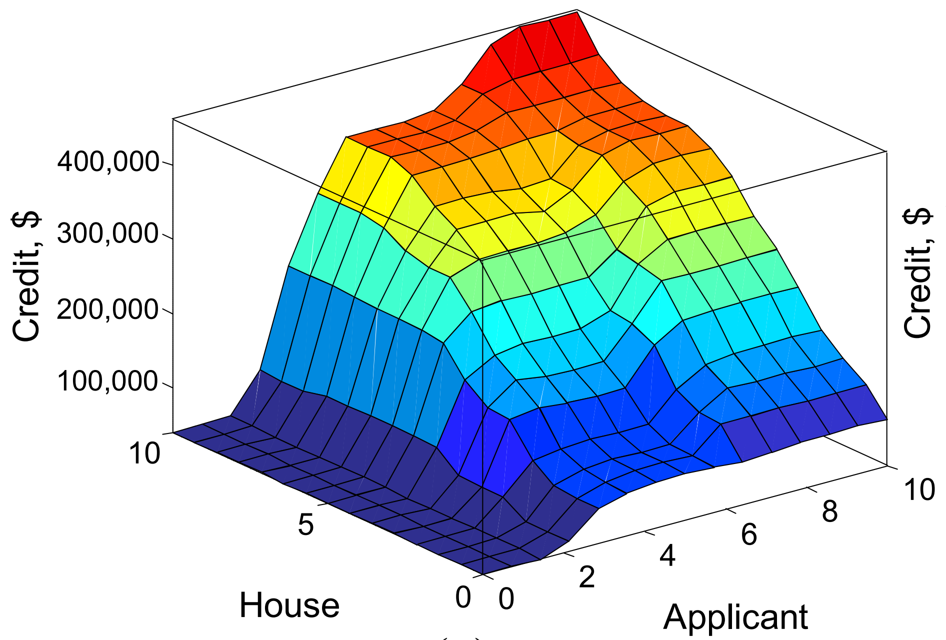


(*a*)

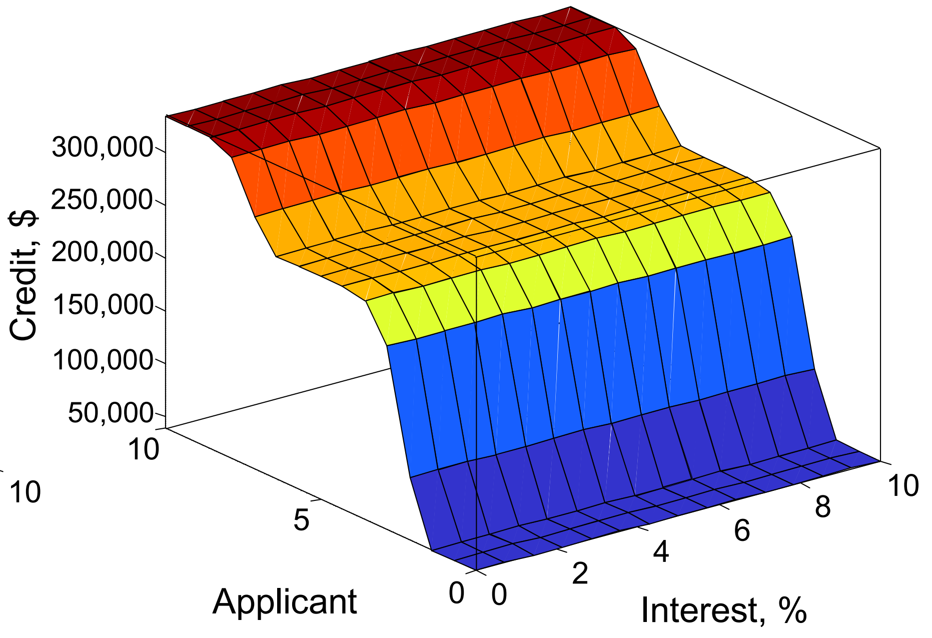


(*b*)

Three-dimensional plots for Rule Base 3



(*a*)



(*b*)