CSCI 323: Artificial Intelligence

Name: eValu8

Group Project

Submitted To: Dr. Farhad Orumchian

Submitted By: Alexander Al Basosi 4563293

Anem Imran Riyaz 4587443

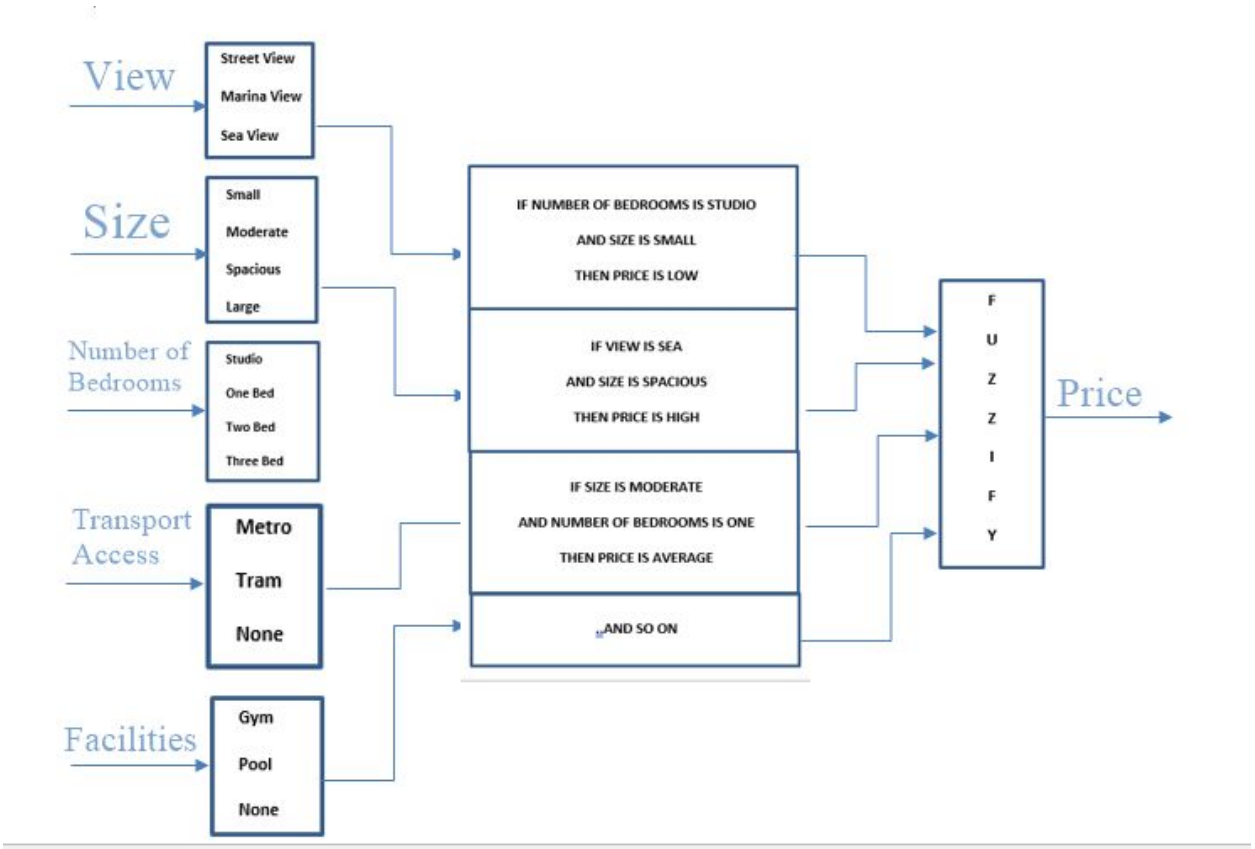
Date Submitted: 08/08/2017

Table Of Contents

What is it?

eValu8 is a C++ based application that evaluates the price of a property based on certain characteristics, such as its view, size in square feet, number of bedrooms, facilities, and access to transportation such as the metro and the tram. It takes in these five inputs, and uses fuzzy logic to classify the inputs into categories, and based on certain pre-determined rules, displays to the user the estimated price of that property. Right now, the application only supports rental apartments in Dubai Marina, but separate fuzzy controllers can be built to incorporate other locations and even apply on a global scale.

A High Level View of the Application



Inputs				
View	**Size (sq. feet)**	**Number of Bedrooms**	**Transportation Access**	**Facilities**
Street View	Small	Studio	None	None
Marina View	Moderate	One Bedroom	Tram	Gym
Sea View	Spacious	Two Bedroom	Metro	Pool
	Large	Three Bedroom		

Based upon the inputs, tables were constructed to determine the memberships values of each of the fuzzy variables.

View	Street View	Marina View	Sea View
0.1	.5	0	0
0.2	1	0	0
0.3	.5	0	0
0.4	0	.5	0
0.5	0	1	0
0.6	0	.5	.0
0.7	0	0	.5
0.8	0	0	1
0.9	0	0	.5

Size	Small	Moderate	Spacious	Large
0	0	0	0	0

250	.5	0	0	0
500	1	0	0	0
750	1	.5	0	0
1000	1	1	0	0
1250	.5	1	.5	0
1500	0	1	1	0
1750	0	.5	1	0
2000	0	0	1	0
2250	0	0	.5	.5
2500	0	0	0	1
2750	0	0	0	1
3000	0	0	0	1

Number of Bedrooms	Studio	One Bedroom	Two Bedroom	Three Bedroom
0.1	.5	0	0	0
0.2	1	0	0	0
0.3	.5	.5	0	0
0.4	0	1	0	0
0.5	0	.5	.5	0
0.6	0	0	1	0
0.7	0	0	.5	.5
0.8	0	0	0	1
0.9	0	0	0	.5

-

Facilities	None	Gym	Pool
0.1	.5	0	0
0.2	1	0	0
0.3	.5	0	0
0.4	0	.5	0
0.5	0	1	0
0.6	0	.5	.0
0.7	0	0	.5
0.8	0	0	1
0.9	0	0	.5

-

Facilities	None	Tram	Metro
0.1	.5	0	0
0.2	1	0	0
0.3	.5	0	0
0.4	0	.5	0
0.5	0	1	0
0.6	0	.5	.0
0.7	0	0	.5
0.8	0	0	1
0.9	0	0	.5

Price Settings	Low	Average	High	Expensive
0	0	0	0	0
25,000	.5	0	0	0
50,000	1	0	0	0
75,000	1	.5	0	0
100,000	1	1	0	0
125,000	.5	1	.5	0
150,000	0	1	1	0
175,000	0	.5	1	.5
200,000	0	0	1	1
225,000	0	0	.5	1
250,000	0	0	0	1

Rules

A set of rules were determined as the training data for the fuzzy controller. Based on this training data, and the tables above, the fuzzy controller calculates the estimated price of the property, after finding the crisp values and defuzzifying them:

Rule 1	IF View is StreetView AND Size is Small AND NumberOfBedrooms is Studio AND Facilities is None AND TransportationAccess is Tram THEN Price is Low
Rule 2	IF View is StreetView AND Size is Small AND NumberOfBedrooms is OneBedroom AND Facilities is Gym AND TransportationAccess is None THEN Price is Low
Rule 3	IF View is StreetView AND Size is Moderate AND NumberOfBedrooms is TwoBedroom AND Facilities is Gym AND TransportationAccess is Metro THEN Price is Average
Rule 4	IF View is StreetView AND Size is Spacious AND NumberOfBedrooms is ThreeBedrooms AND Facilities is Pool AND TransportationAccess is Metro THEN Price is High

Rule 5	IF View is StreetView AND Size is Large AND NumberOfBedrooms is TwoBedroom AND Facilities is None AND TransportationAccess is None THEN Price is Average
Rule 6	IF View is StreetView AND Size is Large AND NumberOfBedrooms is ThreeBedrooms AND Facilities is Gym AND TransportationAccess is None THEN Price is High
Rule 7	IF View is MarinaView AND Size is Small AND NumberOfBedrooms is Studio AND Facilities is Pool AND TransportationAccess is Metro THEN Price is Average
Rule 8	IF View is MarinaView AND Size is Small AND NumberOfBedrooms is OneBedroom AND Facilities is None AND TransportationAccess is Tram THEN Price is Low
Rule 9	IF View is MarinaView AND Size is Moderate AND NumberOfBedrooms is OneBedroom AND Facilities is Pool AND TransportationAccess is Tram THEN Price is Average
Rule 10	IF View is MarinaView AND Size is Moderate AND NumberOfBedrooms is TwoBedroom AND Facilities is Gym AND TransportationAccess is Tram THEN Price is High
Rule 11	IF View is MarinaView AND Size is Spacious AND NumberOfBedrooms is Studio AND Facilities is None AND TransportationAccess is None THEN Price is Low
Rule 12	IF View is MarinaView AND Size is Spacious AND NumberOfBedrooms is OneBedroom AND Facilities is Gym AND TransportationAccess is None THEN Price is Average
Rule 13	IF View is MarinaView AND Size is Spacious AND NumberOfBedrooms is TwoBedroom AND Facilities is None AND TransportationAccess is Tram THEN Price is Expensive
Rule 14	IF View is MarinaView AND Size is Large AND NumberOfBedrooms is Studio AND Facilities is Pool AND TransportationAccess is None THEN Price is Average
Rule 15	IF View is MarinaView AND Size is Large AND NumberOfBedrooms is OneBedroom AND Facilities is None AND TransportationAccess is Metro THEN Price is Average
Rule 16	IF View is MarinaView AND Size is Large AND NumberOfBedrooms is

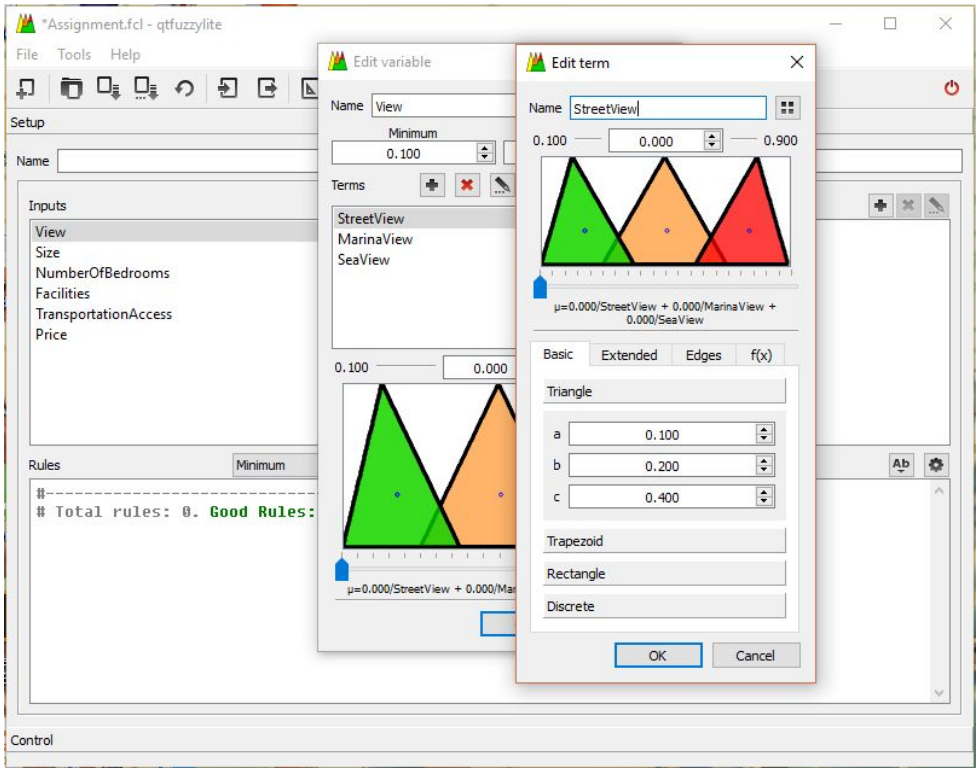
	TwoBedroom AND Facilities is Gym AND TransportationAccess is Tram THEN Price is High
Rule 17	IF View is MarinaView AND Size is Large AND NumberOfBedrooms is ThreeBedrooms AND Facilities is Pool AND TransportationAccess is Metro THEN Price is Expensive
Rule 18	IF View is SeaView AND Size is Small AND NumberOfBedrooms is Studio AND Facilities is Gym AND TransportationAccess is Tram THEN Price is Average
Rule 19	IF View is SeaView AND Size is Small AND NumberOfBedrooms is TwoBedroom AND Facilities is Pool AND TransportationAccess is None THEN Price is High
Rule 20	IF View is SeaView AND Size is Small AND NumberOfBedrooms is ThreeBedrooms AND Facilities is Gym AND TransportationAccess is Metro THEN Price is Average
Rule 21	IF View is SeaView AND Size is Moderate AND NumberOfBedrooms is OneBedroom AND Facilities is Gym AND TransportationAccess is Metro THEN Price is High
Rule 22	IF View is SeaView AND Size is Moderate AND NumberOfBedrooms is TwoBedroom AND Facilities is Pool AND TransportationAccess is None THEN Price is Average
Rule 23	IF View is SeaView AND Size is Moderate AND NumberOfBedrooms is ThreeBedrooms AND Facilities is None AND TransportationAccess is Tram THEN Price is High
Rule 24	IF View is SeaView AND Size is Spacious AND NumberOfBedrooms is Studio AND Facilities is None AND TransportationAccess is Tram THEN Price is Low
Rule 25	IF View is SeaView AND Size is Spacious AND NumberOfBedrooms is OneBedroom AND Facilities is Pool AND TransportationAccess is Tram THEN Price is High
Rule 26	IF View is SeaView AND Size is Spacious AND NumberOfBedrooms is TwoBedroom AND Facilities is None AND TransportationAccess is Metro THEN Price is High
Rule 27	IF View is SeaView AND Size is Large AND NumberOfBedrooms is Studio AND Facilities is None AND TransportationAccess is Metro THEN Price is Average

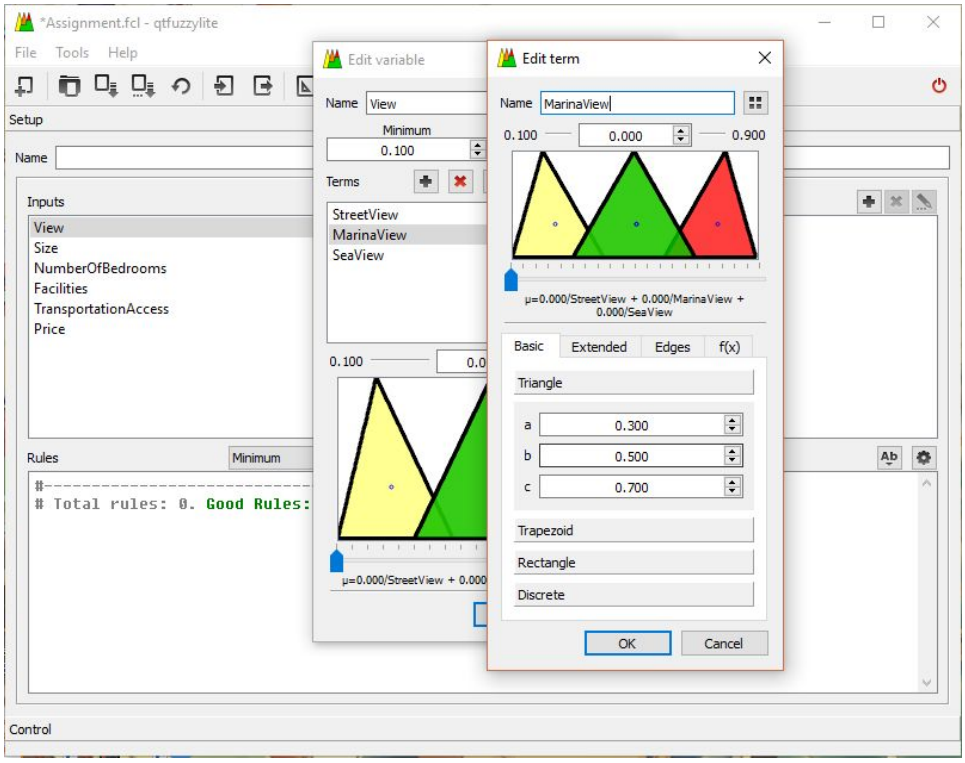
Rule 28	IF View is SeaView AND Size is Large AND NumberOfBedrooms is OneBedroom AND Facilities is None AND TransportationAccess is None THEN Price is Average
Rule 29	IF View is SeaView AND Size is Large AND NumberOfBedrooms is TwoBedroom AND Facilities is Gym AND TransportationAccess is Tram THEN Price is High
Rule 30	IF View is SeaView AND Size is Large AND NumberOfBedrooms is ThreeBedrooms AND Facilities is Pool AND TransportationAccess is Metro THEN Price is Expensive

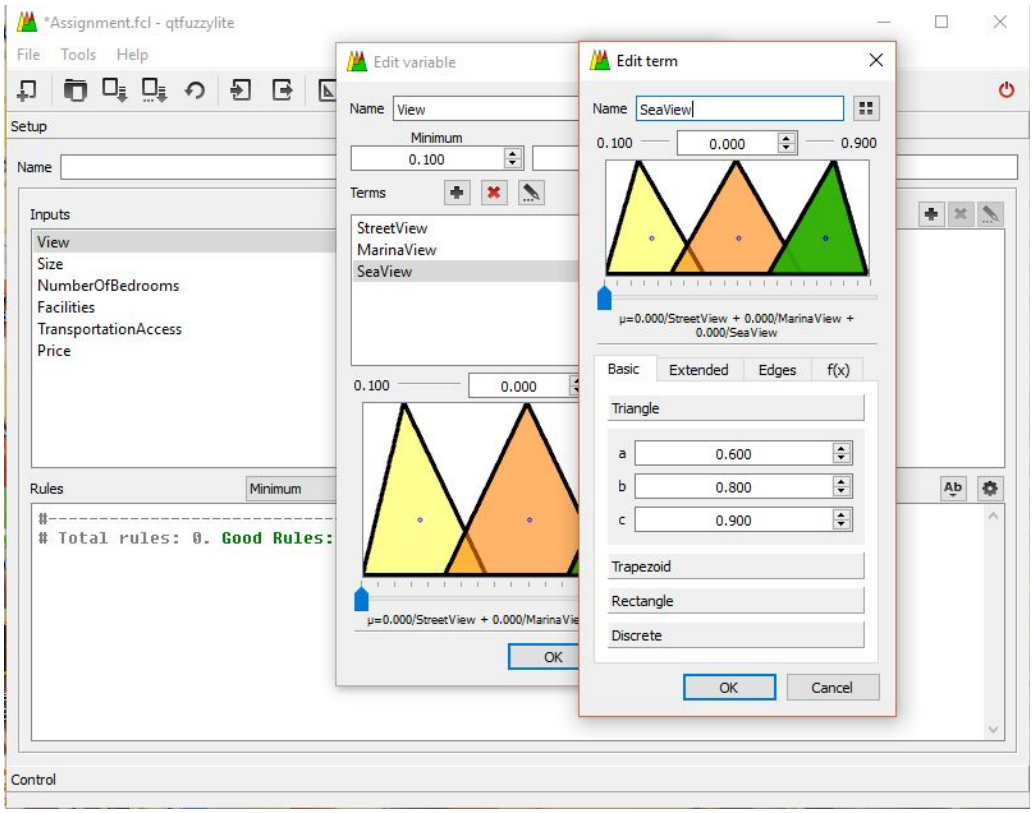
Screenshots

Inputs

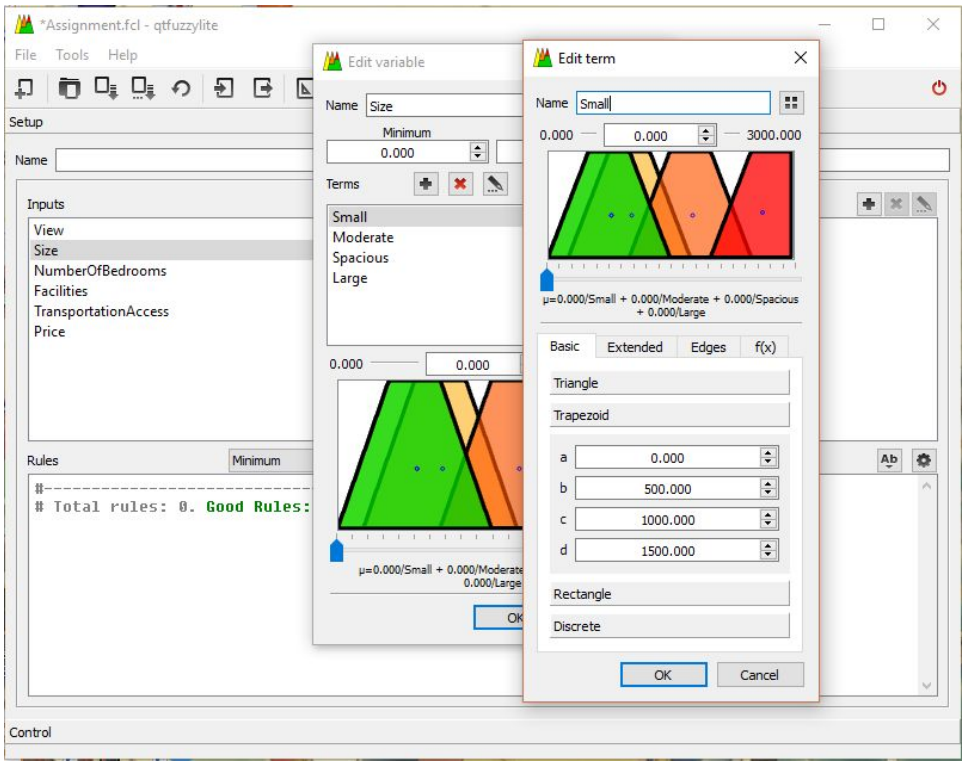
View

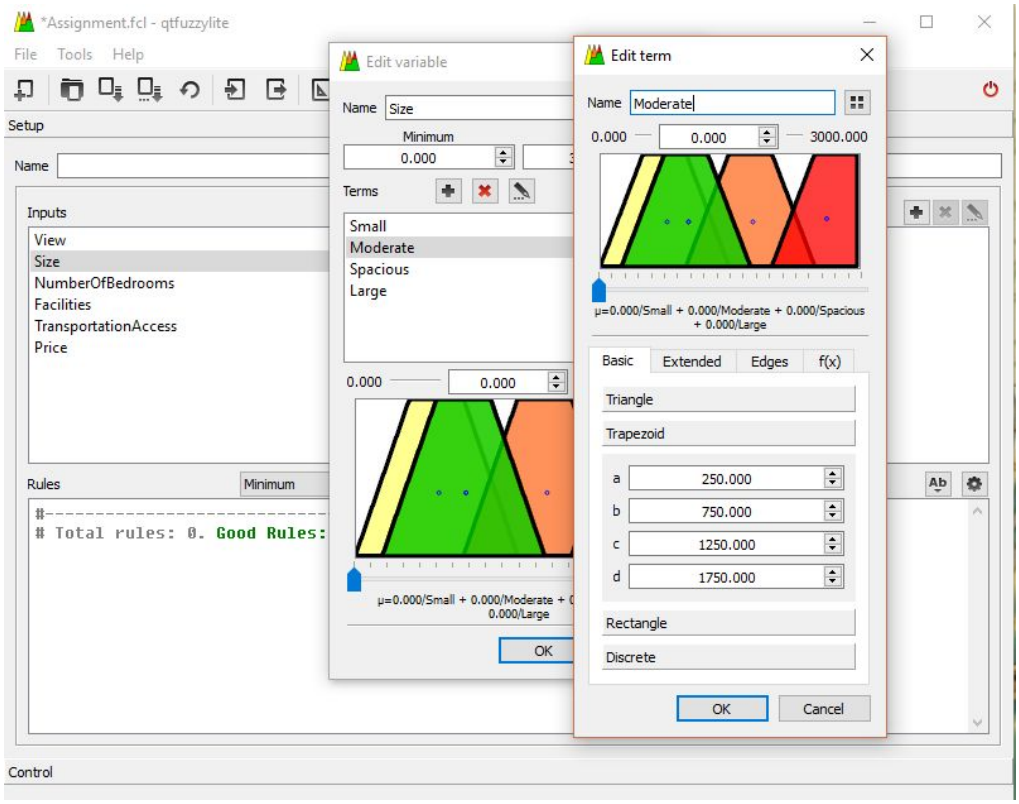


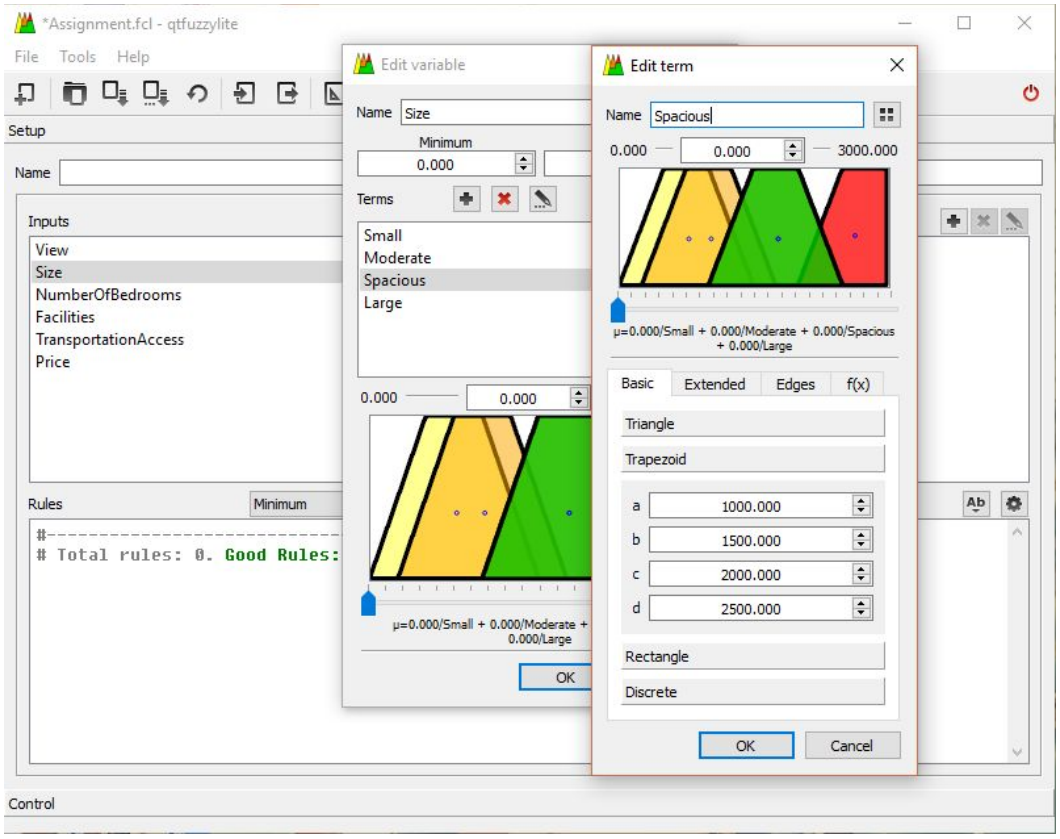

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name View
Minimum
0.100
Terms
StreetView
MarinaView
SeaView
Edit term
Name MarinaView
0.100 0.000 0.900
μ=0.000/StreetView + 0.000/MarinaView + 0.000/SeaView
Basic Extended Edges f(x)
Triangle
a 0.300
b 0.500
c 0.700
Trapezoid
Rectangle
Discrete
OK Cancel

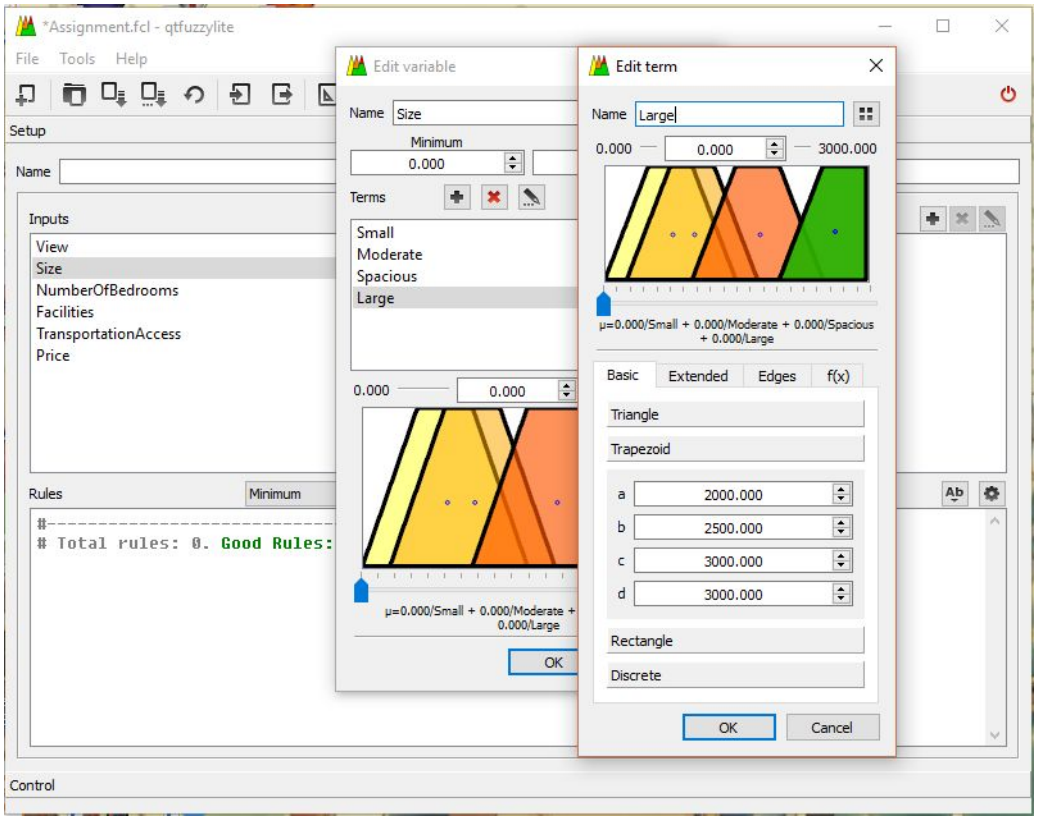

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name View
Minimum
0.100
Terms
StreetView
MarinaView
SeaView
Edit term
Name SeaView
0.100 0.000 0.900
μ=0.000/StreetView + 0.000/MarinaView + 0.000/SeaView
Basic Extended Edges f(x)
Triangle
a 0.600
b 0.800
c 0.900
Trapezoid
Rectangle
Discrete
OK Cancel

Size

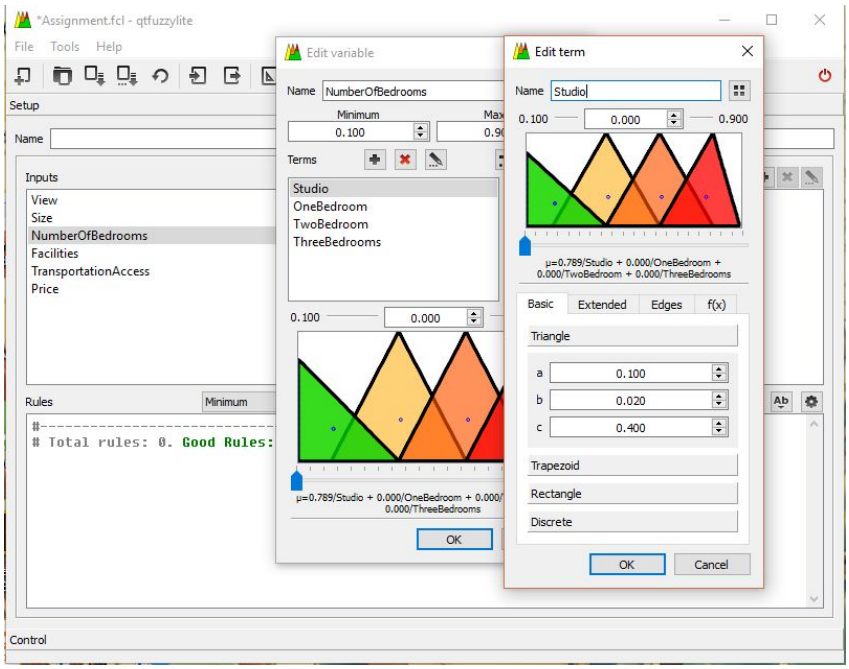

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name Size
Minimum
0.000
Terms
Small
Moderate
Spacious
Large
Edit term
Name Small
0.000 0.000 3000.000
μ=0.000/Small + 0.000/Moderate + 0.000/Spacious + 0.000/Large
Basic Extended Edges f(x)
Triangle
Trapezoid
a 0.000
b 500.000
c 1000.000
d 1500.000
Rectangle
Discrete
OK
Cancel

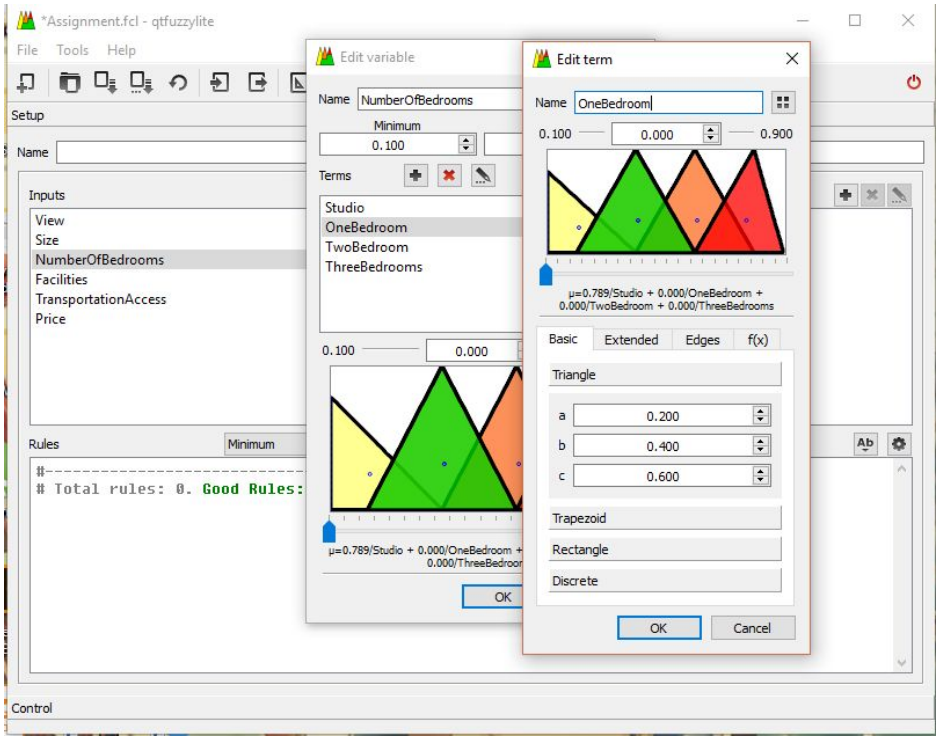

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name Size
Minimum
0.000
Terms
Small
Moderate
Spacious
Large
OK
Edit term
Name Moderate
0.000 0.000 3000.000
μ=0.000/Small + 0.000/Moderate + 0.000/Spacious + 0.000/Large
Basic Extended Edges f(x)
Triangle
Trapezoid
a 250.000
b 750.000
c 1250.000
d 1750.000
Rectangle
Discrete
OK
Cancel

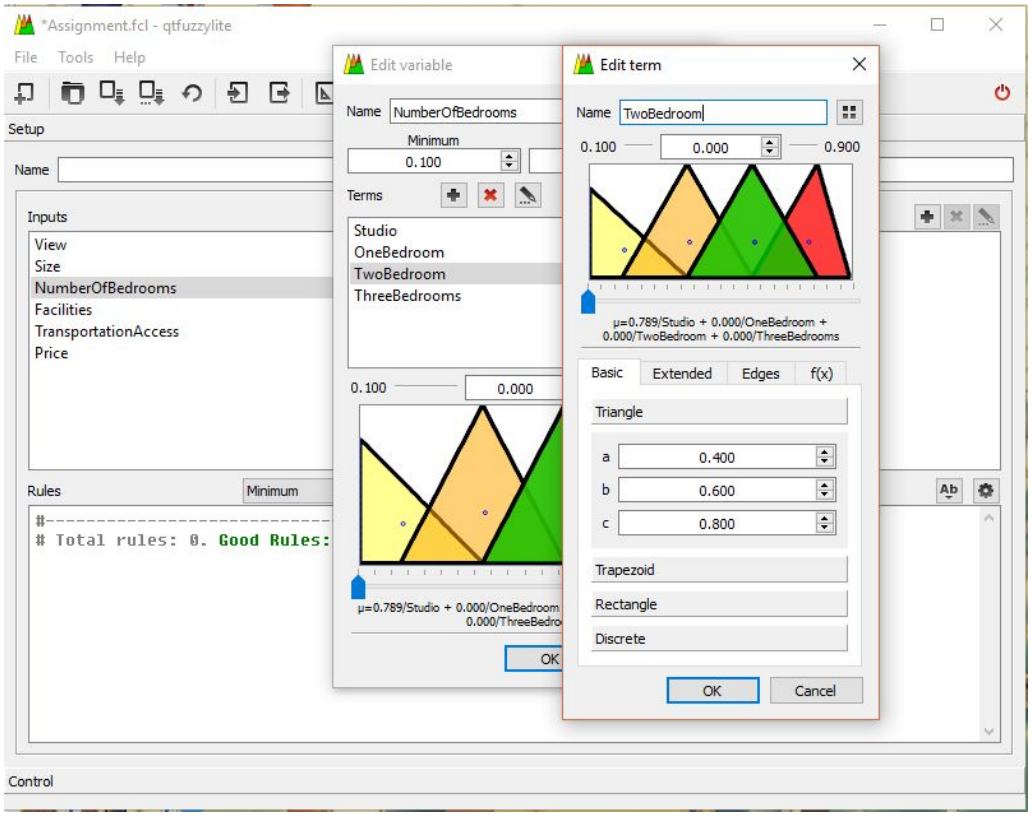

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name Size
Minimum
0.000
Terms
Small
Moderate
Spacious
Large
0.000
0.000
μ=0.000/Small + 0.000/Moderate + 0.000/Large
OK
Edit term
Name Spacious
0.000
0.000
3000.000
μ=0.000/Small + 0.000/Moderate + 0.000/Spacious + 0.000/Large
Basic
Extended
Edges
f(x)
Triangle
Trapezoid
a 1000.000
b 1500.000
c 2000.000
d 2500.000
Rectangle
Discrete
OK
Cancel

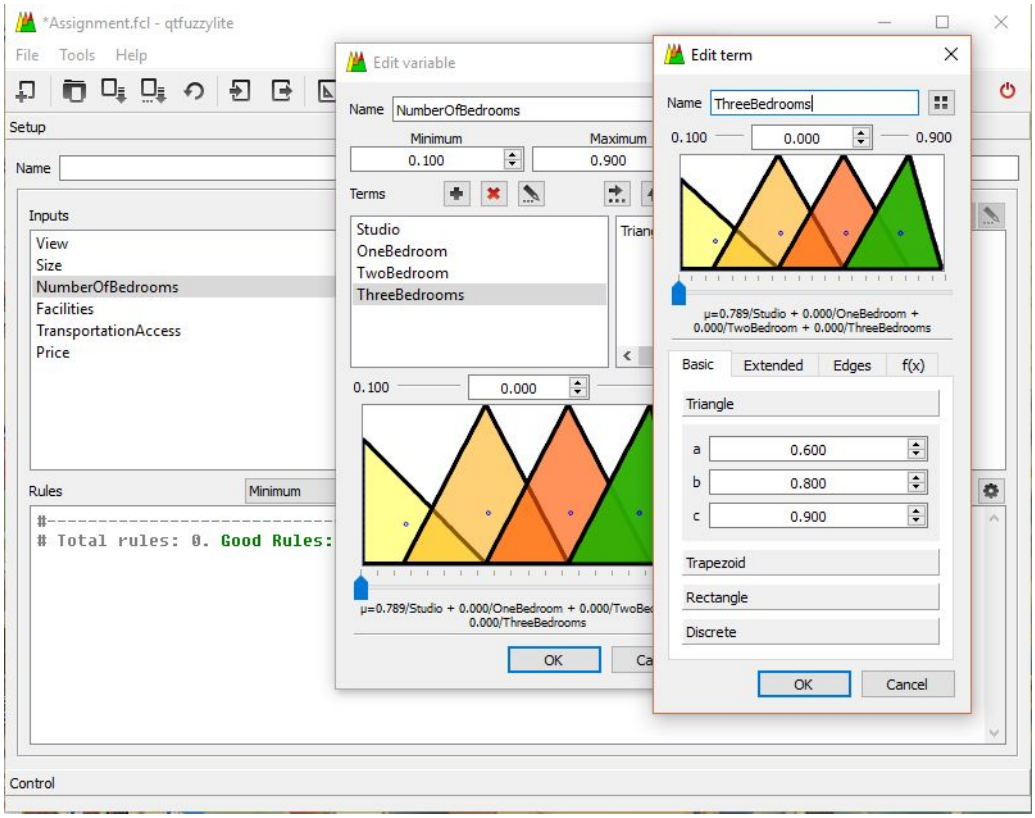

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name Size
Minimum
0.000
Terms
Small
Moderate
Spacious
Large
0.000
0.000
μ=0.000/Small + 0.000/Moderate + 0.000/Large
OK
Edit term
Name Large
0.000
0.000
3000.000
μ=0.000/Small + 0.000/Moderate + 0.000/Spacious + 0.000/Large
Basic
Extended
Edges
f(x)
Triangle
Trapezoid
a 2000.000
b 2500.000
c 3000.000
d 3000.000
Rectangle
Discrete
OK
Cancel

Number Of Bedrooms

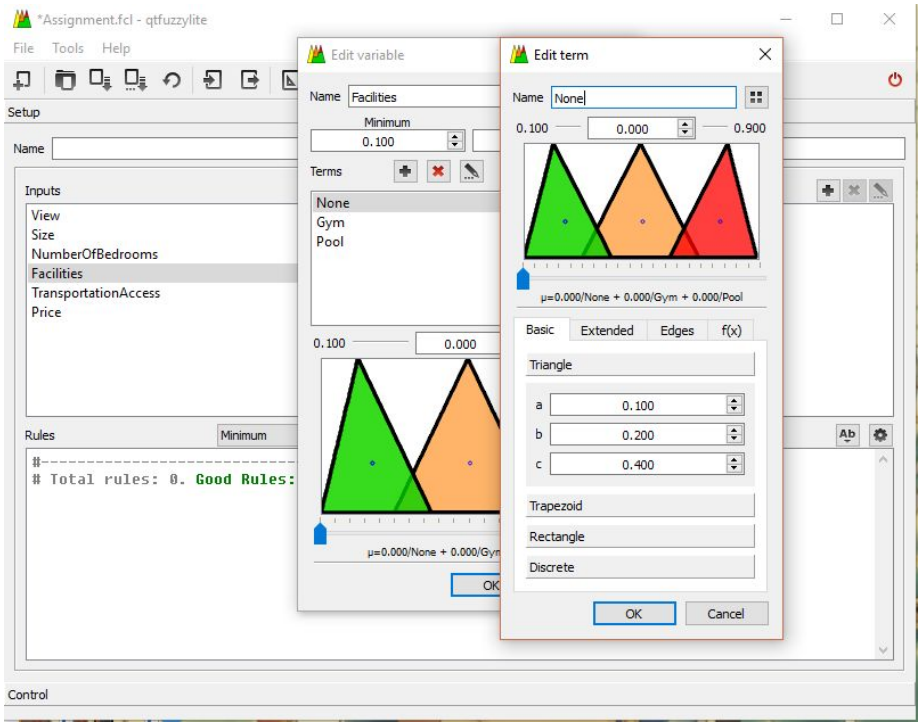


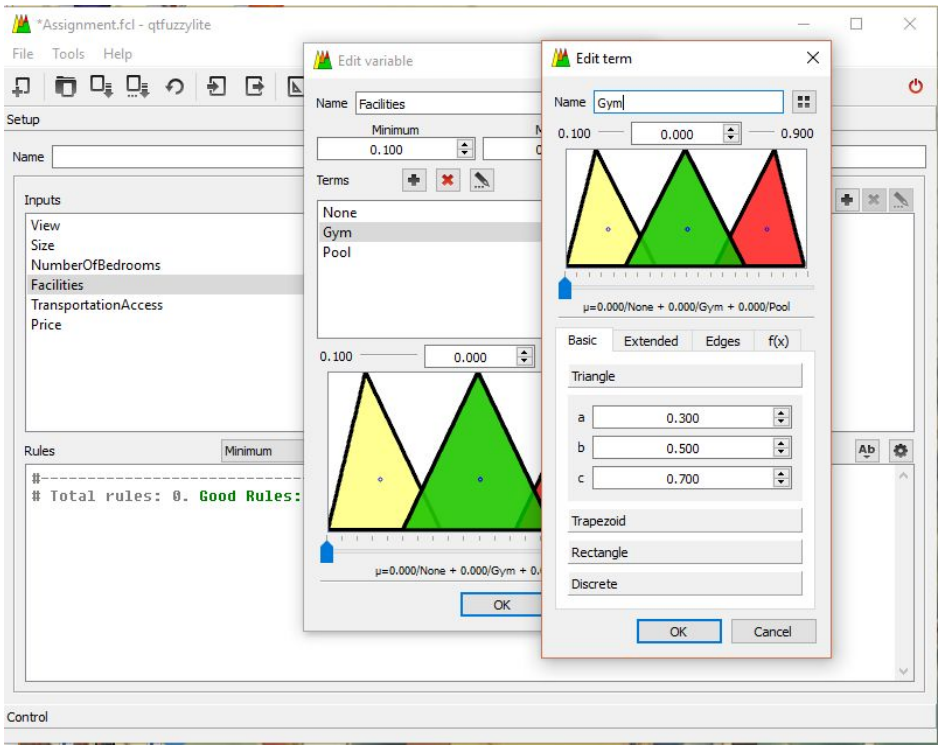


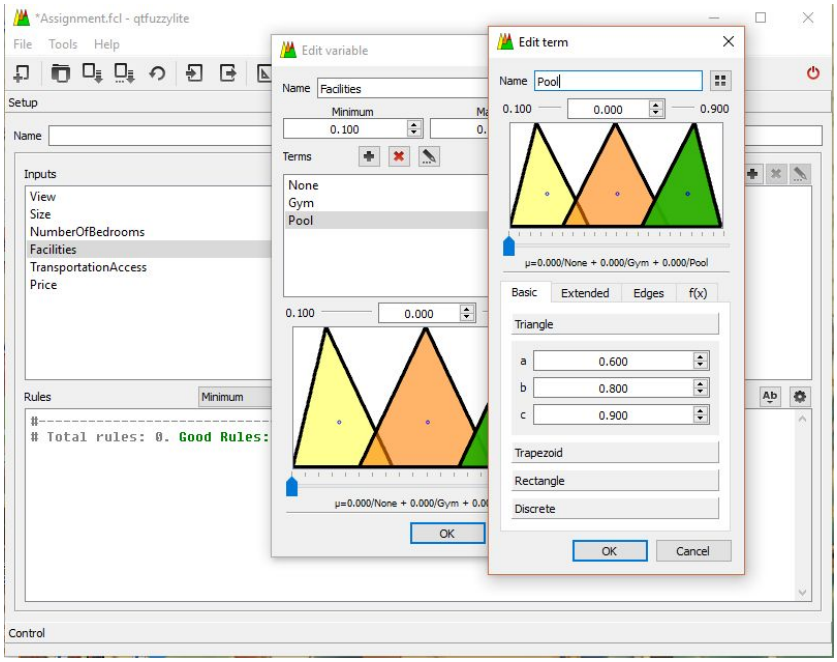

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name NumberOfBedrooms
Minimum
0.100
Terms
Studio
OneBedroom
TwoBedroom
ThreeBedrooms
0.100
0.000
OK
Edit term
Name TwoBedroom
0.100
0.000
0.900
μ=0.789/Studio + 0.000/OneBedroom +
0.000/TwoBedroom + 0.000/ThreeBedrooms
Basic
Extended
Edges
f(x)
Triangle
a
0.400
b
0.600
c
0.800
Trapezoid
Rectangle
Discrete
OK
Cancel


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name NumberOfBedrooms
Minimum
0.100
Maximum
0.900
Terms
Studio
OneBedroom
TwoBedroom
ThreeBedrooms
0.100
0.000
OK
Edit term
Name ThreeBedrooms
0.100
0.000
0.900
μ=0.789/Studio + 0.000/OneBedroom +
0.000/TwoBedroom + 0.000/ThreeBedrooms
Basic
Extended
Edges
f(x)
Triangle
a
0.600
b
0.800
c
0.900
Trapezoid
Rectangle
Discrete
OK
Cancel

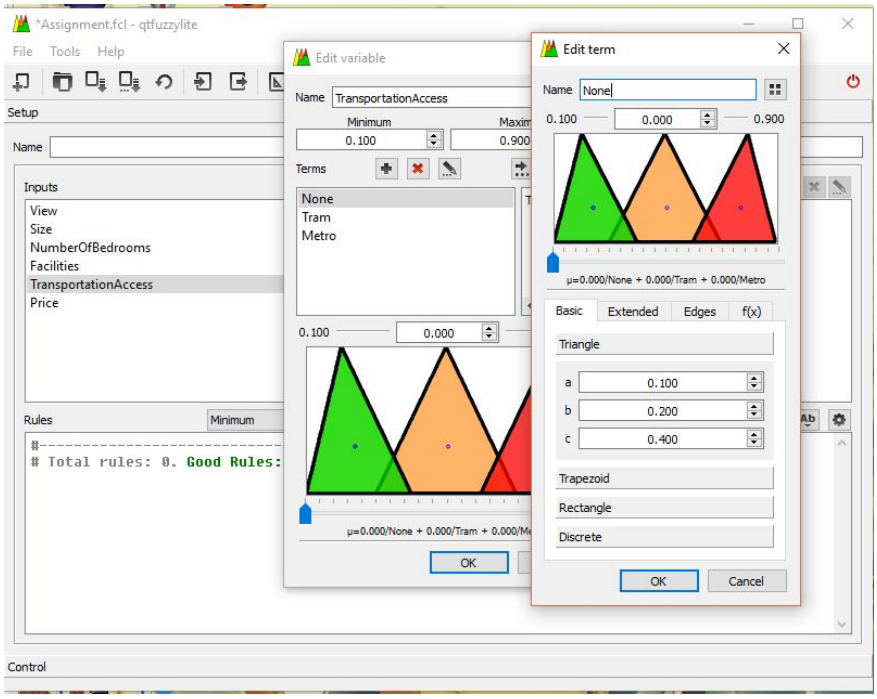
Facilities

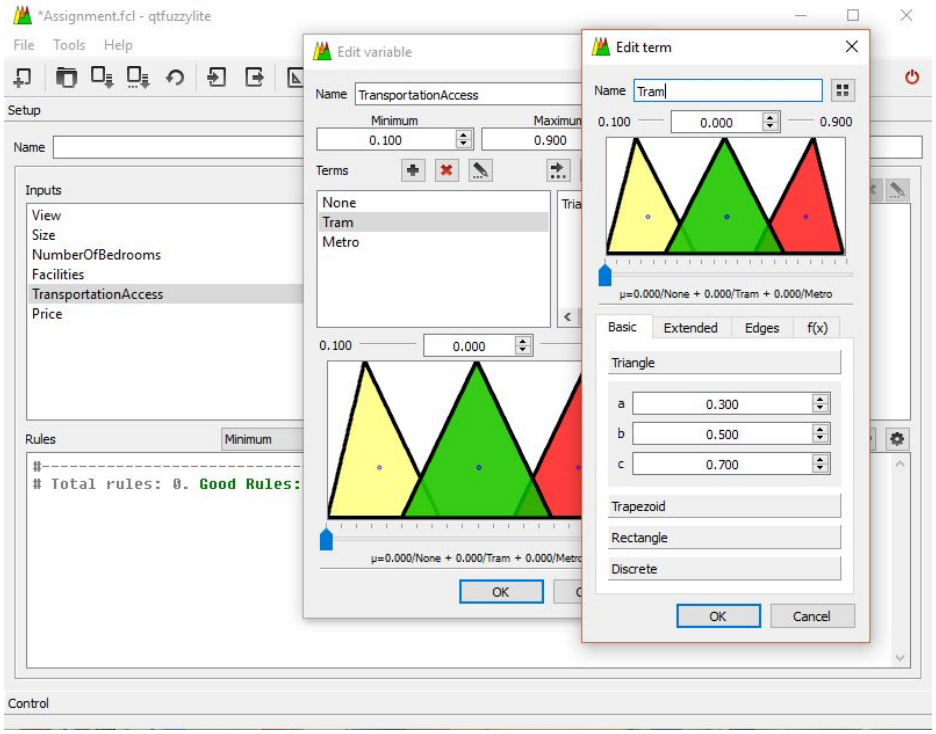

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name Facilities
Minimum
0.100
Terms
None
Gym
Pool
0.100
0.000
OK
Edit term
Name None
0.100
0.000
0.900
μ=0.000/None + 0.000/Gym + 0.000/Pool
Basic
Extended
Edges
f(x)
Triangle
a 0.100
b 0.200
c 0.400
Trapezoid
Rectangle
Discrete
OK
Cancel

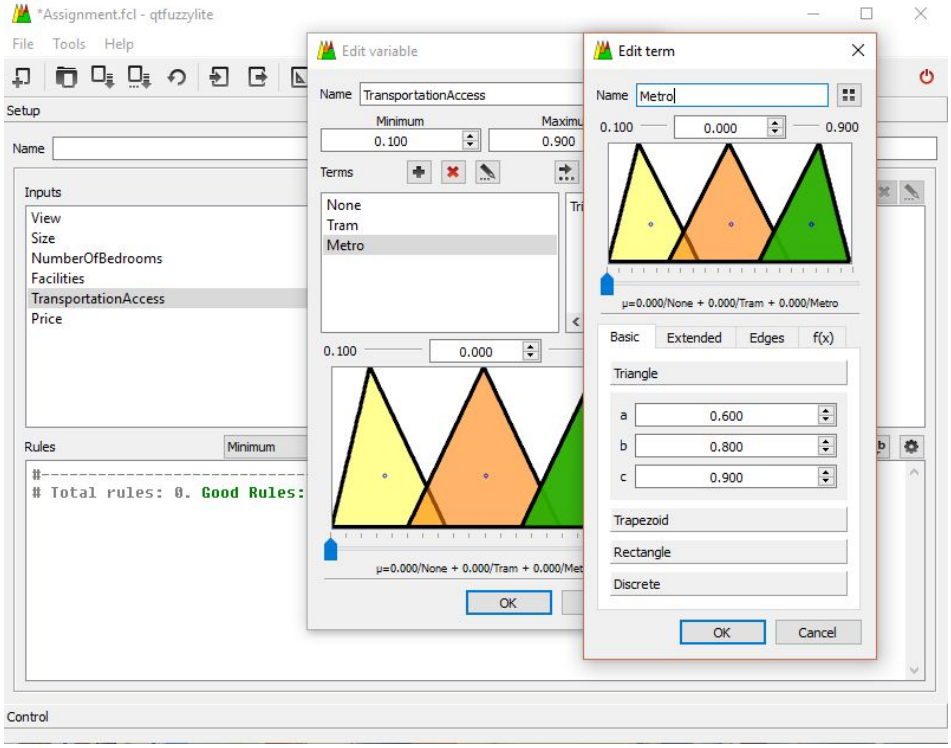

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name Facilities
Minimum
0.100
Terms
None
Gym
Pool
0.100
0.000
OK
Edit term
Name Gym
0.100
0.000
0.900
μ=0.000/None + 0.000/Gym + 0.000/Pool
Basic
Extended
Edges
f(x)
Triangle
a 0.300
b 0.500
c 0.700
Trapezoid
Rectangle
Discrete
OK
Cancel



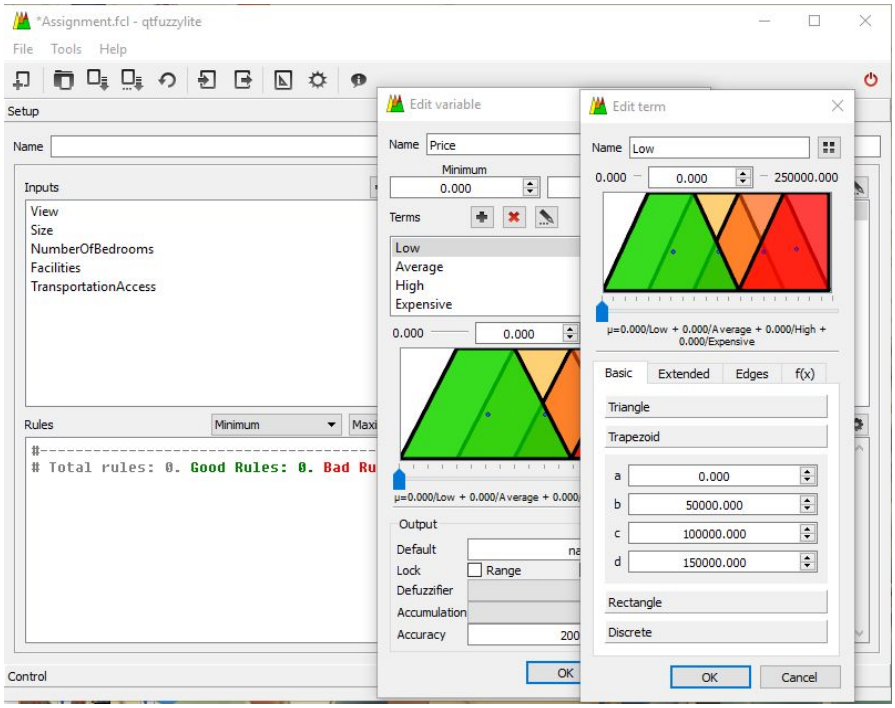
Transportation Access

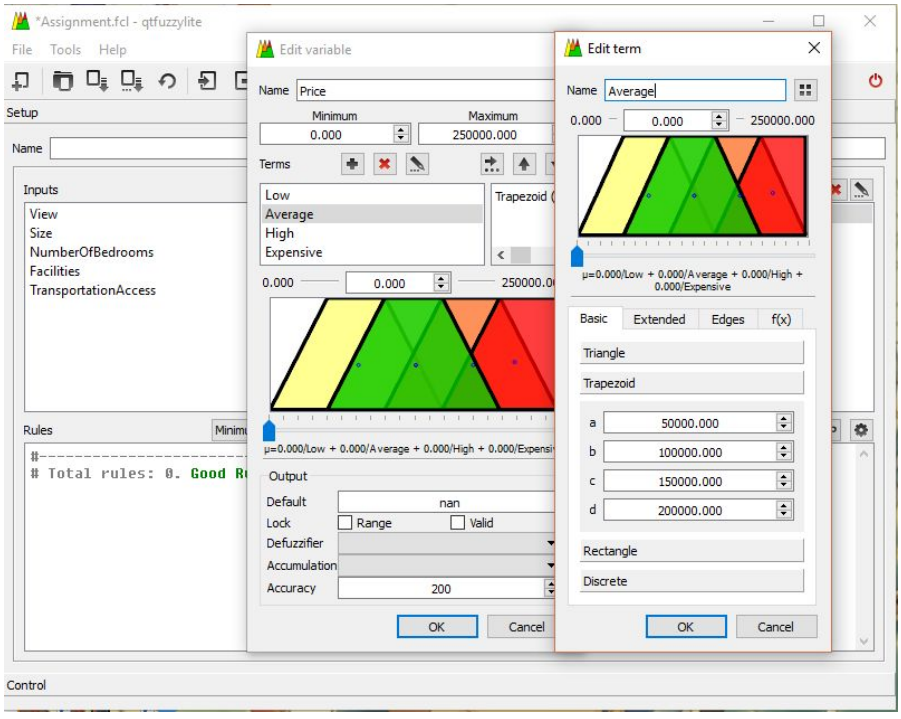


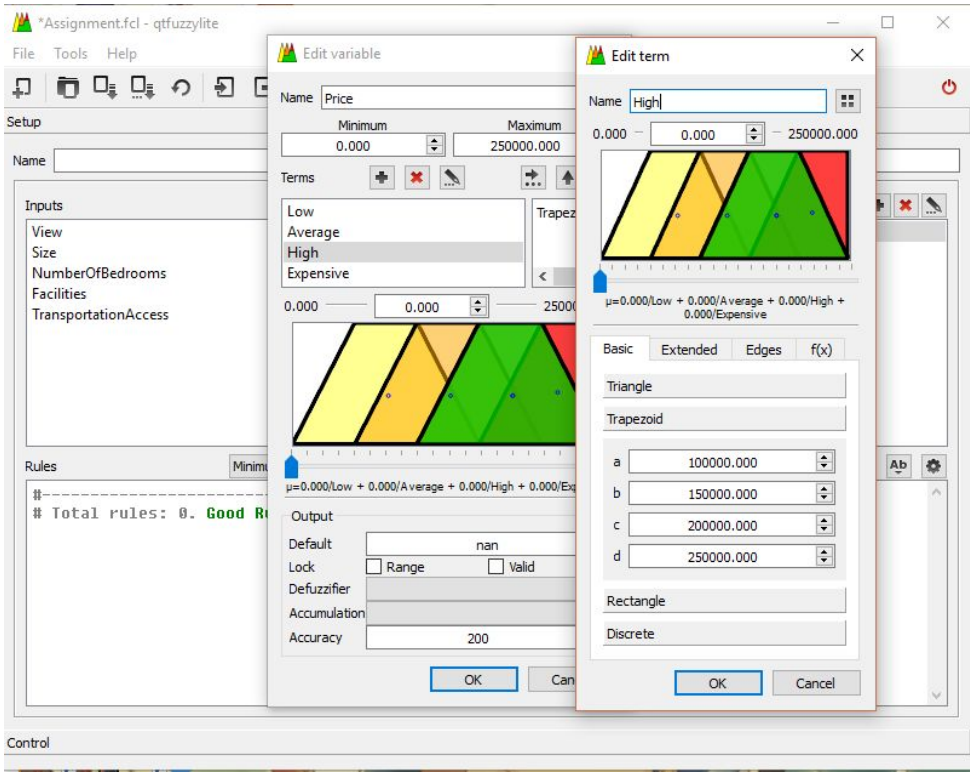

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name TransportationAccess
Minimum 0.100
Maximum 0.900
Terms
None
Tram
Metro
0.100 0.000
μ=0.000/None + 0.000/Tram + 0.000/Metro
OK
Edit term
Name Tram
0.100 0.000 0.900
μ=0.000/None + 0.000/Tram + 0.000/Metro
Basic Extended Edges f(x)
Triangle
a 0.300
b 0.500
c 0.700
Trapezoid
Rectangle
Discrete
OK Cancel

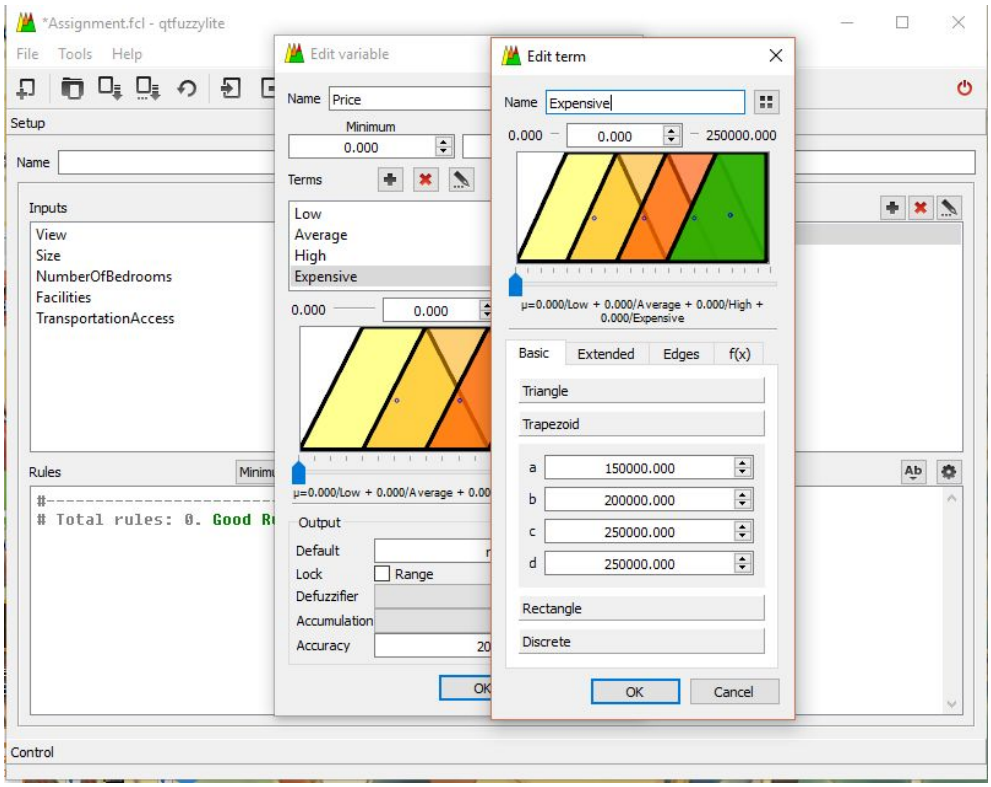

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Price
Rules
Minimum
Total rules: 0. Good Rules:
Control
Edit variable
Name TransportationAccess
Minimum 0.100
Maximum 0.900
Terms
None
Tram
Metro
0.100 0.000
OK
Edit term
Name Metro
0.100 0.000 0.900
μ=0.000/None + 0.000/Tram + 0.000/Metro
Basic Extended Edges f(x)
Triangle
a 0.600
b 0.800
c 0.900
Trapezoid
Rectangle
Discrete
OK Cancel

Price


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Rules
Minimum
Total rules: 0. Good Rules: 0. Bad Ru
Control
Edit variable
Name Price
Minimum
0.000
Terms
Low
Average
High
Expensive
0.000
0.000
Output
Default
Lock
Range
Defuzzifier
Accumulation
Accuracy
OK
Edit term
Name Low
0.000
0.000
250000.000
μ=0.000/Low + 0.000/Average + 0.000/High + 0.000/Expensive
Basic
Extended
Edges
f(x)
Triangle
Trapezoid
a 0.000
b 50000.000
c 100000.000
d 150000.000
Rectangle
Discrete
OK
Cancel


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Rules
Total rules: 0. Good R
Control
Edit variable
Name Price
Minimum
0.000
Maximum
250000.000
Terms
Low
Average
High
Expensive
Trapezoid
0.000
0.000
Output
Default
nan
Lock
Range
Valid
Defuzzifier
Accumulation
Accuracy
200
OK
Cancel
Edit term
Name Average
0.000
0.000
250000.000
μ=0.000/Low + 0.000/Average + 0.000/High + 0.000/Expensive
Basic
Extended
Edges
f(x)
Triangle
Trapezoid
a 50000.000
b 100000.000
c 150000.000
d 200000.000
Rectangle
Discrete
OK
Cancel

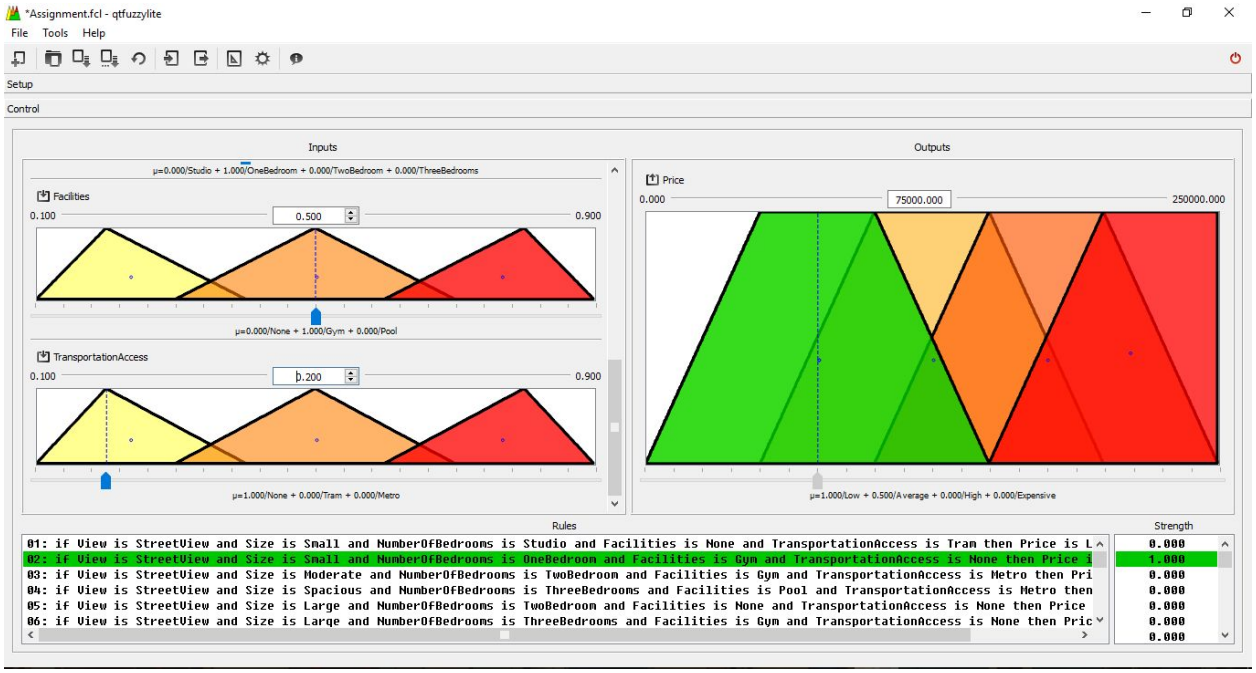

*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Rules
Total rules: 0. Good R
Control
Edit variable
Name Price
Minimum 0.000
Maximum 250000.000
Terms
Low
Average
High
Expensive
Output
Default nan
Lock Range Valid
Defuzzifier
Accumulation
Accuracy 200
OK
Edit term
Name High
0.000 0.000 250000.000
μ=0.000/Low + 0.000/Average + 0.000/High + 0.000/Expensive
Basic Extended Edges f(x)
Triangle
Trapezoid
a 100000.000
b 150000.000
c 200000.000
d 250000.000
Rectangle
Discrete
OK Cancel


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Name
Inputs
View
Size
NumberOfBedrooms
Facilities
TransportationAccess
Rules
Total rules: 0. Good R
Control
Edit variable
Name Price
Minimum 0.000
Terms
Low
Average
High
Expensive
Output
Default
Lock Range
Defuzzifier
Accumulation
Accuracy
Edit term
Name Expensive
0.000 0.000 250000.000
μ=0.000/Low + 0.000/Average + 0.000/High + 0.000/Expensive
Basic Extended Edges f(x)
Triangle
Trapezoid
a 150000.000
b 200000.000
c 250000.000
d 250000.000
Rectangle
Discrete
OK Cancel

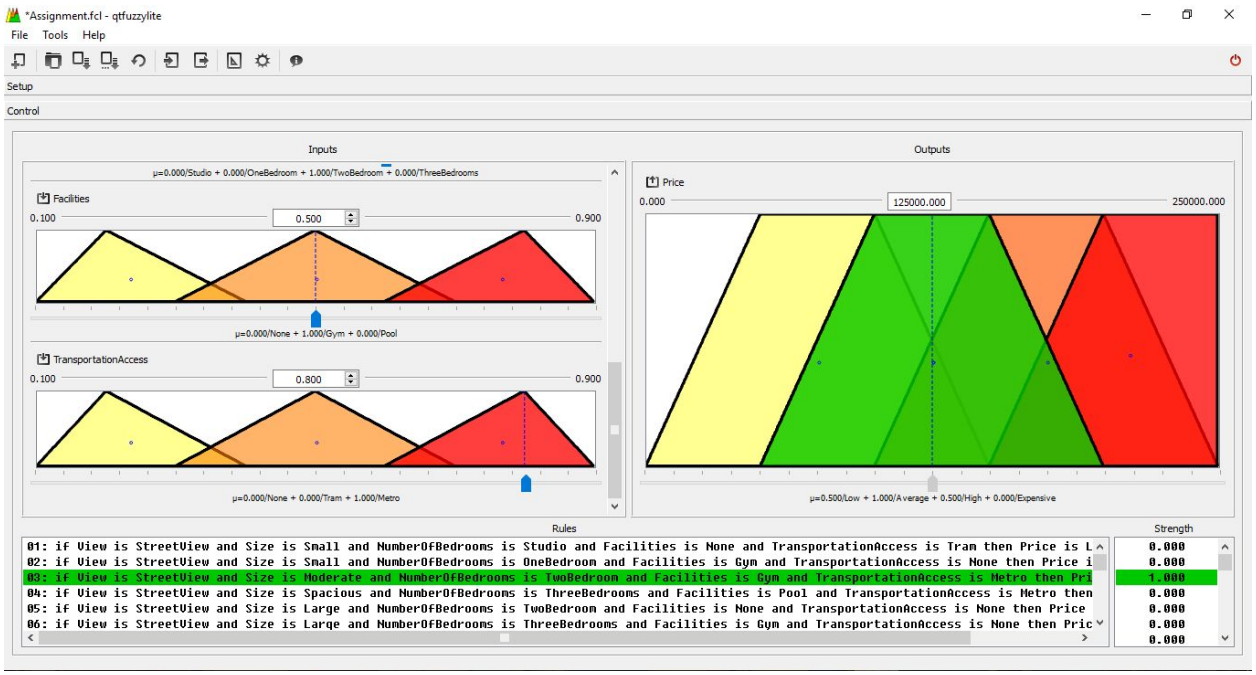
Rules

Rule 1

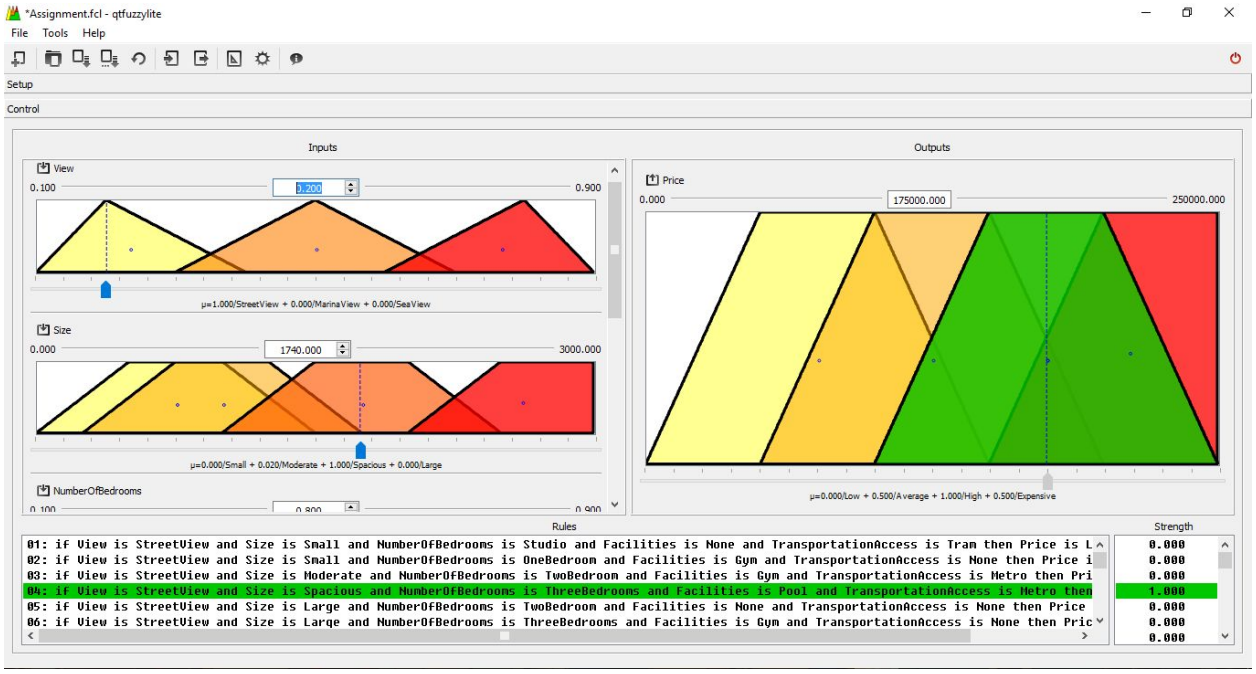
Rule 2



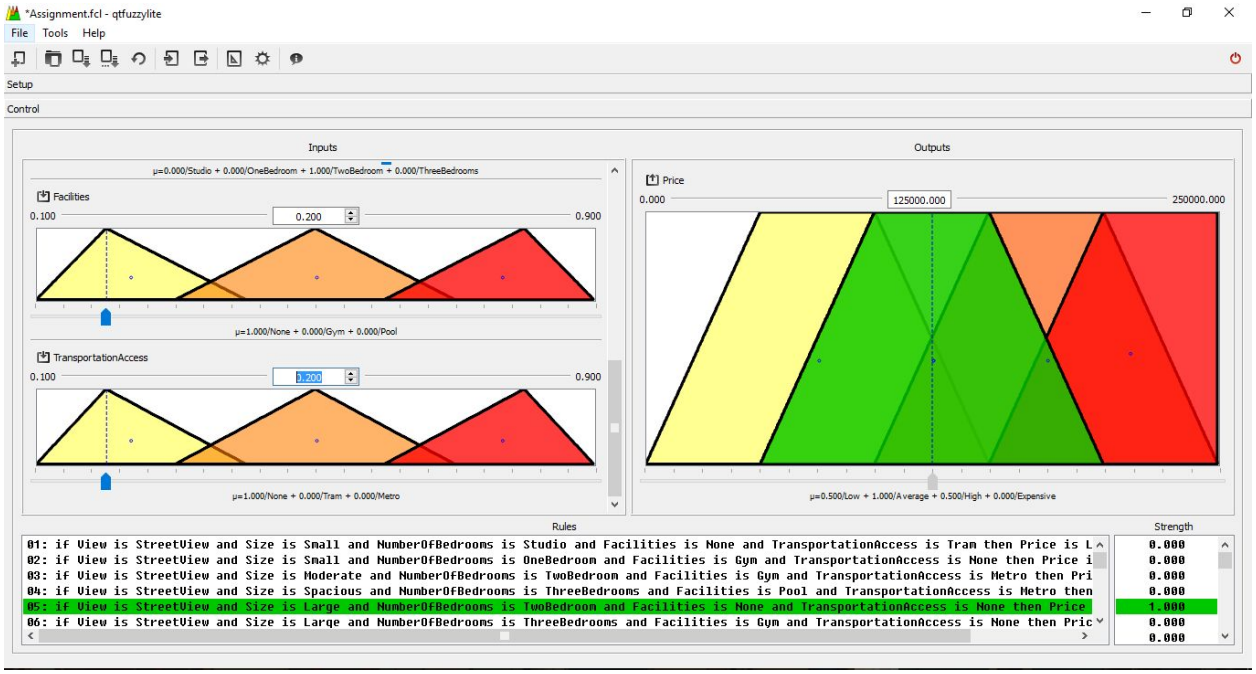
Rule 3


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Control
Inputs
μ=0.000/Studio + 0.000/OneBedroom + 1.000/TwoBedroom + 0.000/ThreeBedrooms
Facilities
0.100
0.500
0.900
μ=0.000/None + 1.000/Gym + 0.000/Pool
TransportationAccess
0.100
0.800
0.900
μ=0.000/None + 0.000/Tram + 1.000/Metro
Outputs
Price
0.000
125000.000
250000.000
μ=0.500/Low + 1.000/Average + 0.500/High + 0.000/Expensive
Rules
Strength
01: if View is StreetView and Size is Small and NumberOfBedrooms is Studio and Facilities is None and TransportationAccess is Tram then Price is L
02: if View is StreetView and Size is Small and NumberOfBedrooms is OneBedroom and Facilities is Gym and TransportationAccess is None then Price i
03: if View is StreetView and Size is Moderate and NumberOfBedrooms is TwoBedroom and Facilities is Gym and TransportationAccess is Metro then Pri
04: if View is StreetView and Size is Spacious and NumberOfBedrooms is ThreeBedrooms and Facilities is Pool and TransportationAccess is Metro then
05: if View is StreetView and Size is Large and NumberOfBedrooms is TwoBedroom and Facilities is None and TransportationAccess is None then Price
06: if View is StreetView and Size is Large and NumberOfBedrooms is ThreeBedrooms and Facilities is Gym and TransportationAccess is None then Pric
0.000
0.000
1.000
0.000
0.000
0.000

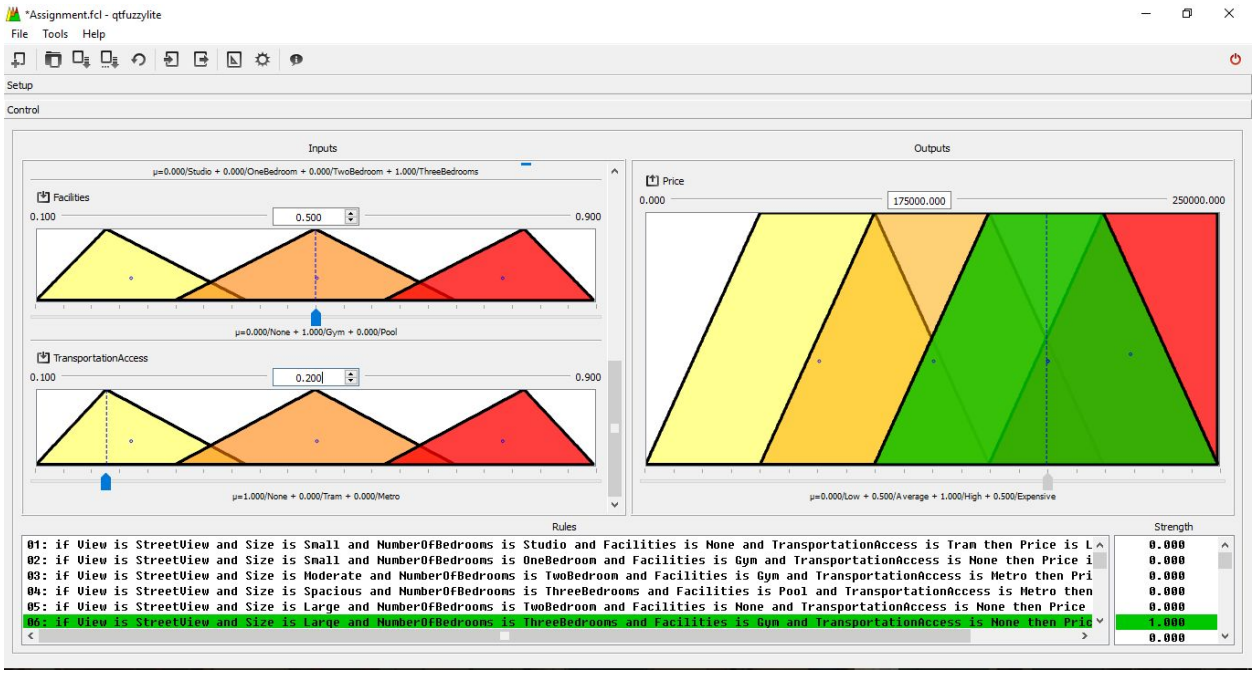
Rule 4


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Control
Inputs
View
0.100
0.200
0.900
μ=1.000/StreetView + 0.000/MarinaView + 0.000/SeaView
Size
0.000
1740.000
3000.000
μ=0.000/Small + 0.020/Moderate + 1.000/Spacious + 0.000/Large
NumberOfBedrooms
Outputs
Price
0.000
175000.000
250000.000
μ=0.000/Low + 0.500/Average + 1.000/High + 0.500/Expensive
Rules
Strength
01: if View is StreetView and Size is Small and NumberOfBedrooms is Studio and Facilities is None and TransportationAccess is Tram then Price is L
02: if View is StreetView and Size is Small and NumberOfBedrooms is OneBedroom and Facilities is Gym and TransportationAccess is None then Price i
03: if View is StreetView and Size is Moderate and NumberOfBedrooms is TwoBedroom and Facilities is Gym and TransportationAccess is Metro then Pri
04: if View is StreetView and Size is Spacious and NumberOfBedrooms is ThreeBedrooms and Facilities is Pool and TransportationAccess is Metro then
05: if View is StreetView and Size is Large and NumberOfBedrooms is TwoBedroom and Facilities is None and TransportationAccess is None then Price
06: if View is StreetView and Size is Large and NumberOfBedrooms is ThreeBedrooms and Facilities is Gym and TransportationAccess is None then Pric
0.000
0.000
0.000
1.000
0.000
0.000
0.000

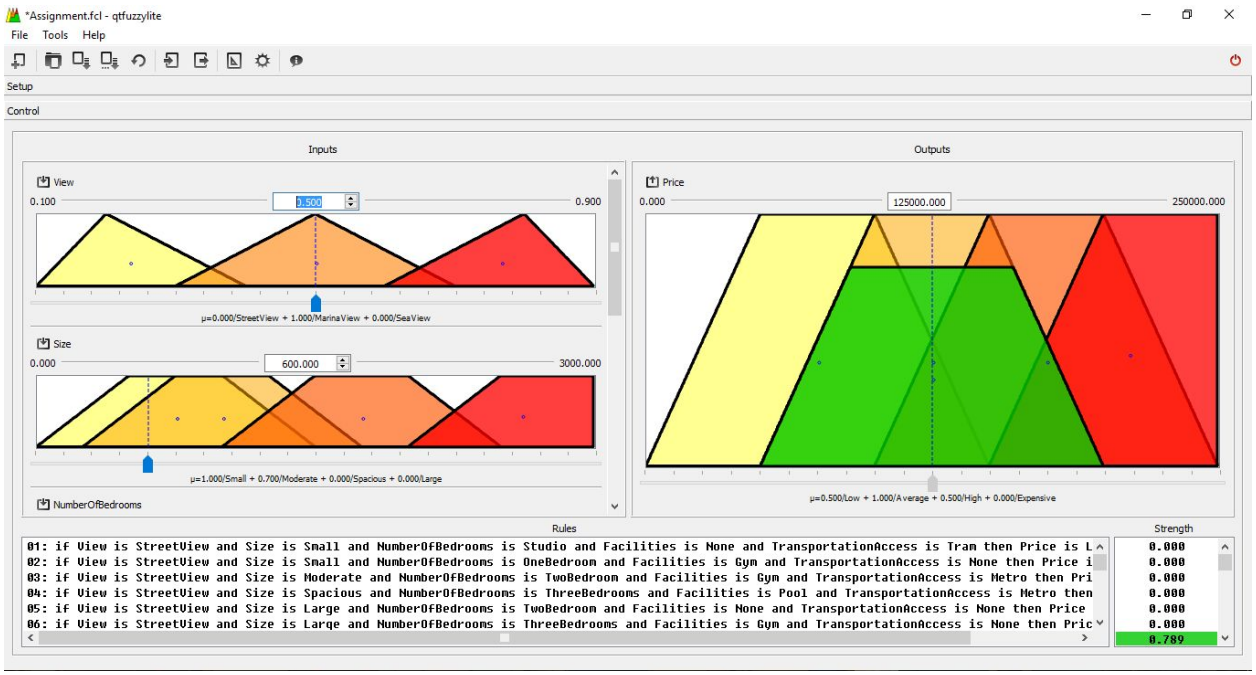
Rule 5


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Control
Inputs
μ=0.000/Studio + 0.000/OneBedroom + 1.000/TwoBedroom + 0.000/ThreeBedrooms
Facilities
0.100
0.200
0.900
μ=1.000/None + 0.000/Gym + 0.000/Pool
TransportationAccess
0.100
0.200
0.900
μ=1.000/None + 0.000/Tram + 0.000/Metro
Outputs
Price
0.000
125000.000
250000.000
μ=0.500/Low + 1.000/Average + 0.500/High + 0.000/Expensive
Rules
Strength
01: if View is StreetView and Size is Small and NumberOfBedrooms is Studio and Facilities is None and TransportationAccess is Tram then Price is L
02: if View is StreetView and Size is Small and NumberOfBedrooms is OneBedroom and Facilities is Gym and TransportationAccess is None then Price i
03: if View is StreetView and Size is Moderate and NumberOfBedrooms is TwoBedroom and Facilities is Gym and TransportationAccess is Metro then Pri
04: if View is StreetView and Size is Spacious and NumberOfBedrooms is ThreeBedrooms and Facilities is Pool and TransportationAccess is Metro then
05: if View is StreetView and Size is Large and NumberOfBedrooms is TwoBedroom and Facilities is None and TransportationAccess is None then Price
06: if View is StreetView and Size is Large and NumberOfBedrooms is ThreeBedrooms and Facilities is Gym and TransportationAccess is None then Pric
0.000
0.000
0.000
0.000
1.000
0.000
0.000

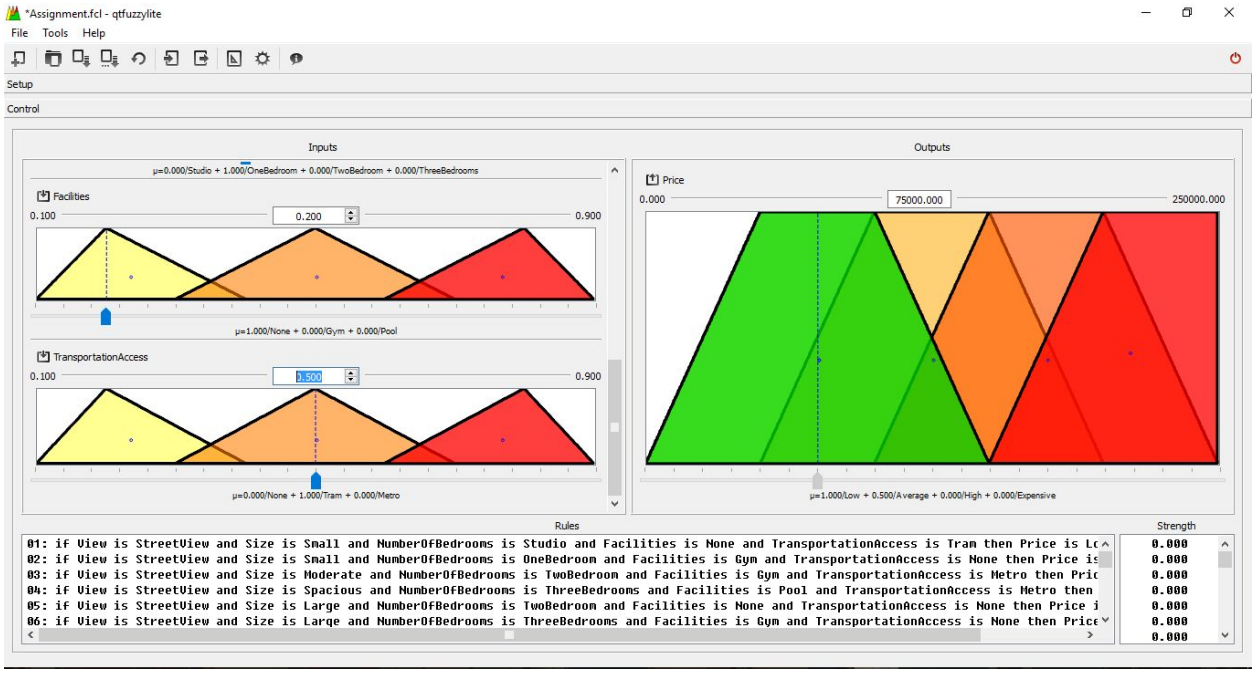
Rule 6


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Control
Inputs
μ=0.000/Studio + 0.000/OneBedroom + 0.000/TwoBedroom + 1.000/ThreeBedrooms
Facilities
0.100
0.500
0.900
μ=0.000/None + 1.000/Gym + 0.000/Pool
TransportationAccess
0.100
0.200
0.900
μ=1.000/None + 0.000/Tram + 0.000/Metro
Outputs
Price
0.000
175000.000
250000.000
μ=0.000/Low + 0.500/Average + 1.000/High + 0.500/Expensive
Rules
Strength
01: if View is StreetView and Size is Small and NumberOfBedrooms is Studio and Facilities is None and TransportationAccess is Tram then Price is L
02: if View is StreetView and Size is Small and NumberOfBedrooms is OneBedroom and Facilities is Gym and TransportationAccess is None then Price i
03: if View is StreetView and Size is Moderate and NumberOfBedrooms is TwoBedroom and Facilities is Gym and TransportationAccess is Metro then Pri
04: if View is StreetView and Size is Spacious and NumberOfBedrooms is ThreeBedrooms and Facilities is Pool and TransportationAccess is Metro then
05: if View is StreetView and Size is Large and NumberOfBedrooms is TwoBedroom and Facilities is None and TransportationAccess is None then Price
06: if View is StreetView and Size is Large and NumberOfBedrooms is ThreeBedrooms and Facilities is Gym and TransportationAccess is None then Pric
0.000
0.000
0.000
0.000
0.000
1.000
0.000

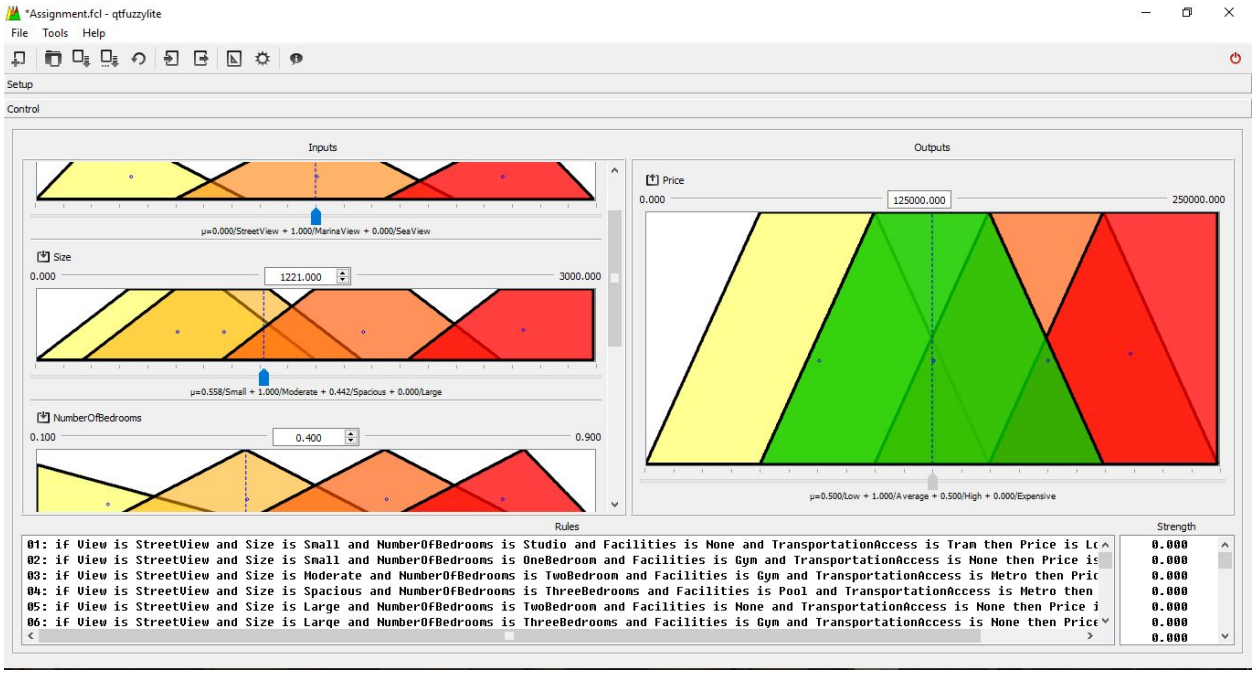
Rule 7


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Control
Inputs
View
0.100
0.500
0.900
μ=0.000/StreetView + 1.000/MarinaView + 0.000/SeaView
Size
0.000
600.000
3000.000
μ=1.000/Small + 0.700/Moderate + 0.000/Spacious + 0.000/Large
NumberOfBedrooms
Outputs
Price
0.000
125000.000
250000.000
μ=0.500/Low + 1.000/Average + 0.500/High + 0.000/Expensive
Rules
Strength
01: if View is StreetView and Size is Small and NumberOfBedrooms is Studio and Facilities is None and TransportationAccess is Tram then Price is L
02: if View is StreetView and Size is Small and NumberOfBedrooms is OneBedroom and Facilities is Gym and TransportationAccess is None then Price i
03: if View is StreetView and Size is Moderate and NumberOfBedrooms is TwoBedroom and Facilities is Gym and TransportationAccess is Metro then Pri
04: if View is StreetView and Size is Spacious and NumberOfBedrooms is ThreeBedrooms and Facilities is Pool and TransportationAccess is Metro then
05: if View is StreetView and Size is Large and NumberOfBedrooms is TwoBedroom and Facilities is None and TransportationAccess is None then Price
06: if View is StreetView and Size is Large and NumberOfBedrooms is ThreeBedrooms and Facilities is Gym and TransportationAccess is None then Pric
0.000
0.000
0.000
0.000
0.000
0.789

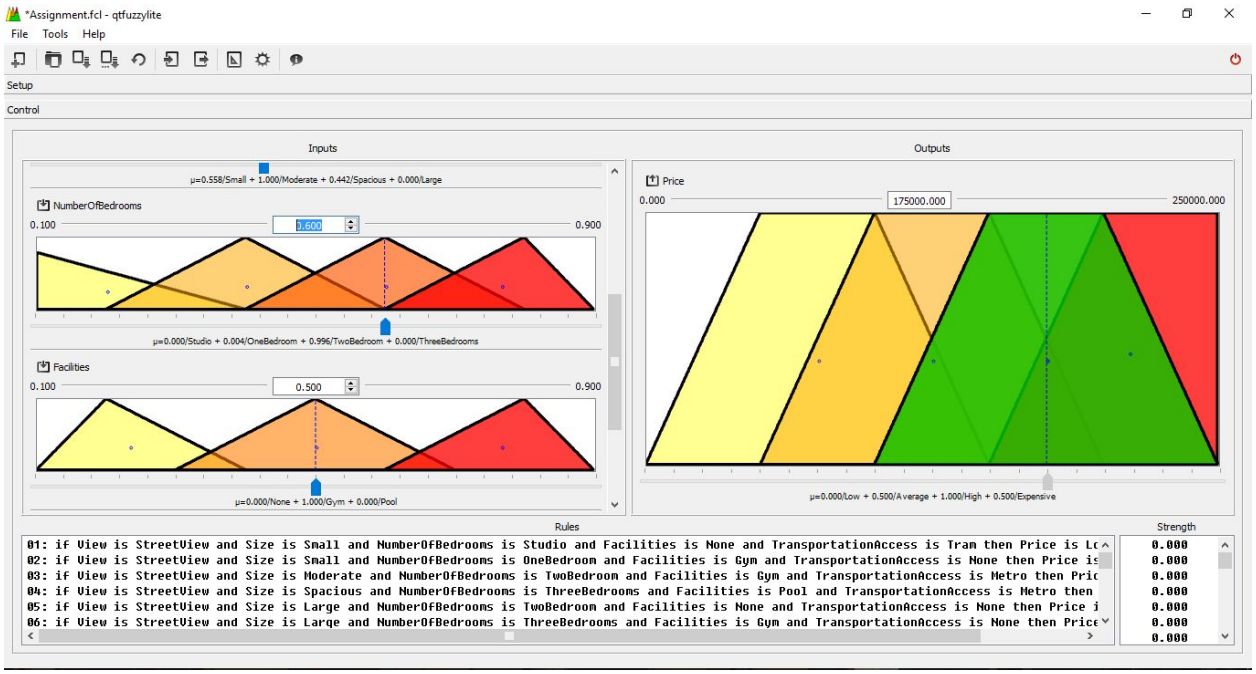
Rule 8


*Assignment.fcl - qtfuzzylite
File Tools Help
Setup
Control
Inputs
μ=0.000/Studio + 1.000/OneBedroom + 0.000/TwoBedroom + 0.000/ThreeBedrooms
Facilities
0.100
0.200
0.900
μ=1.000/None + 0.000/Gym + 0.000/Pool
TransportationAccess
0.100
0.500
0.900
μ=0.000/None + 1.000/Tram + 0.000/Metro
Outputs
Price
0.000
75000.000
250000.000
μ=1.000/Low + 0.500/Average + 0.000/High + 0.000/Expensive
Rules
Strength
01: if View is StreetView and Size is Small and NumberOfBedrooms is Studio and Facilities is None and TransportationAccess is Tram then Price is Lo
02: if View is StreetView and Size is Small and NumberOfBedrooms is OneBedroom and Facilities is Gym and TransportationAccess is None then Price is
03: if View is StreetView and Size is Moderate and NumberOfBedrooms is TwoBedroom and Facilities is Gym and TransportationAccess is Metro then Pric
04: if View is StreetView and Size is Spacious and NumberOfBedrooms is ThreeBedrooms and Facilities is Pool and TransportationAccess is Metro then
05: if View is StreetView and Size is Large and NumberOfBedrooms is TwoBedroom and Facilities is None and TransportationAccess is None then Price i
06: if View is StreetView and Size is Large and NumberOfBedrooms is ThreeBedrooms and Facilities is Gym and TransportationAccess is None then Price
0.000
0.000
0.000
0.000
0.000
0.000
0.000

Rule 9

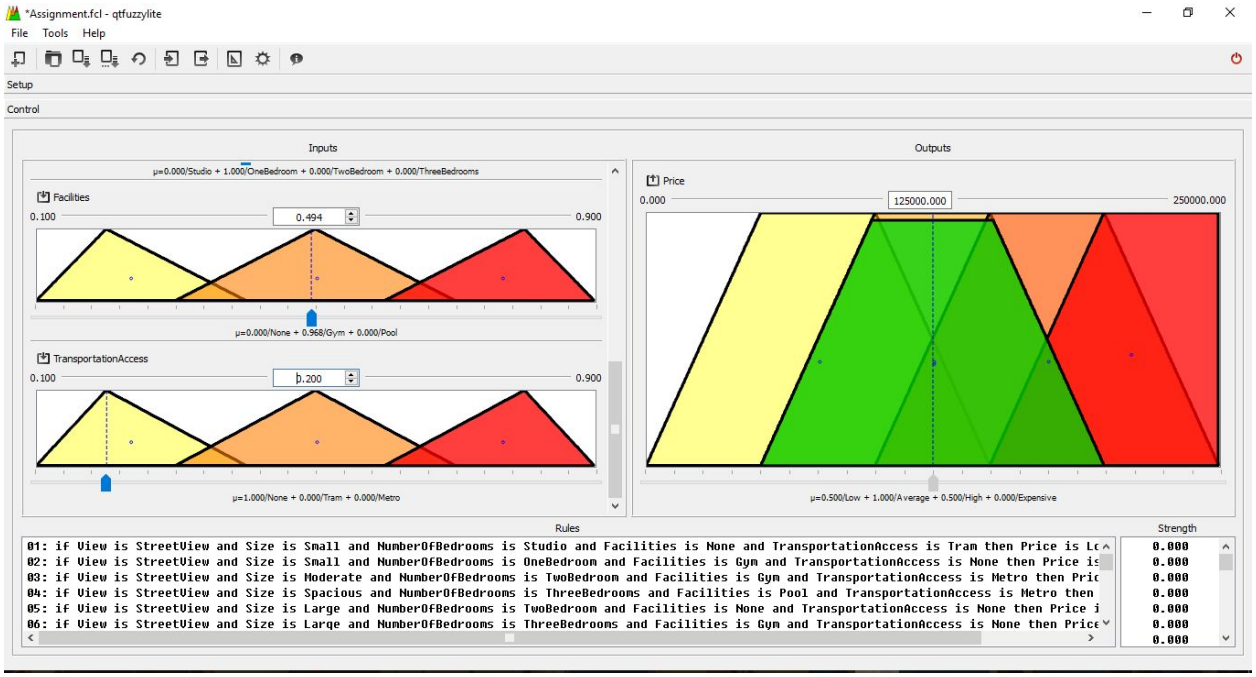


Rule 10

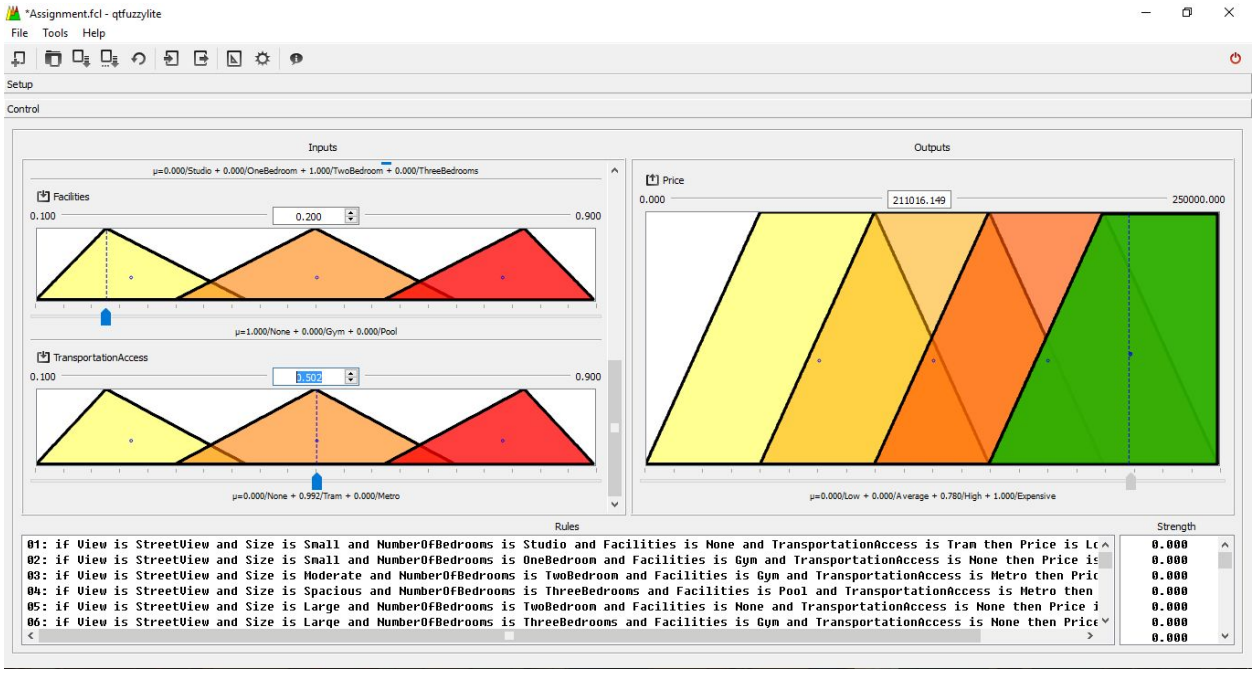


Rule 11

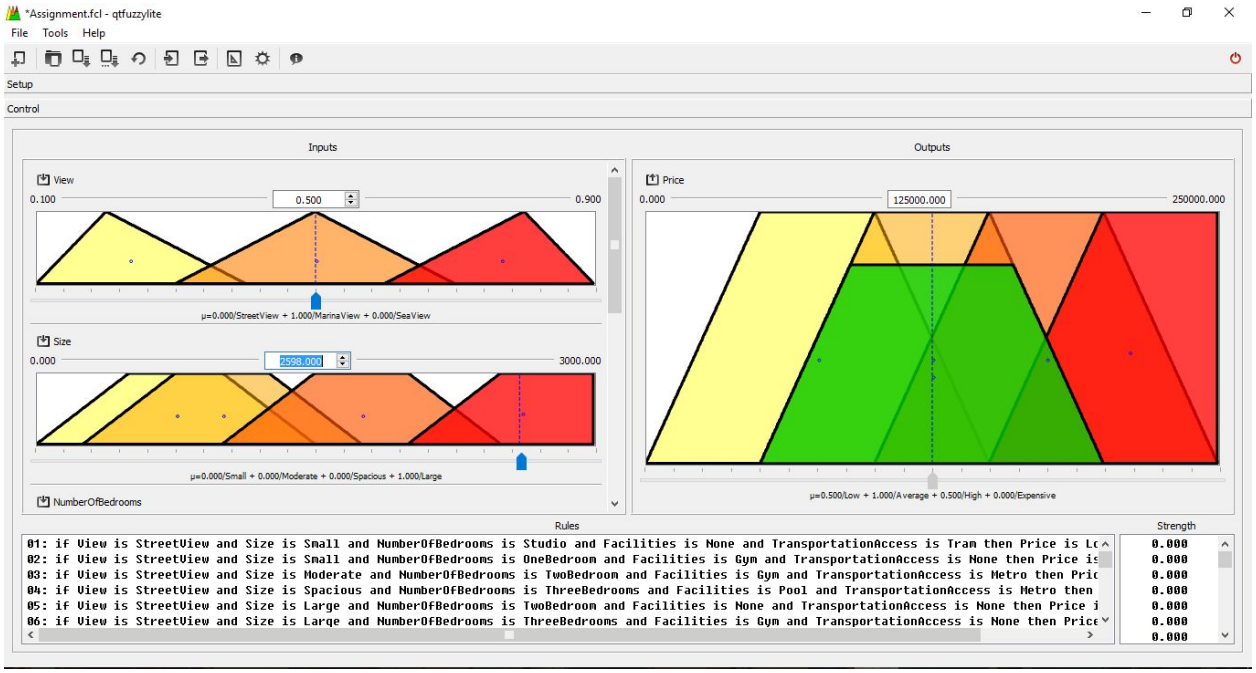
Rule 12



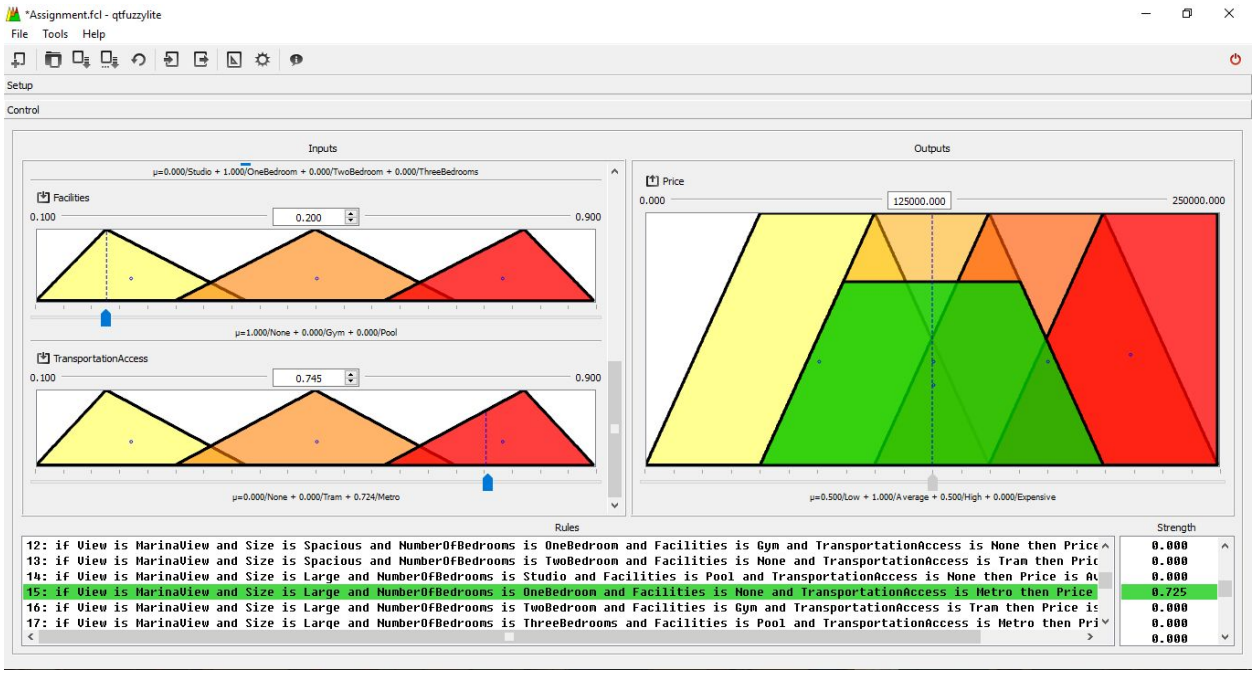
Rule 13



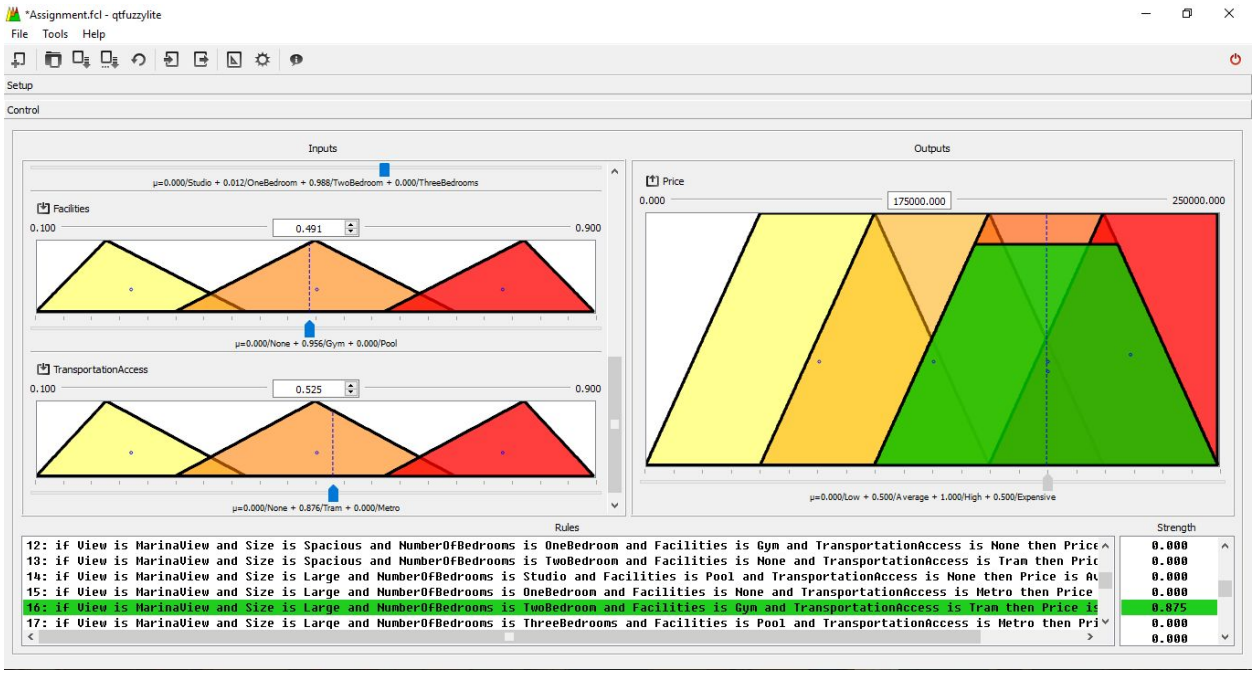
Rule 14



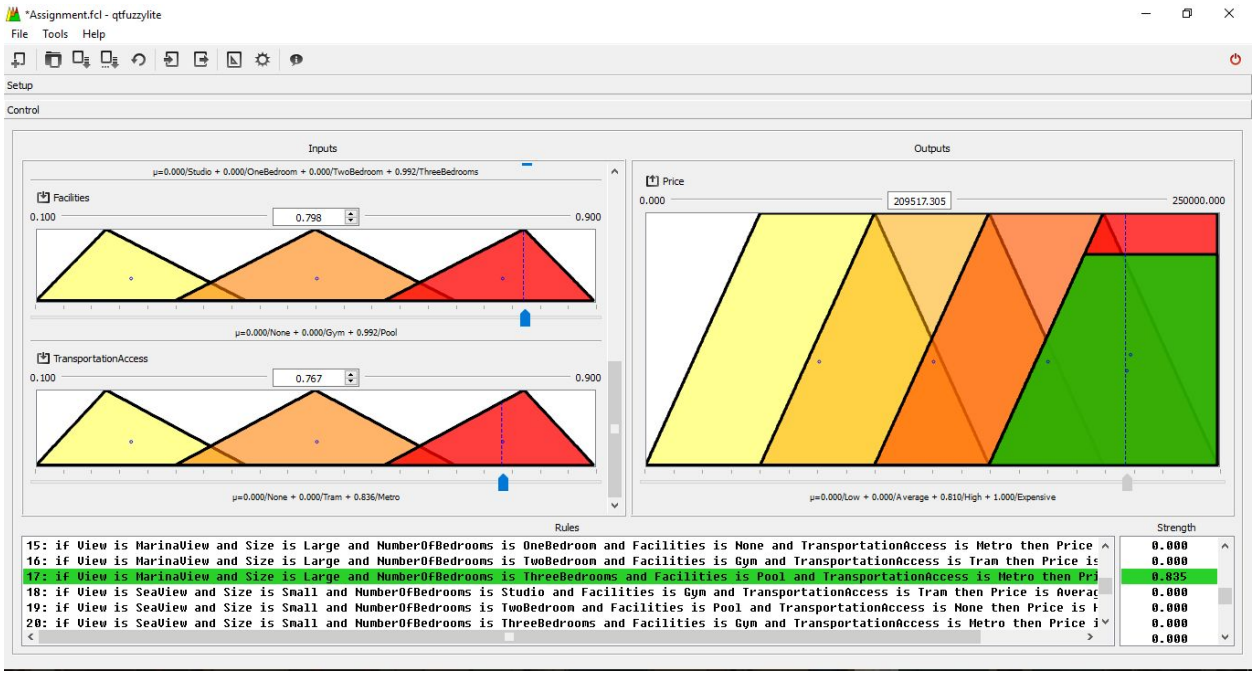
Rule 15



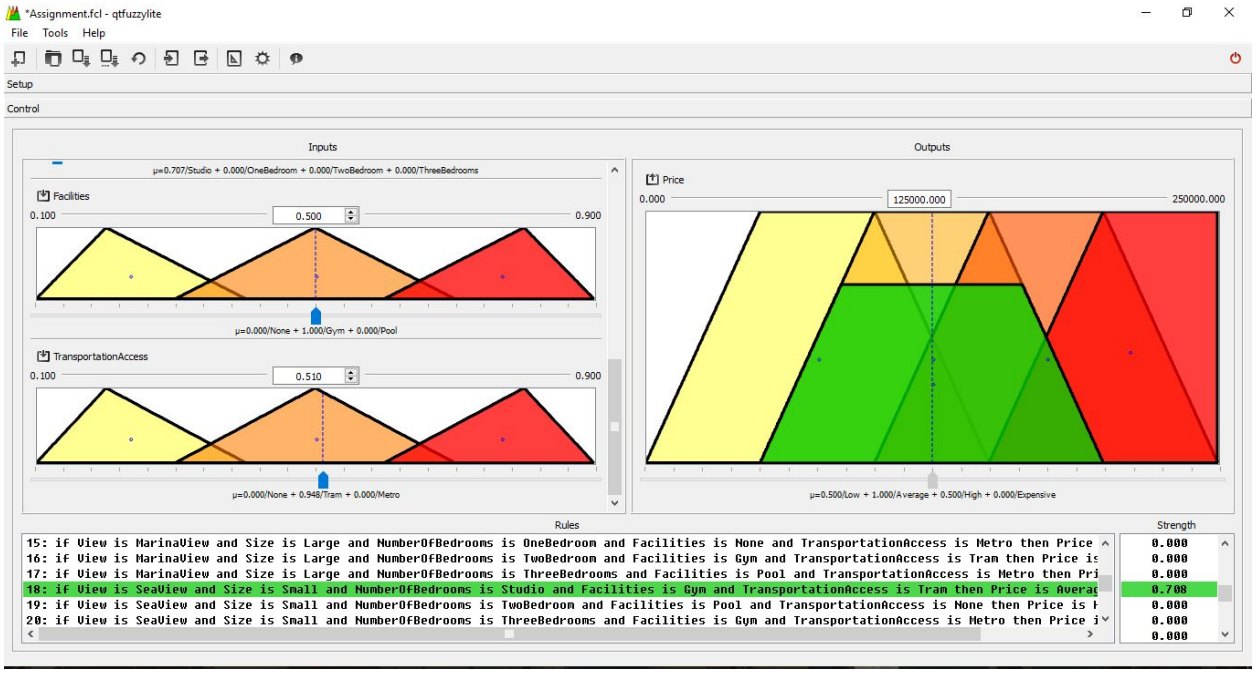
Rule 16



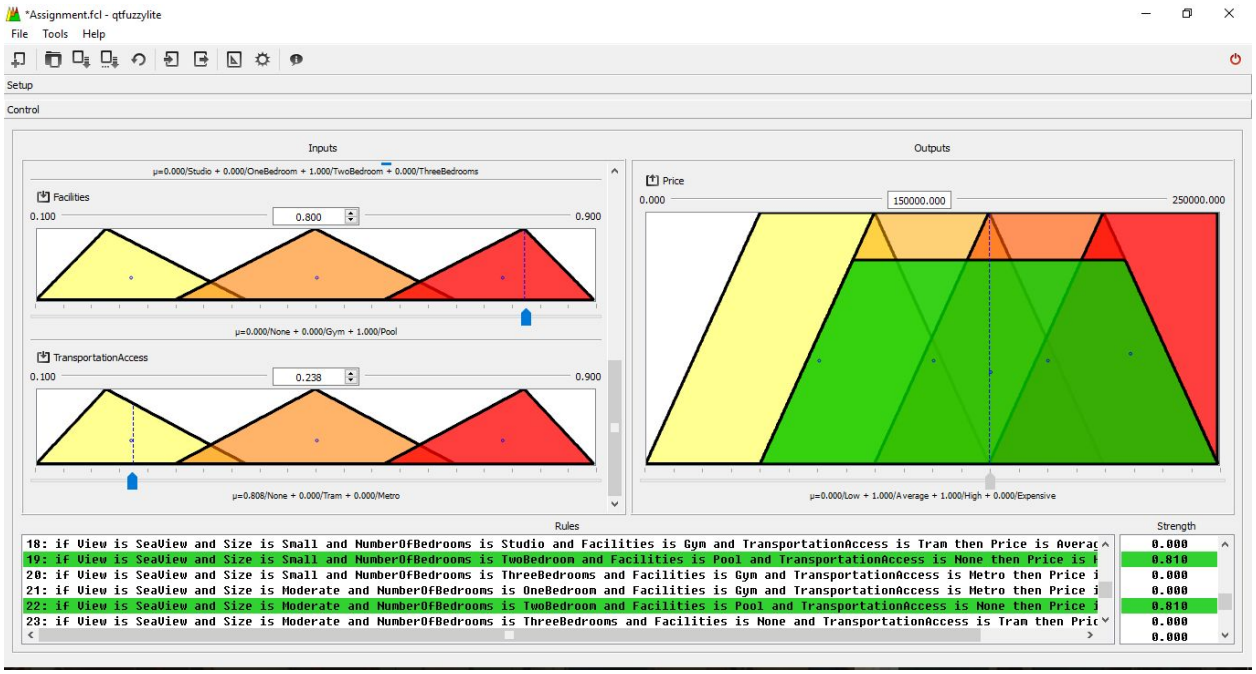
Rule 17



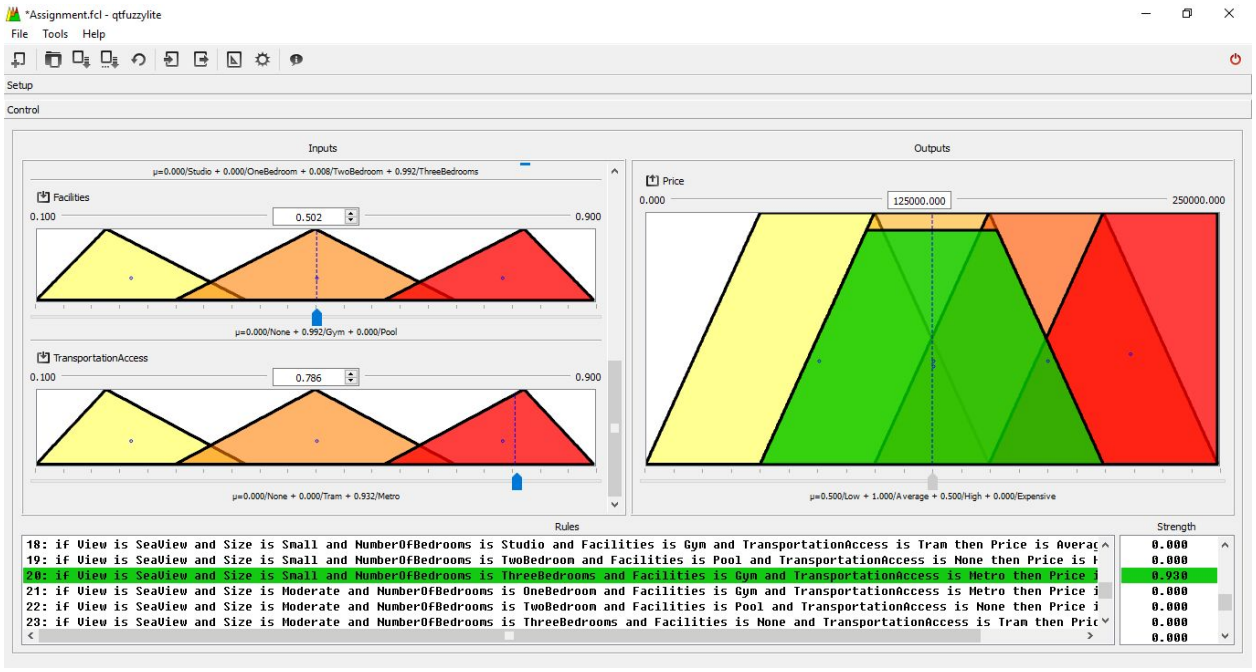
Rule 18



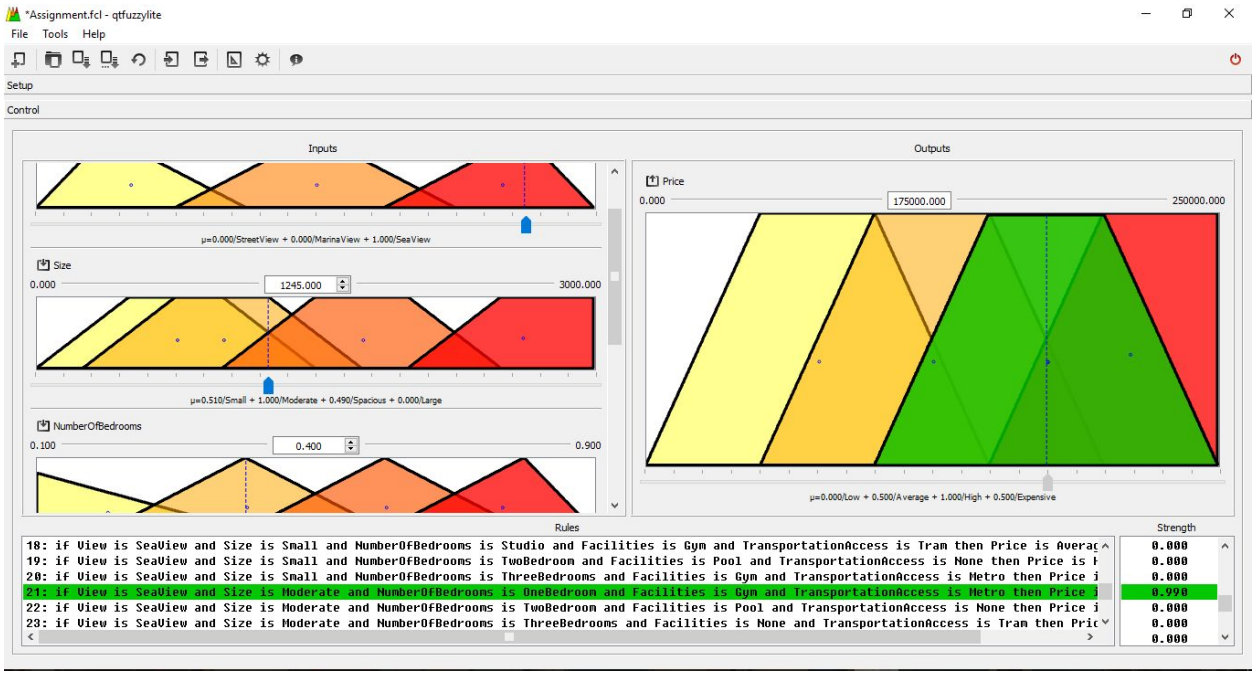
Rule 19



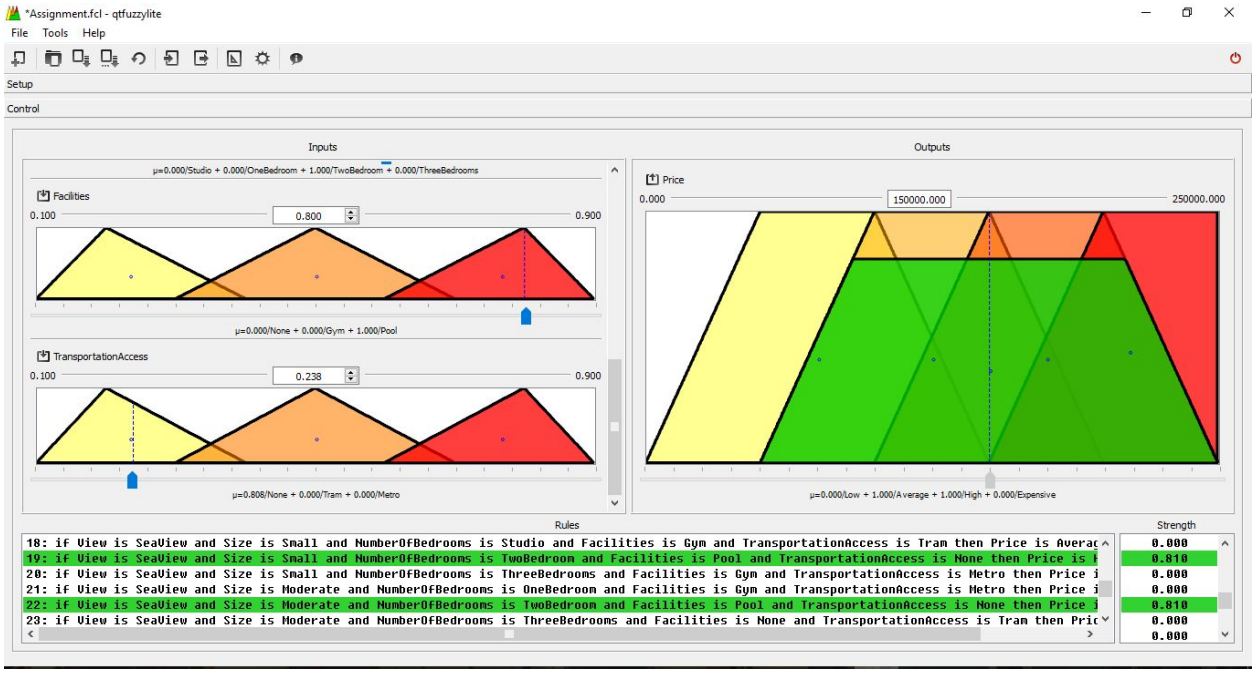
Rule 20



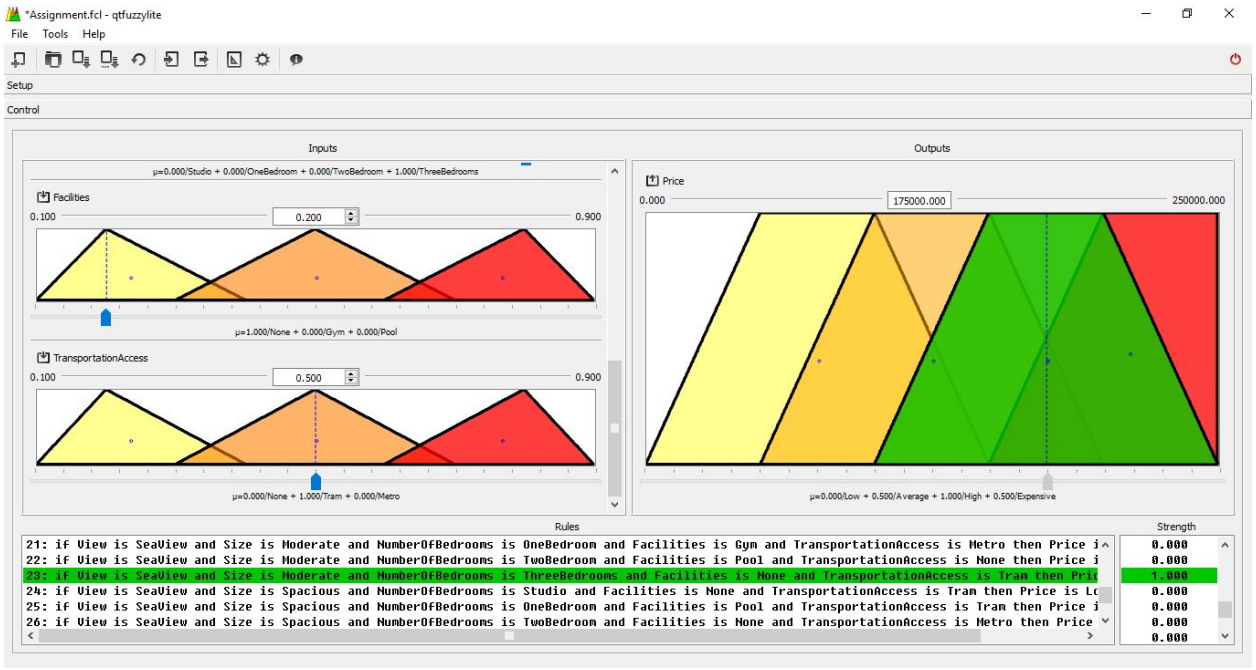
Rule 21



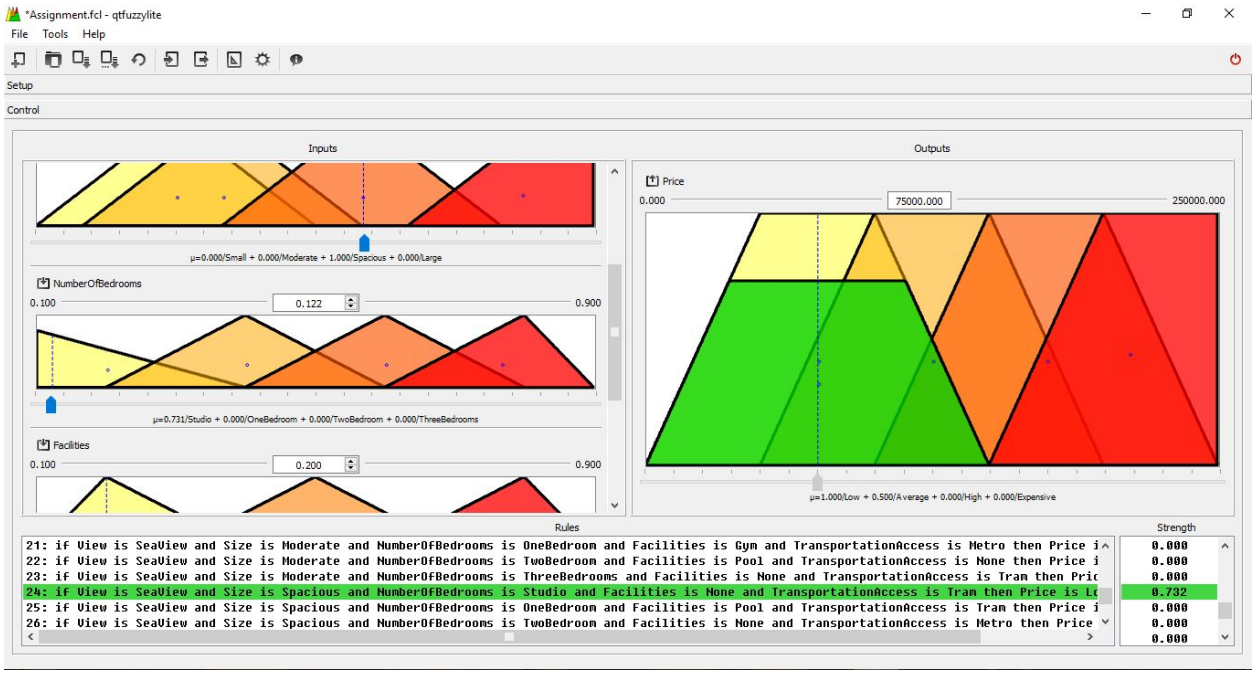
Rule 22



Rule 23

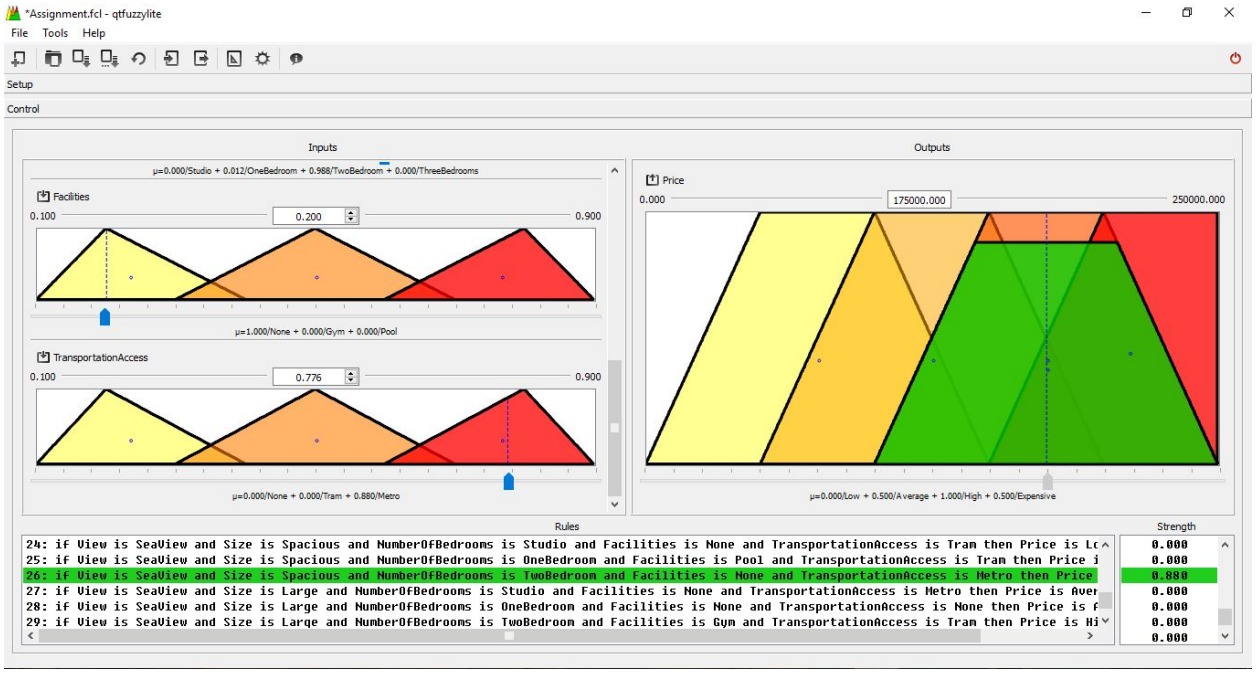


Rule 24

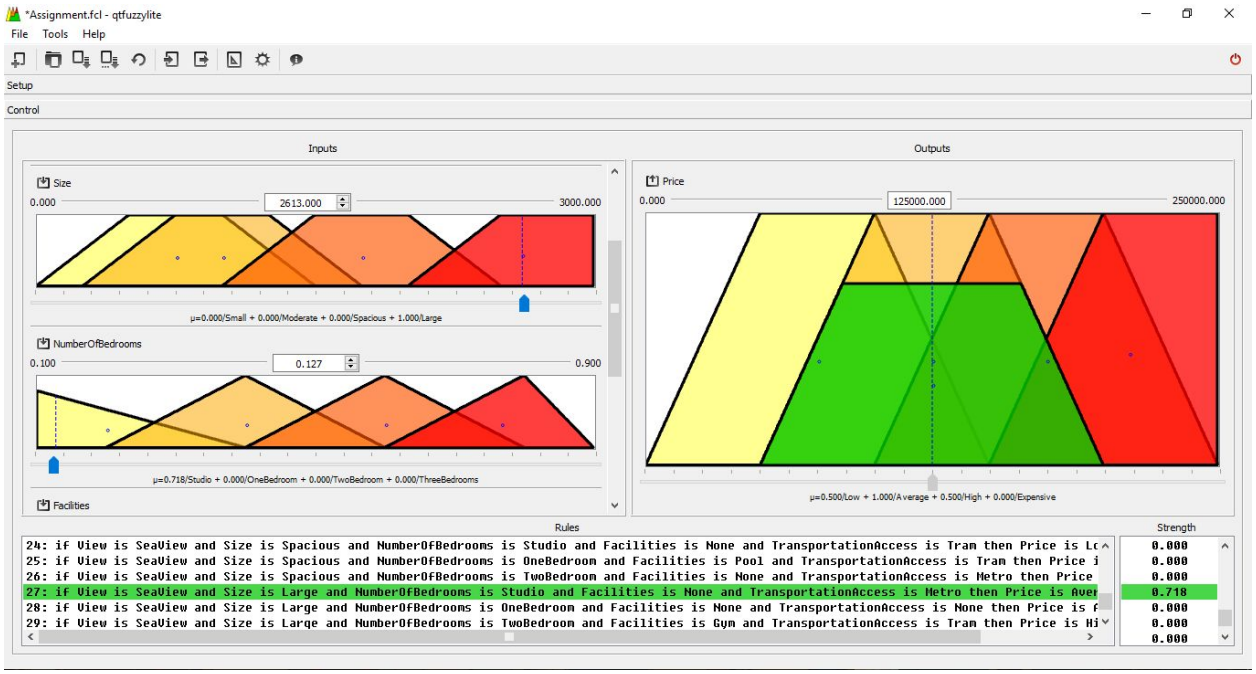


Rule 25

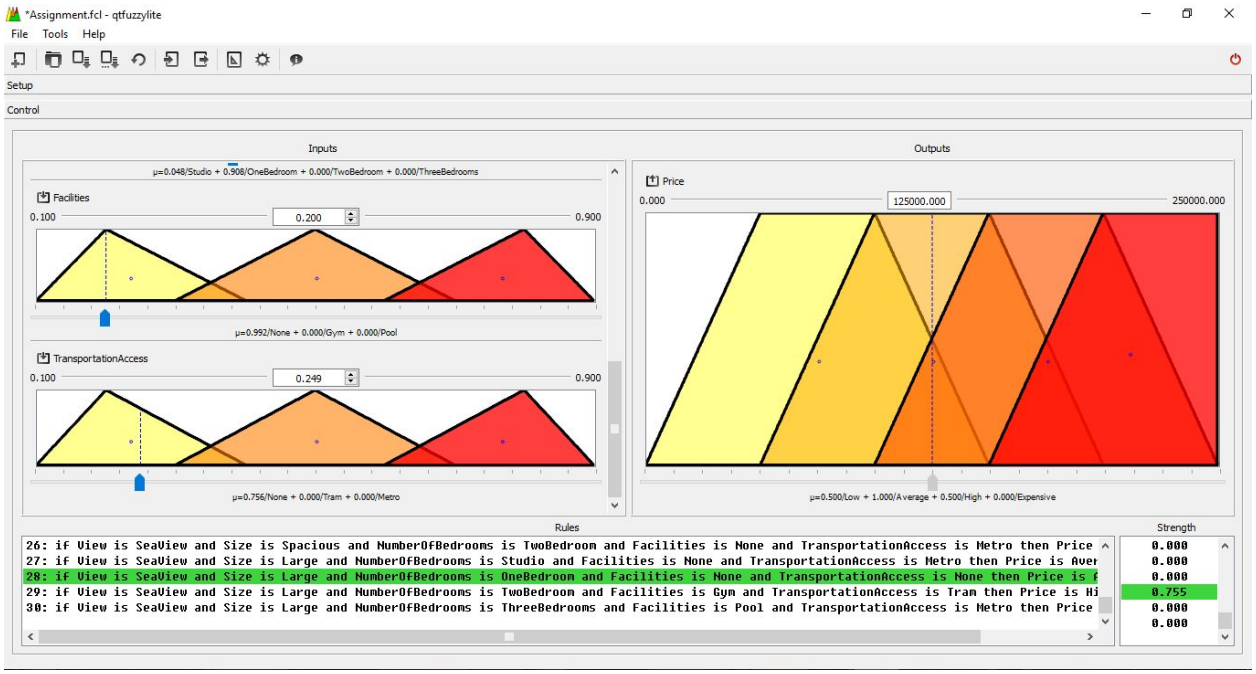
Rule 26



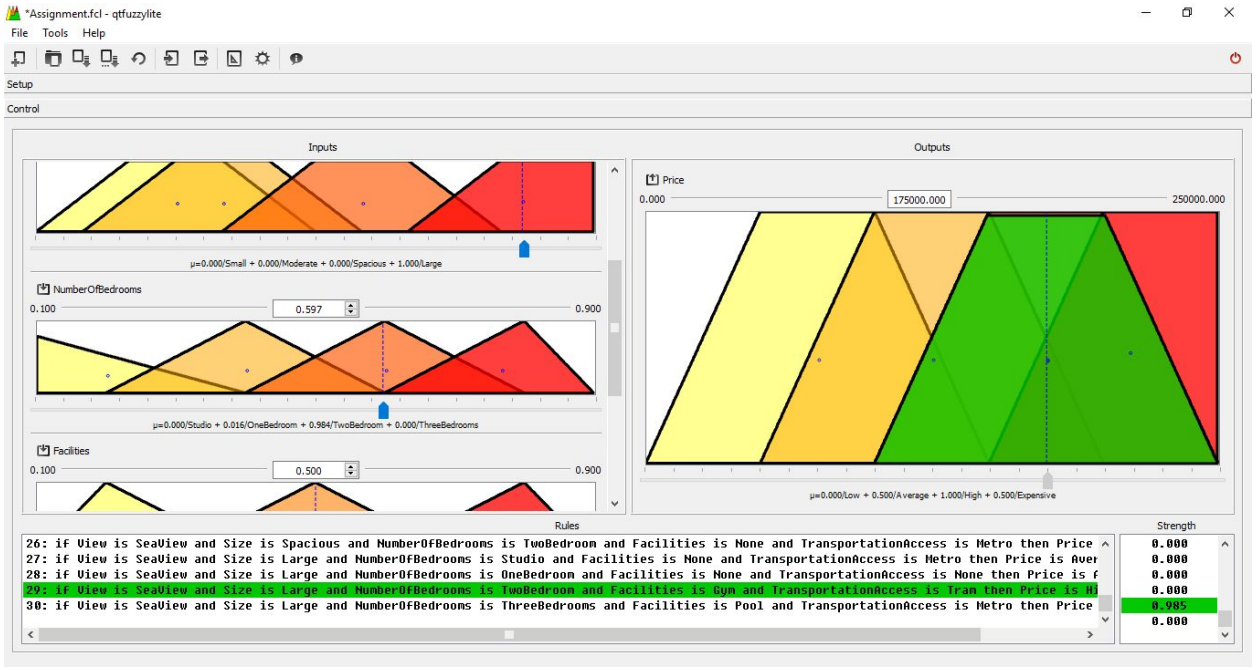
Rule 27



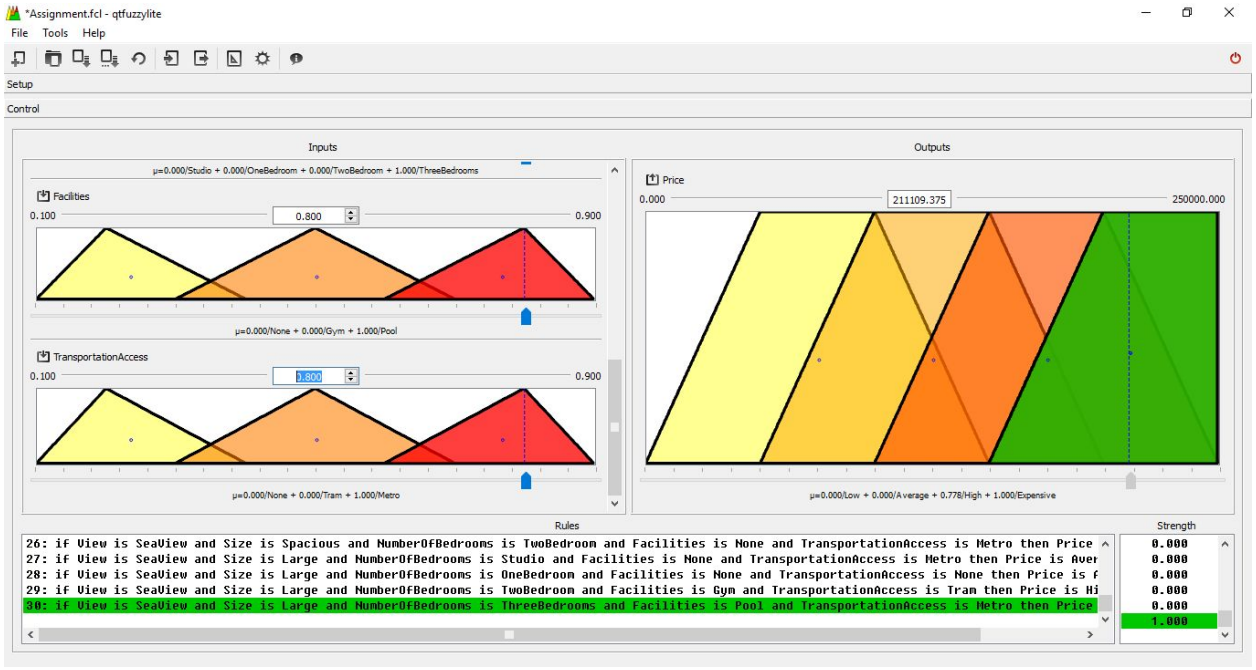
Rule 28



Rule 29



Rule 30



The Program

The program was implemented using C++, using the Code::Blocks Integrated IDE.

Data Structures Used

- A two-dimensional array to store the size lookup table.
- An array to store the crisp values.
- A rule vector to hold the results of the min() or max() values depending on the rules.
- A crisp vector to hold the crisp values of the corresponding rule values from the rule vector.
- A vector to hold the products of the rule and crisp values.

The Algorithm

1. First, the input variables to be fuzzified are inputted from the user.
2. Based upon the input, the values will become fuzzified. For the discrete values it is hardcoded, and for the continuous values, the lookup table is consulted to determine the membership values.
3. Based on these values, the crisp values are determined, and stored into the crispvalues array.
4. The Max and Min objects are called to determine the instantiate the rules, which are stored into a rule vector. For each rule instantiated, the corresponding crisp value is pushed into the crisp vector.
5. For each element in the two vectors, the rule result and the crisp values are multiplied together and stored into a product vector.
6. The price is determined by dividing the sum of the elements in the product vector, by the sum of the rule values, and the price is outputted the user.

Code

Max.cpp

```
#include "Max.h"

Max::Max()
{
    //ctor
}

float Max::max(float a, float b)
{
    if(a > b)
        return a;
    return b;
}

float Max::max(float a, float b, float c)
{
    float arr[3] = {a, b, c};
    return maxarr(arr, 3);
}

float Max::maxarr(float arr[], int size)
{
    float max = arr[0];
    for(int i = 1; i < size; i++)
    {
        if(arr[i] > max)
            max = arr[i];
    }

    return max;
}
```

Min.cpp

```
#include "Min.h"

Min::Min()
{
   //ctor
}

float Min::min(float a, float b)
{
   if(a < b)
      return a;
   return b;
}

float Min::min(float a, float b, float c)
{
   float arr[3] = {a, b, c};
   return minarr(arr, 3);
}

float Min::min(float a, float b, float c, float d)
{
   float arr[4] = {a, b, c, d};
   return minarr(arr, 4);
}

float Min::min(float a, float b, float c, float d, float e)
{
   float arr[5] = {a, b, c, d, e};
   return minarr(arr, 5);
}

float Min::minarr(float arr[], int size)
{
   float min = arr[0];
   for(int i = 1; i < size; i++)
```

```
    {
        if(arr[i] < min)
            min = arr[i];
    }

    return min;
}
```

Main.cpp

```
#include <iostream>
#include <vector> //for crisp values and rule vectors
#include <Max.h>
#include <Min.h>
using namespace std;

float average(float, float);
float average(float, float, float);

int main()
{
    cout << "****The Apartment Price Calculator****" << endl << endl << endl;

    string fuzzyview, fuzzybed, fuzzytrans, fuzzyfac;
    float fuzzysize;
    char choice;

    do
    {
        //user enters details of apartment
        cout << "Please enter the View: " << endl;
        cin >> fuzzyview;
        cout << "Please enter the Size (in square feet): " << endl;
        cin >> fuzzysize;
        cout << "Please enter the Number of Bedrooms: " << endl;
        cin >> fuzzybed;
        cout << "Please enter the Transportation Access: " << endl;
        cin >> fuzzytrans;
        cout << "Please enter the Facilities:" << endl;
        cin >> fuzzyfac;
```

```
	//code for creating tables, populating them

	float sizetable[13][5] = {{0, 0, 0, 0, 0}, {250, 0.5, 0, 0, 0}, {500, 1, 0, 0, 0}, {750, 1, 0.5, 0, 0},
{1000, 1, 1, 0, 0}, \
	{1250, 0.5, 1, 0.5, 0}, {1500, 0, 1, 1, 0}, {1750, 0, 0.5, 1, 0}, {2000, 0, 0, 1, 0}, {2250, 0, 0,
0.5, 0.5}, {2500, 0, 0, 0, 1}, \
	{2750, 0, 0, 0, 1}, {3000, 0, 0, 0, 1}};

	//code for membership attributes (aka fuzzification)
	cout << endl << endl << "fuzzifying..." << endl << endl << endl;

	//view
	float mstreet, mmarina, msea;
	if(fuzzyview == "street")
	{
	   mstreet = 1;
	   mmarina = 0;
	   msea = 0;
	}
	else if(fuzzyview == "marina")
	{
	   mstreet = 0;
	   mmarina = 1;
	   msea = 0;
	}
	else if(fuzzyview == "sea")
	{
	   mstreet = 0;
	   mmarina = 0;
	   msea = 1;
	}

	//size
	float msmall, mmoderate, mspacious, mlarge;
	for (int i = 0; i < 13; i++)
	{
	   if(fuzzysize == sizetable[i][0])
	   {
	      msmall = sizetable[i][1];
	      mmoderate = sizetable[i][2];
	      mspacious = sizetable[i][3];
	      mlarge = sizetable[i][4];
	   }
```

```
else
{
   if(fuzzysize > 0 && fuzzysize < 250)
   {
      msmall = average(sizetable[0][1], sizetable[1][1]);
      mmoderate = average(sizetable[0][2], sizetable[1][2]);
      mspacious = average(sizetable[0][3], sizetable[1][3]);
      mlarge = average(sizetable[0][4], sizetable[1][4]);
   }
   else if(fuzzysize > 250 && fuzzysize < 500)
   {
      msmall = average(sizetable[1][1], sizetable[2][1]);
      mmoderate = average(sizetable[1][2], sizetable[2][2]);
      mspacious = average(sizetable[1][3], sizetable[2][3]);
      mlarge = average(sizetable[1][4], sizetable[2][4]);
   }
   else if(fuzzysize > 500 && fuzzysize < 750)
   {
      msmall = average(sizetable[2][1], sizetable[3][1]);
      mmoderate = average(sizetable[2][2], sizetable[3][2]);
      mspacious = average(sizetable[2][3], sizetable[3][3]);
      mlarge = average(sizetable[2][4], sizetable[3][4]);
   }
   else if(fuzzysize > 750 && fuzzysize < 1000)
   {
      msmall = average(sizetable[3][1], sizetable[4][1]);
      mmoderate = average(sizetable[3][2], sizetable[4][2]);
      mspacious = average(sizetable[3][3], sizetable[4][3]);
      mlarge = average(sizetable[3][4], sizetable[4][4]);
   }
   else if(fuzzysize > 1000 && fuzzysize < 1250)
   {
      msmall = average(sizetable[4][1], sizetable[5][1]);
      mmoderate = average(sizetable[4][2], sizetable[5][2]);
      mspacious = average(sizetable[4][3], sizetable[5][3]);
      mlarge = average(sizetable[4][4], sizetable[5][4]);
   }
   else if(fuzzysize > 1250 && fuzzysize < 1500)
   {
      msmall = average(sizetable[5][1], sizetable[6][1]);
      mmoderate = average(sizetable[5][2], sizetable[6][2]);
      mspacious = average(sizetable[5][3], sizetable[6][3]);
      mlarge = average(sizetable[5][4], sizetable[6][4]);
```

```
}
else if(fuzzysize > 1500 && fuzzysize < 1750)
{
   msmall = average(sizetable[6][1], sizetable[7][1]);
   mmoderate = average(sizetable[6][2], sizetable[7][2]);
   mspacious = average(sizetable[6][3], sizetable[7][3]);
   mlarge = average(sizetable[6][4], sizetable[7][4]);
}
else if(fuzzysize > 1750 && fuzzysize < 2000)
{
   msmall = average(sizetable[7][1], sizetable[8][1]);
   mmoderate = average(sizetable[7][2], sizetable[8][2]);
   mspacious = average(sizetable[7][3], sizetable[8][3]);
   mlarge = average(sizetable[7][4], sizetable[8][4]);
}
else if(fuzzysize > 2000 && fuzzysize < 2250)
{
   msmall = average(sizetable[8][1], sizetable[9][1]);
   mmoderate = average(sizetable[8][2], sizetable[9][2]);
   mspacious = average(sizetable[8][3], sizetable[9][3]);
   mlarge = average(sizetable[8][4], sizetable[9][4]);
}
else if(fuzzysize > 2250 && fuzzysize < 2500)
{
   msmall = average(sizetable[9][1], sizetable[10][1]);
   mmoderate = average(sizetable[9][2], sizetable[10][2]);
   mspacious = average(sizetable[9][3], sizetable[10][3]);
   mlarge = average(sizetable[9][4], sizetable[10][4]);
}
else if(fuzzysize > 2500 && fuzzysize < 2750)
{
   msmall = average(sizetable[10][1], sizetable[11][1]);
   mmoderate = average(sizetable[10][2], sizetable[11][2]);
   mspacious = average(sizetable[10][3], sizetable[11][3]);
   mlarge = average(sizetable[10][4], sizetable[11][4]);
}
else if(fuzzysize > 2750 && fuzzysize < 3000)
{
   msmall = average(sizetable[11][1], sizetable[12][1]);
   mmoderate = average(sizetable[11][2], sizetable[12][2]);
   mspacious = average(sizetable[11][3], sizetable[12][3]);
   mlarge = average(sizetable[11][4], sizetable[12][4]);
}
```

```
    }
}

//number of bedrooms
float mstudio, mone, mtwo, mthree;
if(fuzzybed == "studio")
{
    mstudio = 1;
    mone = 0;
    mtwo = 0;
    mthree = 0;
}
else if(fuzzybed == "one")
{
    mstudio = 0;
    mone = 1;
    mtwo = 0;
    mthree = 0;
}
else if(fuzzybed == "two")
{
    mstudio = 0;
    mone = 0;
    mtwo = 1;
    mthree = 0;
}
else if(fuzzybed == "three")
{
    mstudio = 0;
    mone = 0;
    mtwo = 0;
    mthree = 1;
}

//facilities
float mnone, mgym, mpool;
if(fuzzyfac == "none")
{
   mnone = 1;
   mgym = 0;
   mpool = 0;
}
else if(fuzzyfac == "gym")
```

```
{
  mnone = 0;
  mgym = 1;
  mpool = 0;
}
else if(fuzzyfac == "pool")
{
  mnone = 0;
  mgym = 0;
  mpool = 1;
}

//transportation access
float mtransnone, mtram, mmetro;
if(fuzzytrans == "none")
{
  mtransnone = 1;
  mtram = 0;
  mmetro = 0;
}
else if(fuzzytrans == "tram")
{
  mtransnone = 0;
  mtram = 1;
  mmetro = 0;
}
else if(fuzzytrans == "metro")
{
  mtransnone = 0;
  mtram = 0;
  mmetro = 1;
}


//code for determining the crisp values (from the price vector)

cout << "crispifying..." << endl << endl << endl;

float crispvalues[4];
crispvalues[0] = average(50000, 75000, 100000);
crispvalues[1] = average(100000, 125000, 150000);
crispvalues[2] = average(150000, 175000, 200000);
crispvalues[3] = average(200000, 225000, 250000);
```

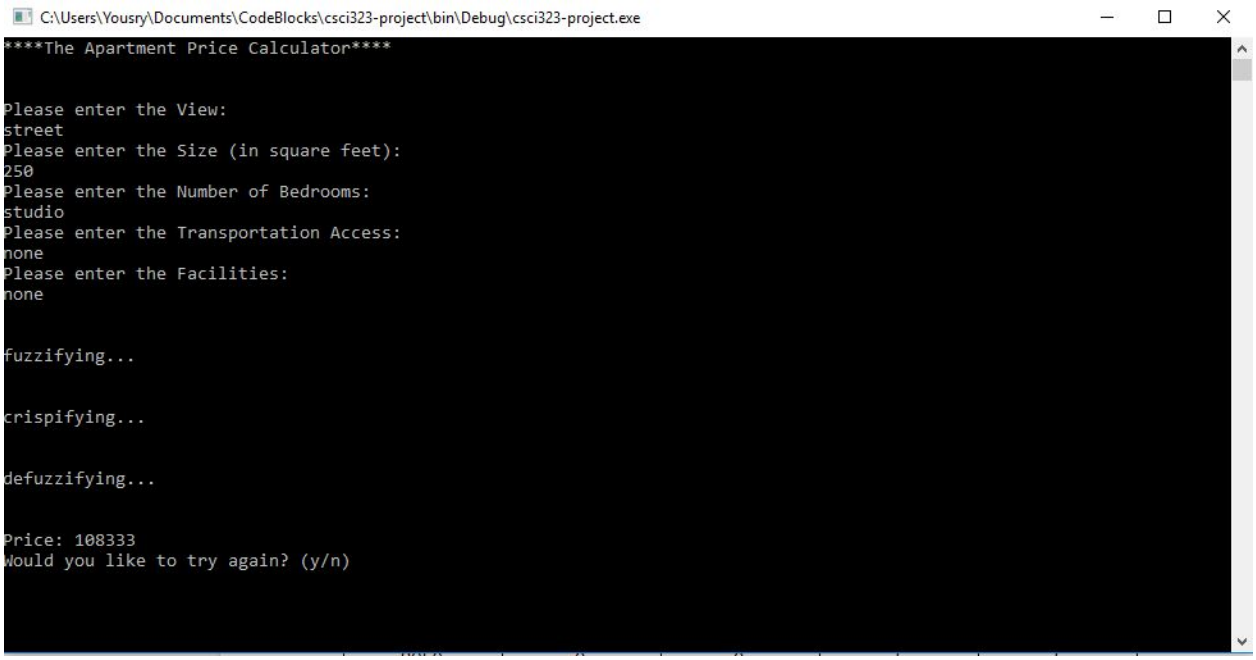
```
vector<float> rulevector; // a vector to store the result of applying the rules
vector<float> crispvector; // a vector to store the respective crisp values

Max max; // to use the max functions
Min min; // to use the min functions

float rule1 = min.min(mstudio, msmall);
rulevector.push_back(rule1);
crispvector.push_back(crispvalues[0]);

float rule2 = min.min(msea, mspacious);
rulevector.push_back(rule2);
crispvector.push_back(crispvalues[2]);

float rule3 = min.min(mmoderate, mone);
rulevector.push_back(rule3);
crispvector.push_back(crispvalues[1]);

float rule4 = min.min(mthree, mlarge);
rulevector.push_back(rule4);
crispvector.push_back(crispvalues[3]);

float rule5 = min.min(mmarina, mmoderate);
rulevector.push_back(rule5);
crispvector.push_back(crispvalues[1]);

float rule6 = max.max(mspacious, mtwo);
rulevector.push_back(rule6);
crispvector.push_back(crispvalues[2]);

float rule7 = max.max(mgym, mone);
rulevector.push_back(rule3);
crispvector.push_back(crispvalues[1]);

float rule8 = max.max(mnone, msmall);
rulevector.push_back(rule8);
crispvector.push_back(crispvalues[0]);

float rule9 = max.max(msea, mpool);
rulevector.push_back(rule9);
crispvector.push_back(crispvalues[2]);
```

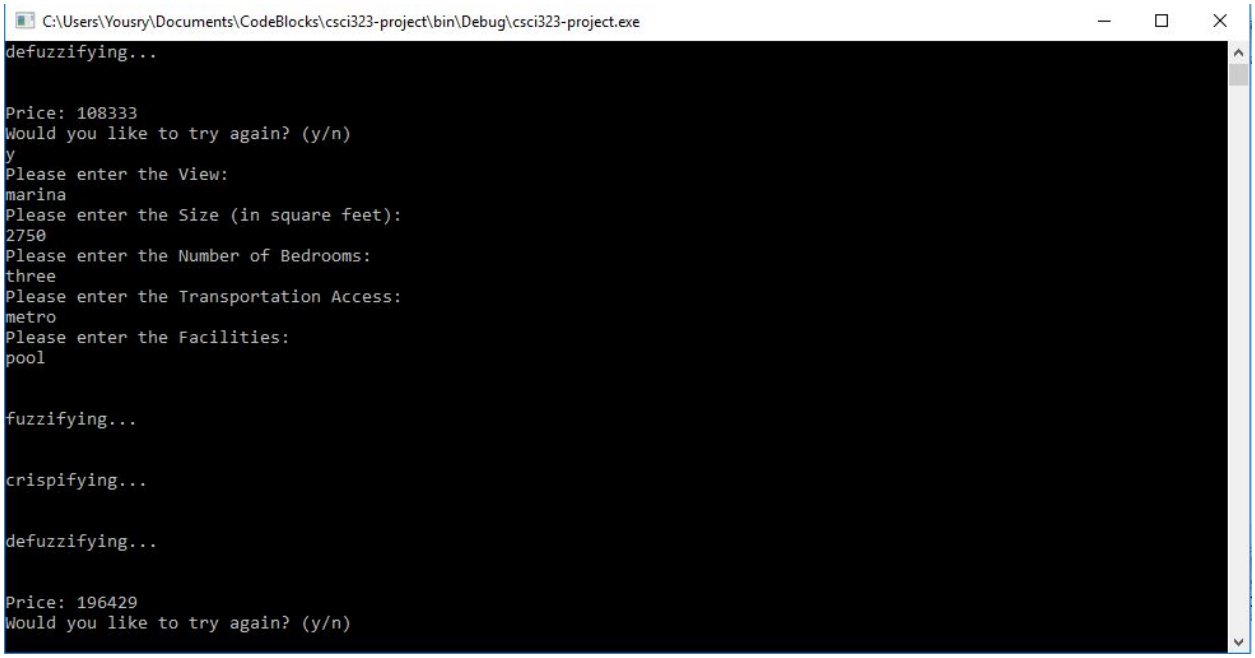
```
float rule10 = max.max(mthree, mlarge);
rulevector.push_back(rule10);
crispvector.push_back(crispvalues[3]);

float rule11 = min.min(mtransnone, mnone);
rulevector.push_back(rule11);
crispvector.push_back(crispvalues[0]);

float rule12 = min.min(mtram, mgym);
rulevector.push_back(rule12);
crispvector.push_back(crispvalues[1]);

float rule13 = min.min(mtwo, mmetro);
rulevector.push_back(rule13);
crispvector.push_back(crispvalues[2]);

float rule14 = min.min(mtwo, mpool);
rulevector.push_back(rule14);
crispvector.push_back(crispvalues[2]);

float rule15 = min.min(mpool, mmetro);
rulevector.push_back(rule15);
crispvector.push_back(crispvalues[2]);

float rule16 = min.min(mpool, mspacious);
rulevector.push_back(rule16);
crispvector.push_back(crispvalues[2]);

float rule17 = max.max(mtwo, mlarge);
rulevector.push_back(rule17);
crispvector.push_back(crispvalues[3]);

float rule18 = max.max(mmarina, mstudio);
rulevector.push_back(rule18);
crispvector.push_back(crispvalues[1]);

float rule19 = min.min(msmall, mstreet, mstudio);
rulevector.push_back(rule19);
crispvector.push_back(crispvalues[0]);

float rule20 = min.min(mmarina, mone, mmoderate);
rulevector.push_back(rule20);
crispvector.push_back(crispvalues[1]);
```

```
float rule21 = min.min(mtwo, mspacious, msea);
rulevector.push_back(rule21);
crispvector.push_back(crispvalues[2]);

float rule22 = min.min(mmarina, mgym, mtram, mone, mmoderate);
rulevector.push_back(rule22);
crispvector.push_back(crispvalues[1]);

float rule23 = min.min(mmetro, mtwo, mpool, mspacious, msea);
rulevector.push_back(rule23);
crispvector.push_back(crispvalues[2]);

float rule24 = min.min(mstreet, mstudio, mtransnone, msmall, mnone);
rulevector.push_back(rule24);
crispvector.push_back(crispvalues[0]);

float rule25 = min.min(mlarge, mstudio);
rulevector.push_back(rule25);
crispvector.push_back(crispvalues[3]);

float rule26 = min.min(mmetro, mtwo, mpool, msea);
rulevector.push_back(rule26);
crispvector.push_back(crispvalues[2]);

float rule27 = min.min(mmarina, mtram, mone, mmoderate);
rulevector.push_back(rule27);
crispvector.push_back(crispvalues[1]);

float rule28 = min.min(mmarina, mgym, mtram, mmoderate);
rulevector.push_back(rule28);
crispvector.push_back(crispvalues[1]);

float rule29 = max.max(mstreet, mlarge, mnone);
rulevector.push_back(rule29);
crispvector.push_back(crispvalues[3]);

float rule30 = min.min(mnone, mstudio, mtransnone, msmall);
rulevector.push_back(rule30);
crispvector.push_back(crispvalues[0]);
```

```
        //code for determining the price based on the result from the rules and the crisp values
(aka defuzzification)
        cout << "defuzzifying..." << endl << endl << endl;

        float sum = 0, sum2 = 0;
        vector<float> prodvector;

        for(int i = 0; i < crispvector.size(); i++)
        {
            prodvector.push_back(rulevector[i] * crispvector[i]);
            sum2 += rulevector[i];
        }

        for(int i = 0; i < prodvector.size(); i++)
        {
            sum += prodvector[i];
        }

        float price = sum/sum2;

        cout << "Price: " << price << endl;

        cout << "Would you like to try again? (y/n)" << endl;
        cin >> choice;

    }while(choice == 'y');

    cout << "Thank you for using this program!" << endl;

    return 0;
}

float average(float a, float b)
{
    return ((a + b)/2);
}

float average(float a, float b, float c)
{
    return ((a + b + c)/3);
}
```

Screenshots




C:\Users\Yousry\Documents\CodeBlocks\csci323-project\bin\Debug\csci323-project.exe
defuzzifying...
Price: 108333
Would you like to try again? (y/n)
y
Please enter the View:
marina
Please enter the Size (in square feet):
2750
Please enter the Number of Bedrooms:
three
Please enter the Transportation Access:
metro
Please enter the Facilities:
pool
fuzzifying...
crispifying...
defuzzifying...
Price: 196429
Would you like to try again? (y/n)