Lab 09 : Fuzzy Logic

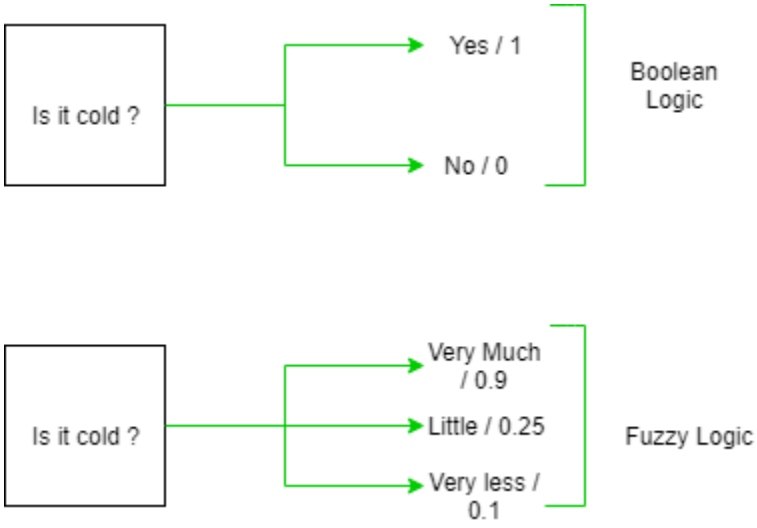
Fuzzy Logic

This is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1, instead of just the traditional values of true or false. It is used to deal with imprecise or uncertain information and is a mathematical method for representing vagueness and uncertainty in decision-making.

Fuzzy Logic is based on the idea that in many cases, the concept of true or false is too restrictive, and that there are many shades of gray in between. It allows for partial truths, where a statement can be partially true or false, rather than fully true or false.
Fuzzy Logic is used in a wide range of applications, such as control systems, image processing, natural language processing, medical diagnosis, and artificial intelligence.

Working

The fundamental concept of Fuzzy Logic is the membership function, which defines the degree of membership of an input value to a certain set or category. The membership function is a mapping from an input value to a membership degree between 0 and 1, where 0 represents non-membership and 1 represents full membership.
Fuzzy Logic is implemented using Fuzzy Rules, which are if-then statements that express the relationship between input variables and output variables in a fuzzy way. The output of a Fuzzy Logic system is a fuzzy set, which is a set of membership degrees for each possible output value.



Implementation in Python

Example problem: Disaster prediction

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

rainfall = ctrl.Antecedent(np.arange(0, 201, 1), 'rainfall')
wind_speed = ctrl.Antecedent(np.arange(0, 151, 1), 'wind_speed')
temperature = ctrl.Antecedent(np.arange(-20, 51, 1), 'temperature')
disaster_risk = ctrl.Consequent(np.arange(0, 101, 1), 'disaster_risk')

rainfall['low'] = fuzz.trimf(rainfall.universe, [0, 0, 50])
rainfall['moderate'] = fuzz.trimf(rainfall.universe, [30, 80, 130])
rainfall['high'] = fuzz.trimf(rainfall.universe, [100, 150, 200])

wind_speed['calm'] = fuzz.trimf(wind_speed.universe, [0, 0, 30])
wind_speed['breezy'] = fuzz.trimf(wind_speed.universe, [20, 60, 100])
wind_speed['stormy'] = fuzz.trimf(wind_speed.universe, [80, 120, 150])

temperature['cold'] = fuzz.trimf(temperature.universe, [-20, -10, 10])
temperature['mild'] = fuzz.trimf(temperature.universe, [0, 20, 30])
temperature['hot'] = fuzz.trimf(temperature.universe, [25, 35, 50])

disaster_risk['low'] = fuzz.trimf(disaster_risk.universe, [0, 0, 40])
disaster_risk['moderate'] = fuzz.trimf(disaster_risk.universe, [30, 50, 70])
disaster_risk['high'] = fuzz.trimf(disaster_risk.universe, [60, 100, 100])

rule1 = ctrl.Rule(rainfall['low'] & wind_speed['calm'] & temperature['mild'], disaster_risk['low'])
rule2 = ctrl.Rule(rainfall['moderate'] & wind_speed['breezy'] & temperature['mild'],
disaster_risk['moderate'])
rule3 = ctrl.Rule(rainfall['high'] & wind_speed['stormy'] & temperature['hot'],
disaster_risk['high'])
rule4 = ctrl.Rule(rainfall['high'] & wind_speed['stormy'], disaster_risk['high'])
rule5 = ctrl.Rule(rainfall['moderate'] & wind_speed['stormy'], disaster_risk['moderate'])
rule6 = ctrl.Rule(rainfall['low'] & wind_speed['calm'] & temperature['cold'], disaster_risk['low'])

disaster_ctrl = ctrl.ControlSystem([rule1, rule2, rule3, rule4, rule5, rule6])
disaster_simulation = ctrl.ControlSystemSimulation(disaster_ctrl)
```

```
disaster_simulation.input['rainfall'] = 120
disaster_simulation.input['wind_speed'] = 90
disaster_simulation.input['temperature'] = 35

disaster_simulation.compute()

print(f"Disaster Risk Level: {disaster_simulation.output['disaster_risk']:.2f}%")
```

Output

Input Metrics given: rainfall = 120mm, wind speed = 90km/h, and temperature = 35°C

Disaster Risk Level: 75.00%