A MINI PROJECT REPORT ON

“ASSETS VALUATION USING FUZZY LOGIC”

Submitted to

Department of Computer Engineering

BY

SUSMIT DESHPANDE (Roll No 110)
ROHIT CHAUDHARI (Rol1No 105)
KETAN CHITALE (Rol1 No 107)

UNDER THE GUIDANCE OF MRS.

SANKIRTI SHIRAVALE

MMCOE

Karvenagar, PUNE – 411052

2020-2021

PROBLEM STATEMENT: Implement Assets Valuation System using Fuzzy Logic. In this System, we are taking Applicant's Assets and Income as input for valuating applicant. Also, we are taking Applicant's House Location and Market Value of House for valuating Applicant's House.

HARDWARE AND SOFTWARE REQUIREMENTS:

Hardware Requirements-

1. Ram – 2 GB or more

2. Rom – 500 GB

3. Processor – Intel i3 or higher

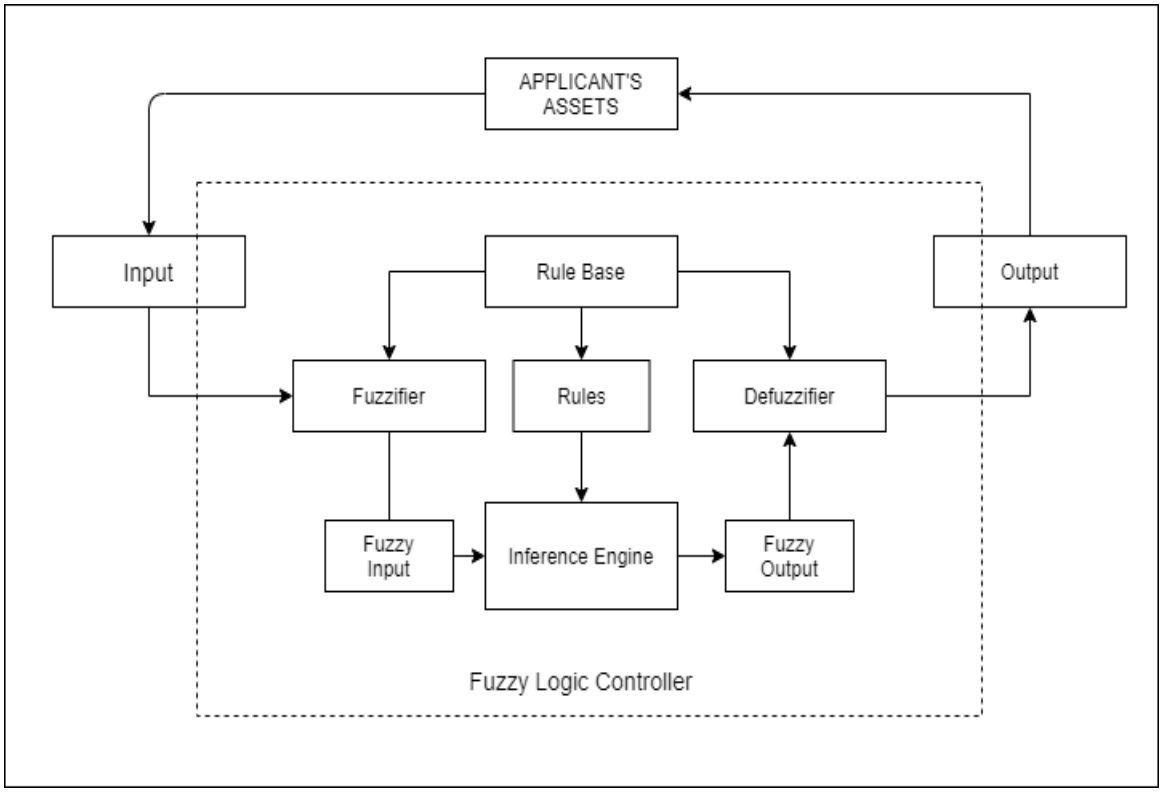
Software Requirements-

1. Python

2. Anaconda Navigator

3. Operating System: Windows/ Ubuntu

SYSTEM ARCHITECTURE DIAGRAM:



LITERATURE SURVEY:

PAPER TITLE	AUTHOR	YEAR	DESCRIPTION
Fuzzy logic approach applied to credit scoring for microfinance in Morocco.	Ghita Bennouna , Mohamed Tkiouat	2018	Fuzzy Logic allows to provide visibility on the behaviour of all customers by referring not only to statistical studies carried out beforehand but also to the opinion of the manager who is in direct contact with the customers. It should be noted that in this context, the result obtained is variable, since the classification of certain inputs varies over time, hence the need to update them or to add decision filters if necessary.
Fuzzy Logic Based Loan Evaluation System	Sadig Mammadli	2016	In this article, a fuzzy logic approach to evaluating retail loans that can be used to describe imprecise knowledge or human subjective judgment by linguistic terms is proposed. The fuzzy expert system provides an accurate evaluation of credit standing, according to subjective assessment,
A Fuzzy Inference System for Credit Scoring using Boolean Consistent Fuzzy Logic	Milica Latinovic, Ivana Dragovic, Vesna Bogojevic Arsic, Bratislav Petrovic	2018	Fuzzy inference system (FIS) allows domain experts to express their knowledge in the form of fuzzy rules, which enables combination of automatic rating with human judgment. Empirical results show that this property generates better results compared to the conventional approach. Accuracy of our model is 73.75%, compared to conventional FIS, which accuracy was 65%.

ALGORITHM:

A] Applicant Valuation :

Step 1: Defining the fuzzy sets and membership functions.

Step 2: From the rule base creating a fuzzy rule base configuration.

Step 3: Accept crisp values for input variables as Persons Income and current assets and apply fuzzification.

Step 4: Apply the rules and obtain decision rule.

Step 5: Apply Defuzzification by using Mean of Maximum method.

Step 6: Display the Applicant's Valuation as output crisp value.

A] House Valuation :

Step 1: Defining the fuzzy sets and membership functions.

Step 2: From the rule base creating a fuzzy rule base configuration.

Step 3: Accept crisp values for input variables as House Location and Market Value and apply fuzzification.

Step 4: Apply the rules and obtain decision rule.

Step 5: Apply Defuzzification by using Mean of Maximum method.

Step 6: Display the House's Valuation as output crisp value.

RULE BASE:

Applicant Evaluation Rule base 1

1. If (Asset is Low) and (Income is Low) then (Applicant is Low)
   - (Asset == Low AND Income == Low) AND Applicant == Low ==> C1
2. If (Asset is Low) and (Income is Medium) then (Applicant is Low)
   - (Asset == Low AND Income == Medium) AND Applicant == Low ==> C2
3. If (Asset is Low) and (Income is High) then (Applicant is Medium)
   - (Asset == Low AND Income == High) AND Applicant == Medium ==> C3
4. If (Asset is Low) and (Income is Very_high) then (Applicant is High)
   - (Asset == Low AND Income == Very_high) AND Applicant == High ==> C4
5. If (Asset is Medium) and (Income is Low) then (Applicant is Low)
   - (Asset == Medium AND Income == Low) AND Applicant == Low ==> C5
6. If (Asset is Medium) and (Income is Medium) then (Applicant is Medium)
   - (Asset == Medium AND Income == Medium) AND Applicant == Medium ==> C6
7. If (Asset is Medium) and (Income is High) then (Applicant is High)
   - (Asset == Medium AND Income == High) AND Applicant == High ==> C7
8. If (Asset is Medium) and (Income is Very_high) then (Applicant is High)
   - (Asset == Medium AND Income == Very_high) AND Applicant == High ==> C8
9. If (Asset is High) and (Income is Low) then (Applicant is Medium)
   - (Asset == High AND Income == Low) AND Applicant == Medium ==> C9
10. If (Asset is High) and (Income is Medium) then (Applicant is Medium)
    - (Asset == High AND Income == Medium) AND Applicant == Medium ==> C10
11. If (Asset is High) and (Income is High) then (Applicant is High)
    - (Asset == High AND Income == High) AND Applicant == High ==> C11
12. If (Asset is High) and (Income is Very_high) then (Applicant is High)
    - (Asset == High AND Income == Very_high) AND Applicant == High ==> C12

Rule Base 1 Combining

- => Rule = C1 OR C2 OR C3 OR C4 OR C5 OR C6 OR C7 OR C8 OR C9 OR C10 OR C11 OR C12

```
    # combine the rules
    applicant_act_low = or_rule(applicant_act_low1, applicant_act_low2, applicant_act_low3)

    step = or_rule(applicant_act_medium1, applicant_act_medium2, applicant_act_medium3)
    applicant_act_medium = np.fmax(step, applicant_act_medium4)

    step = or_rule(applicant_act_high1, applicant_act_high2, applicant_act_high3)
    applicant_act_high = or_rule(step, applicant_act_high4, applicant_act_high5)

    applicant = or_rule(applicant_act_low, applicant_act_medium, applicant_act_high)

    # if we want to see the graph of the output
    if verbose == 1:
        plt.rcParams["figure.figsize"] = 15, 4
        plt.plot(x_applicant, applicant_low, 'b', linestyle='--', linewidth=1.5, label='Low')
        plt.plot(x_applicant, applicant_medium, 'g', linestyle='--', linewidth=1.5, label='Medium')
        plt.plot(x_applicant, applicant_high, 'r', linestyle='--', linewidth=1.5, label='High'),plt.title("Applicant Eva
        plt.legend()

        plt.fill_between(x_applicant, applicant, color='r')
        plt.ylim(-0.1, 1.1)
        plt.grid(True)
        plt.show()

    return applicant
```

House Evaluation Rule base 2

1. If (Market_value is Low) then (House is Low)
   - Market_value == Low AND House == Low ==> C1
2. If (Location is Bad) then (House is Low)
   - Location == Bad AND House == Low ==> C2
3. If (Location is Bad) and (Market_value is Low) then (House is Very_low)
   - (Location == Bad AND Market_value == Low) AND House == Very_Low ==> C3
4. If (Location is Bad) and (Market_value is Medium) then (House is Low)
   - (Location == Bad AND Market_value == Medium) AND House == Low ==> C4
5. If (Location is Bad) and (Market_value is High) then (House is Medium)
   - (Location == Bad AND Market_value == High) AND House == Medium ==> C5
6. If (Location is Bad) and (Market_value is Very_high) then (House is High)
   - (Location == Bad AND Market_value == Very_high) AND House == High ==> C6
7. If (Location is Fair) and (Market_value is Low) then (House is Low)
   - (Location == Fair AND Market_value == Low) AND House == Low ==> C7
8. If (Location is Fair) and (Market_value is Medium) then (House is Medium)
   - (Location == Fair AND Market_value == Medium) AND House == Medium ==> C8
9. If (Location is Fair) and (Market_value is High) then (House is High)
   - (Location == Fair AND Market_value == High) AND House == High ==> C9
10. If (Location is Fair) and (Market_value is Very_high) then (House is Very_high)
    - (Location == Fair AND Market_value == Very_high) AND House == Very_high ==> C10
11. If (Location is Excellent) and (Market_value is Low) then (House is Medium)
    - (Location == Excellent AND Market_value == Low) AND House == Medium ==> C11
12. If (Location is Excellent) and (Market_value is Medium) then (House is High)
    - (Location == Excellent AND Market_value == Medium) AND House == High ==> C12
13. If (Location is Excellent) and (Market_value is High) then (House is Very_high)
    - (Location == Excellent AND Market_value == high) AND House == Very_high ==> C13
14. If (Location is Excellent) and (Market_value is Very_high) then (House is Very_high)
    - (Location == Excellent AND Market_value == Very_high) AND House == Very_high ==> C14

```
    # combine the rules
    step = or_rule(house_act_low1, house_act_low2, house_act_low3)
    house_act_low = np.fmax(step, house_act_low4)

    house_act_medium = or_rule(house_act_medium1, house_act_medium2, house_act_medium3)

    house_act_high = or_rule(house_act_high1, house_act_high2, house_act_high3)

    house_act_very_high = or_rule(house_act_very_high1, house_act_very_high2, house_act_very_high3)

    step = or_rule(house_act_very_low, house_act_low, house_act_medium)
    house = or_rule(step, house_act_high, house_act_very_high)

    # if we want to see the graph of the output
    if verbose == 1:
        plt.rcParams["figure.figsize"] = 15, 4
        plt.plot(x_house, house_very_low, 'c', linestyle='--', linewidth=1.5, label='Very Low')
        plt.plot(x_house, house_low, 'b', linestyle='--', linewidth=1.5, label='Low')
        plt.plot(x_house, house_medium, 'g', linestyle='--', linewidth=1.5, label='Medium')
        plt.plot(x_house, house_high, 'r', linestyle='--', linewidth=1.5, label='High')
        plt.plot(x_house, house_very_high, 'y', linestyle='--', linewidth=1.5, label='Very High'),plt.title("House Evalu
        plt.legend()

        plt.fill_between(x_house, house, color='r')
        plt.ylim(-0.1, 1.1)
        plt.grid(True)
        plt.show()

    return house
```

RESULTS:

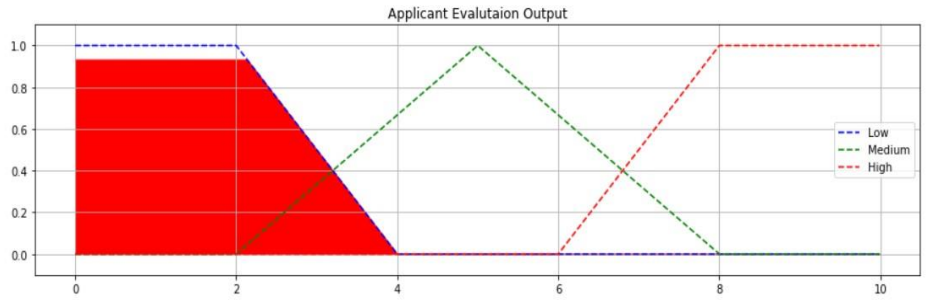
After implementing the Assets Valuation System, we obtained results for different inputs. Based on Rule Base, applicable rules are generated and then aggregated rules are formed. This fuzzy system accepts input in the form of two values, namely the Assets and Income of the applicant. Using these two inputs and the specified rules, system determines the Applicant's Worth. Similarly, system determines the worth of Applicant's House by taking House Location and House Market Value as inputs.

SCREENSHOTS OF OUTPUT:

Example Output For Rule Base 1 [Applicant Valuation]

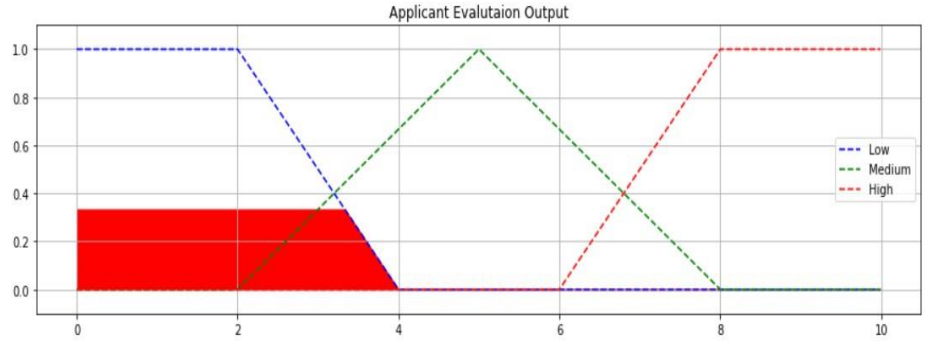
In [9]:
```
def make_decision(assets, income):
    # defuzzification with mean of maximum
    a_eval = apply_applicant_rules(assets, income, verbose=1)
    defuzz_applicant = fuzz.defuzz(x_applicant, a_eval, 'mom')
    max_n = np.max(a_eval)


    print ("Output: ₹", defuzz_applicant,"x 10^5 ")
    return defuzz_applicant
```

In [10]:
```
applicant_worth = make_decision(10,5)
```



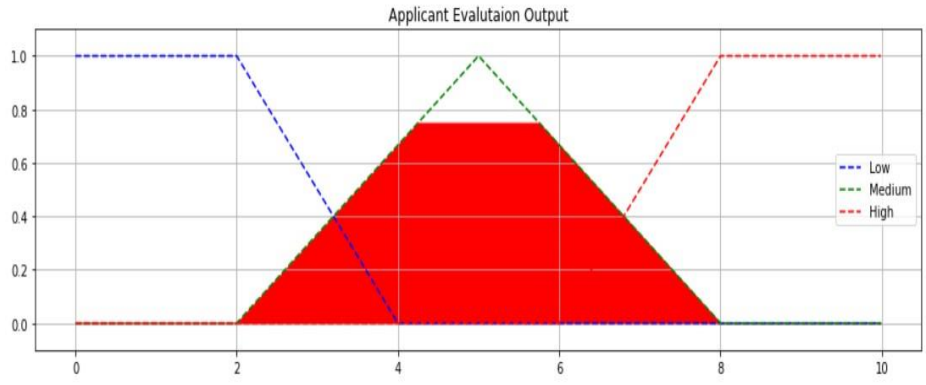

```
Output: ₹ 1.0650000000000002 x 10^5
```

In [11]:
```
applicant_worth = make_decision(100,10)
```




```
Output: ₹ 1.6649999999999996 x 10^5
```

In [16]:
```
applicant_worth = make_decision(500,40)
```

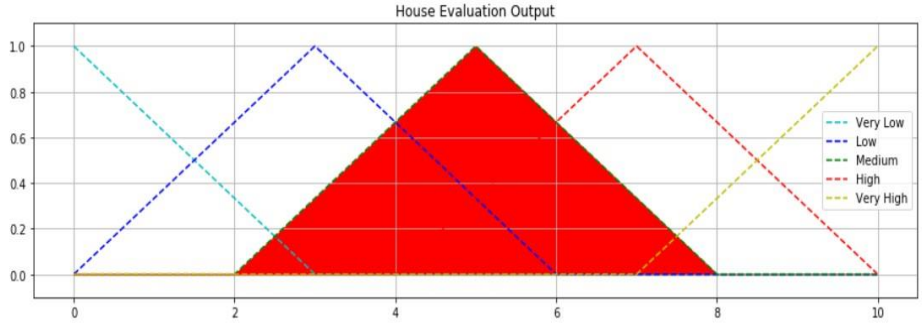



```
Output: ₹ 5.0 x 10^5
```

OUTPUT FOR RULE BASE 2 [HOUSE VALUATION]:

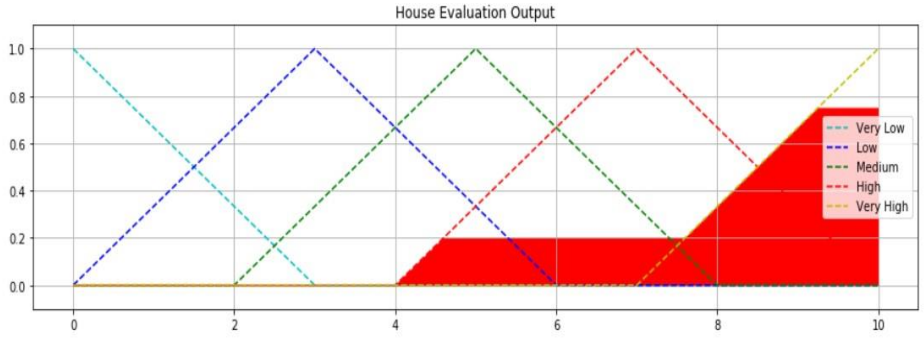
In [21]:
```
def make_decision2(market_value, location):
    # defuzzification with mean of maximum
    h_eval = apply_house_rules(market_value, location, verbose=1)
    defuzz_house = fuzz.defuzz(x_house, h_eval, 'mom')
    max_n = np.max(h_eval)


    print ("Output: ", defuzz_house)
    return defuzz_house
```

In [27]:
```
house_worth = make_decision2(100,5)
```



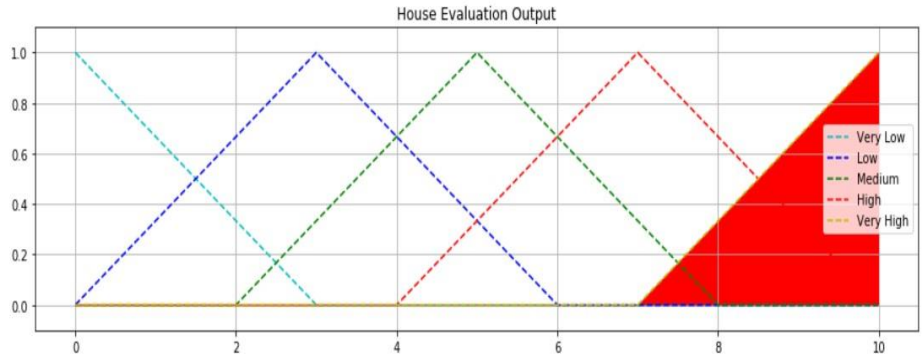

```
Output:  5.0
```

In [28]:
```
house_worth = make_decision2(800,8)
```



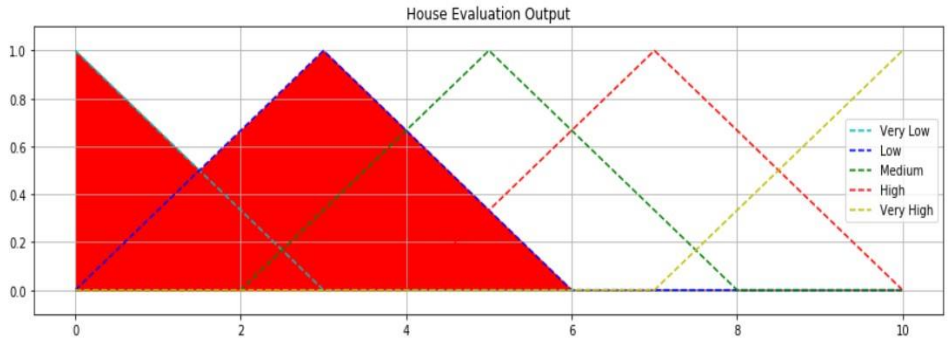

```
Output:  9.62
```

In [30]:
```
house_worth = make_decision2(900,9)
```




```
Output:  9.99
```

```
In [31]: house_worth = make_decision2(10,1)
```




```
Output:  1.5
```

CONCLUSION:

Thus, we have successfully designed an Assets Valuation System using Fuzzy Logic to determine the Worth of Applicant and worth of Applicant's House.

REFERENCES:

1] Ghita Bennouna , Mohamed Tkiouat. 2018. Fuzzy logic approach applied to credit scoring for microfinance in Morocco.

2] Sadig Mammadli. 2016. Fuzzy Logic Based Loan Evaluation System

3] Milica Latinovic, Ivana Dragovic, Vesna Bogojevic Arsic, Bratislav Petrovic. 2018. A Fuzzy Inference System for Credit Scoring using Boolean Consistent Fuzzy Logic

4] https://www.guru99.com/what-is-fuzzy-logic.html