Republic of Tunisia Ministry of Higher Education
and Scientific Research General Directorate of Technological
Studies Higher Institute of Technological Studies of Bizerte
Department of Electrical Engineering

Neural networks and fuzzy logic Project

Prepared by

Andolsi Skander

Hkiri Dhia

Title of the project

Fuzzy Logic Tip Calculator

1 Python

1.1 Definition of Python :

Python is a high-level, interpreted programming language that is widely used for web development, data analysis, artificial intelligence, and scientific computing. It was first released in 1991 and has since become a popular language due to its simplicity, readability, and extensive library of pre-built tools and modules that make it easy to develop a wide range of applications. Python is known for its simplicity, which allows developers to express concepts in fewer lines of code compared to other languages, and its flexibility, which makes it well-suited for a variety of tasks. It has a large, active community of developers who contribute to the development of the language and its ecosystem of libraries and tools.

1.2 Python for Fuzzy logic system :

Python is a powerful programming language that is widely used in many fields, including data science, machine learning, and artificial intelligence. It is also a popular language for implementing fuzzy logic systems.

Fuzzy logic is a mathematical approach to representing and manipulating uncertain or imprecise information. It is used to model systems with complex behavior, where precise calculations may not be possible or practical.

There are several libraries available in Python for implementing fuzzy logic systems. Some popular options include:

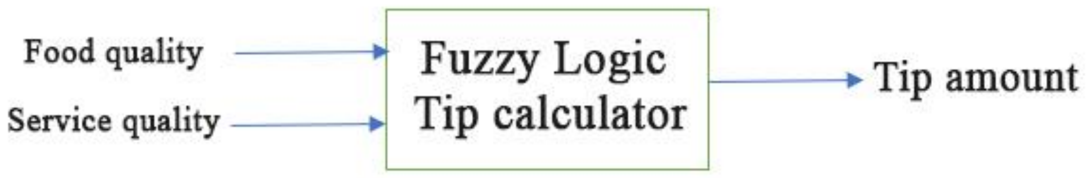
- **scikit-fuzzy**: This is a powerful library for fuzzy logic that includes functions for defining fuzzy sets, creating membership functions, and performing fuzzy inference.
- **fuzzywuzzy**: This is a library for fuzzy string matching, which can be useful for implementing fuzzy logic systems that involve processing natural language data.
- **pyfuzzy**: This is a library for designing and simulating fuzzy logic systems. It includes functions for defining fuzzy variables, creating membership functions, and performing fuzzy inference.

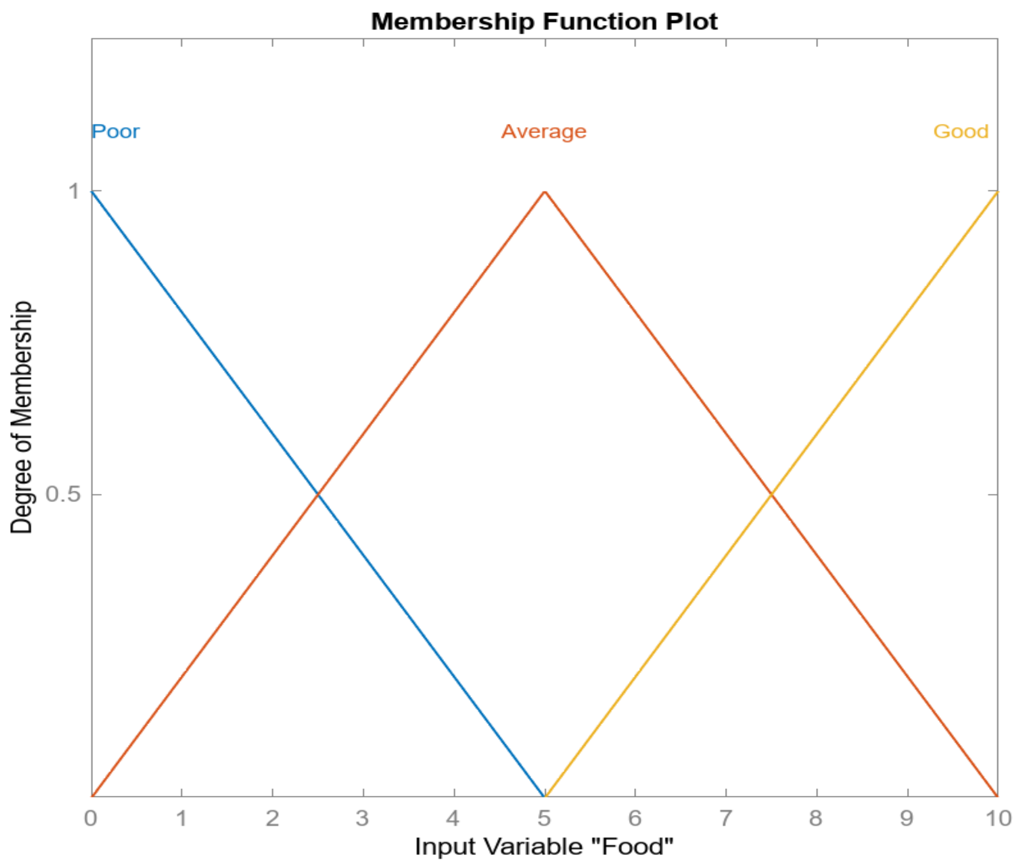
It is also possible to implement fuzzy logic systems from scratch using Python's built-in data types and functions. However, using a library can save you time and make it easier to develop and test your fuzzy logic system.

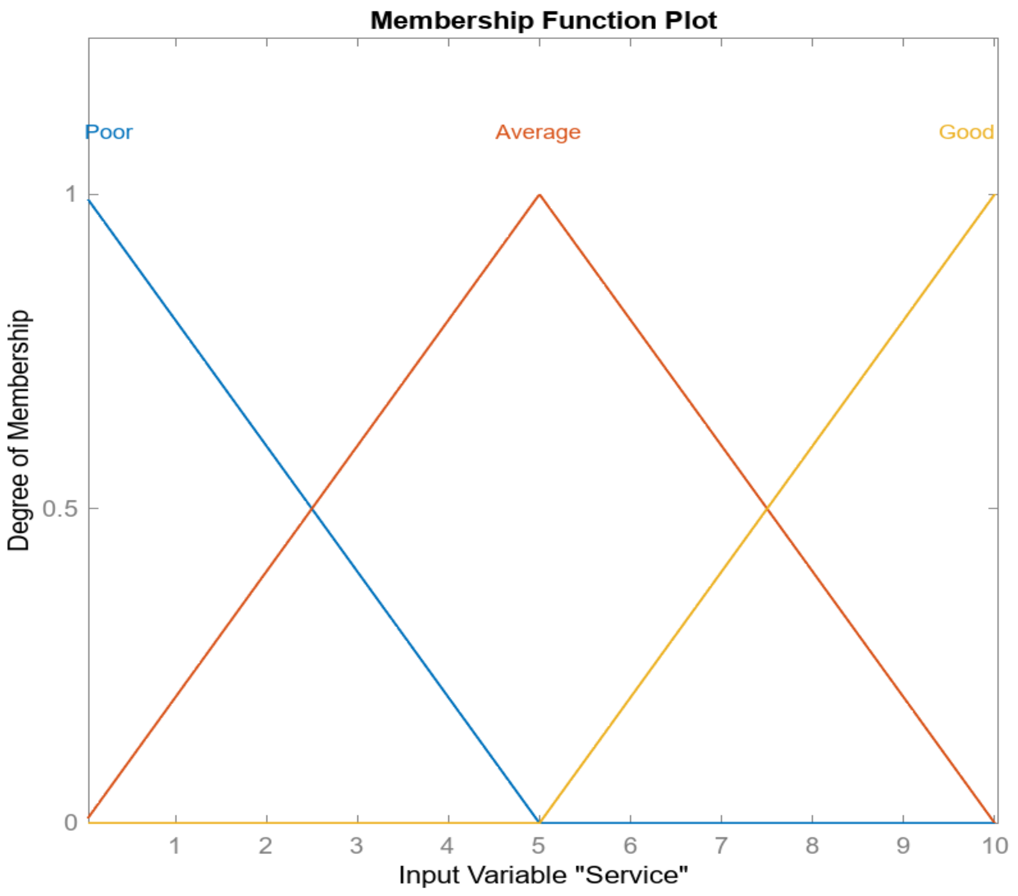
2 Context

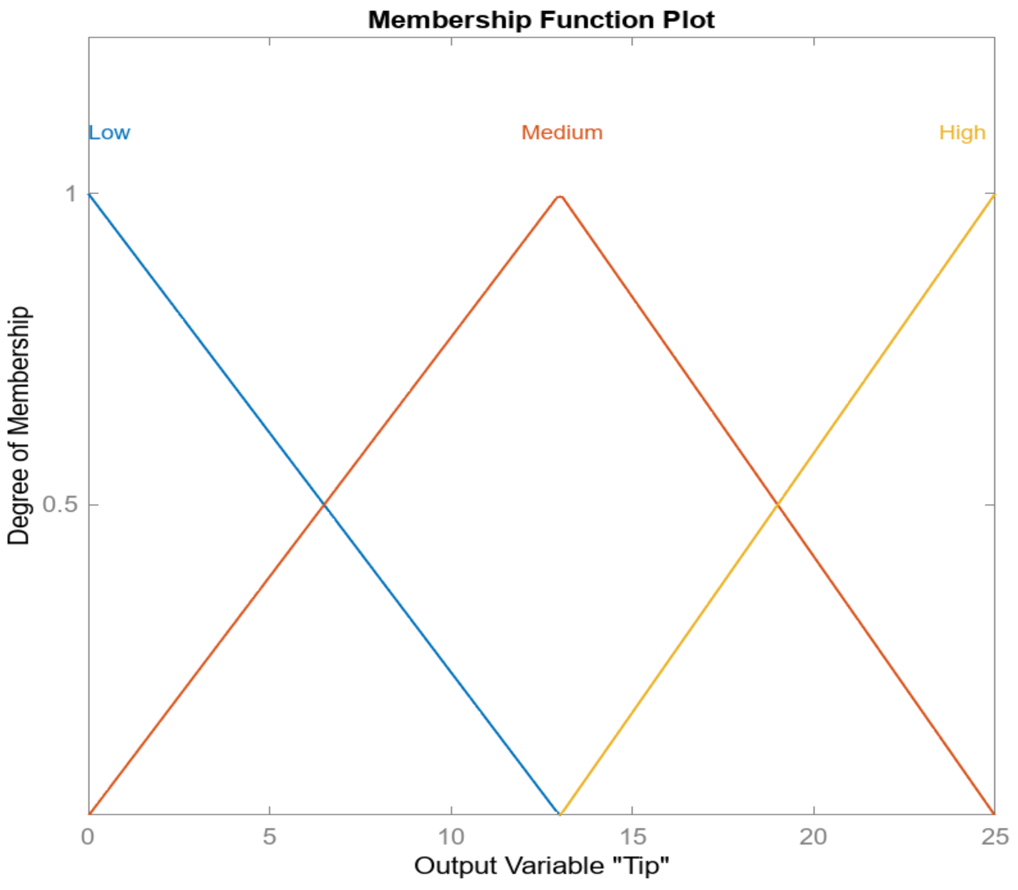
In this project, I will be using fuzzy logic to develop a tip calculator in the Python programming language that takes into account the subjective ratings of food and service quality at a restaurant. Fuzzy logic is a mathematical tool that allows for the use of imprecise or subjective information to make decisions. It is particularly useful in situations where traditional logic may not be applicable or where subjective data needs to be taken into account. In the case of determining a recommended tip amount, food and service quality can be subjective and may not fit neatly into a traditional true/false or yes/no category. By using fuzzy logic, we can incorporate these subjective ratings and use them to make a recommendation for a tip amount that reflects the overall quality of the dining experience. To implement our fuzzy logic tip calculator in Python, I will use the “skfuzzy” library, which provides a set of tools for working with fuzzy logic in Python. I will define the input data (food and service quality ratings) and the rules that determine the recommended tip amount based on those ratings.

I will then test the calculator using a variety of input data to see how it performs.

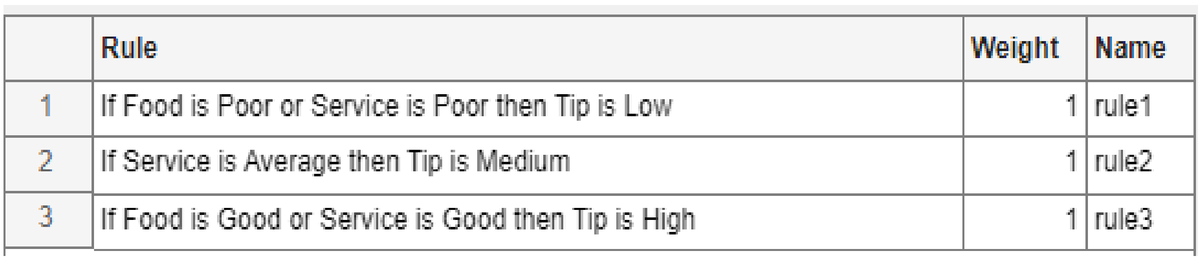





Membership Function Plot
Poor
Average
Good
Degree of Membership
1
0.5
0
1
2
3
4
5
6
7
8
9
10
Input Variable "Service"


Membership Function Plot
Low
Medium
High
Degree of Membership
1
0.5
0
5
10
15
20
25
Output Variable "Tip"

Rules table :



	Rule	Weight	Name
1	If Food is Poor or Service is Poor then Tip is Low	1	rule1
2	If Service is Average then Tip is Medium	1	rule2
3	If Food is Good or Service is Good then Tip is High	1	rule3

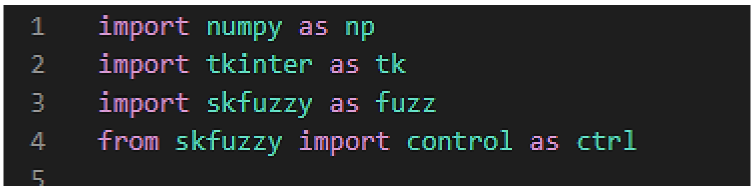
3 Motivation

Fuzzy logic can be applied to find the recommended tip amount in a restaurant based on food and service quality inputs because it allows for the use of imprecise or subjective information to make decisions. In the case of determining a recommended tip amount, food and service quality can be subjective and may not fit neatly into a traditional true/false or yes/no category. For example, the quality of food and service at a restaurant can be applied to predict the amount of tips. However, these amounts are subjective and can vary from person to person. Using fuzzy logic, it is possible to take these subjective ratings and use them to make a recommendation for a tip amount that takes into account the overall quality of the experience at the restaurant. Fuzzy logic allows for the use of "fuzzy" or imprecise input data and uses a set of rules to make a decision based on that data. In this case, the input data would be the ratings for food and service quality, and the rules would determine the recommended tip amount based on those ratings. Fuzzy logic can be a useful tool in situations where traditional logic may not be applicable or where imprecise or subjective information needs to be taken into account.

4 Project Code

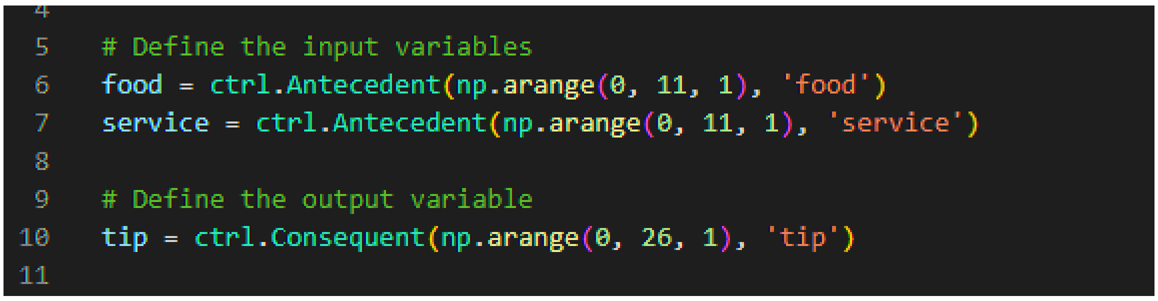
❖ This code imports three libraries: **Numpy**, **Skfuzzy**, **tkinter**, and the **control** module from **Skfuzzy** :

- ✓ Numpy is a library for working with large, multi-dimensional arrays and matrices of numerical data, and is a fundamental package for scientific computing with Python.
- ✓ Skfuzzy is a library for fuzzy logic control systems development in Python. Fuzzy logic is a mathematical approach to modeling complex, imprecise, or ill-defined systems. It allows for the representation of uncertainty and imprecision in data, and can be used to build intelligent control systems that can adapt and make decisions based on ambiguous or incomplete information.
- ✓ The control module from skfuzzy provides tools for creating and manipulating fuzzy control systems. This includes functions for defining fuzzy variables, rules, and membership functions, as well as tools for simulating and evaluating the performance of a fuzzy control system.
- ✓ tkinter is a module in Python that is used for creating graphical user interfaces (GUIs). It provides a powerful set of tools for creating GUI applications in Python. The tk alias is used as a shorthand for tkinter throughout the rest of the code, so instead of writing tkinter.Button, for example, you can just write tk.Button. This makes the code easier to read and reduces the amount of typing required.



```python
import numpy as np
import tkinter as tk
import skfuzzy as fuzz
from skfuzzy import control as ctrl
```

❖ Define of input/output range :

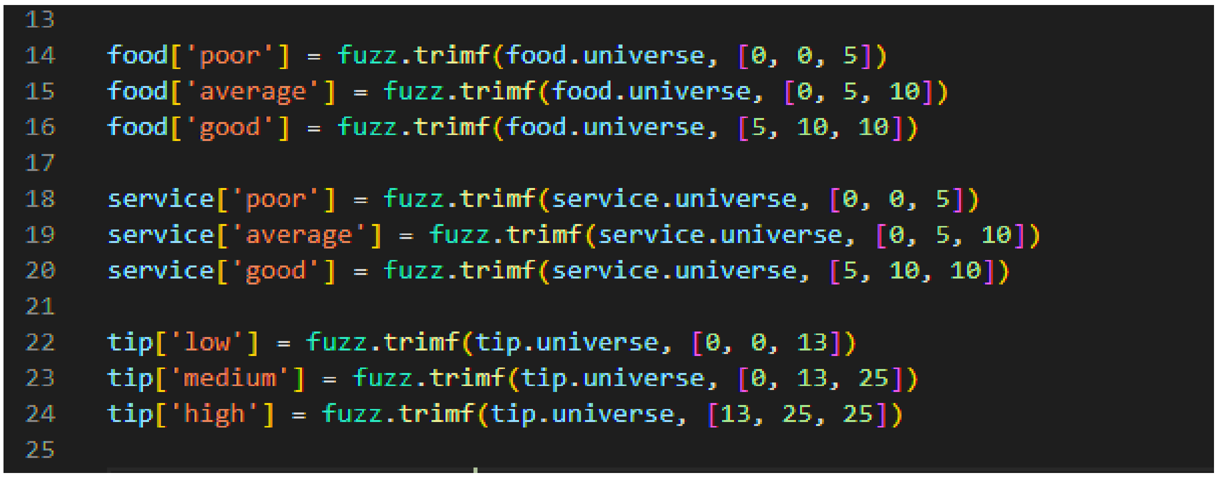


```python

# Define the input variables
food = ctrl.Antecedent(np.arange(0, 11, 1), 'food')
service = ctrl.Antecedent(np.arange(0, 11, 1), 'service')

# Define the output variable
tip = ctrl.Consequent(np.arange(0, 26, 1), 'tip')

```

The **Antecedent** class is used to define input variables, while the **Consequent** class is used to define output variables. These functions take two arguments: a NumPy array specifying the range of the variable, and a string label for the variable.

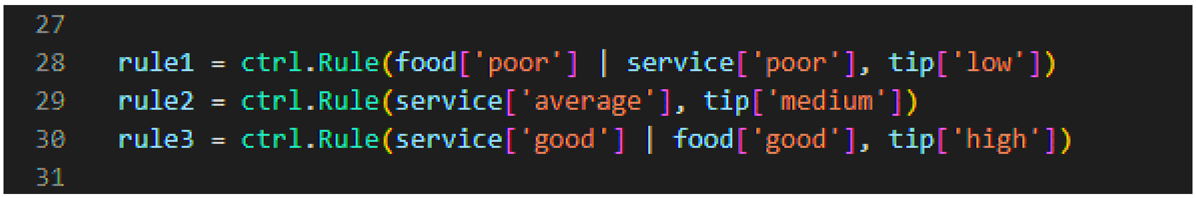
❖ Define of the membership functions of input and output value :

All the membership functions are defined using the **trimf** (triangular membership function) method from the fuzz module



```python

food['poor'] = fuzz.trimf(food.universe, [0, 0, 5])
food['average'] = fuzz.trimf(food.universe, [0, 5, 10])
food['good'] = fuzz.trimf(food.universe, [5, 10, 10])

service['poor'] = fuzz.trimf(service.universe, [0, 0, 5])
service['average'] = fuzz.trimf(service.universe, [0, 5, 10])
service['good'] = fuzz.trimf(service.universe, [5, 10, 10])

tip['low'] = fuzz.trimf(tip.universe, [0, 0, 13])
tip['medium'] = fuzz.trimf(tip.universe, [0, 13, 25])
tip['high'] = fuzz.trimf(tip.universe, [13, 25, 25])

```

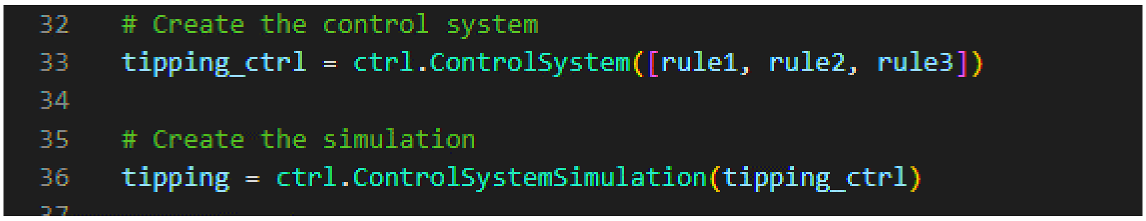
❖ Define the Fuzzy rules for the output variable :



```python

rule1 = ctrl.Rule(food['poor'] | service['poor'], tip['low'])
rule2 = ctrl.Rule(service['average'], tip['medium'])
rule3 = ctrl.Rule(service['good'] | food['good'], tip['high'])

```

Each rule is defined using the ctrl.Rule function, which takes two arguments: an antecedent (a logical condition based on the input variables) and a consequent (the output to be produced if the antecedent is true).

❖ Creates a fuzzy control system and a simulation of that control system :



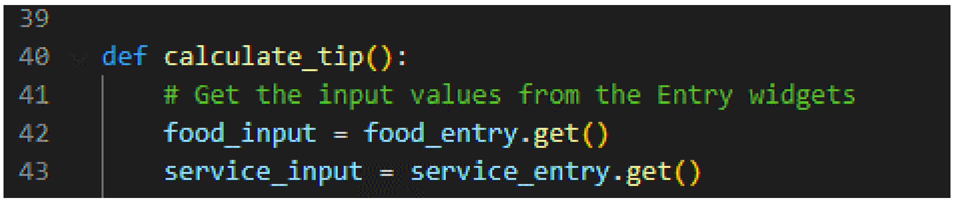
```python
# Create the control system
tipping_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])

# Create the simulation
tipping = ctrl.ControlSystemSimulation(tipping_ctrl)
```

The ctrl.ControlSystem function is used to create a control system object from a list of rules. The control system object is responsible for storing and organizing the rules, and for applying them to determine the output of the system.

The ctrl.ControlSystemSimulation function is used to create a simulation object from a control system. The simulation object can be used to evaluate the performance of the control system under different input conditions.

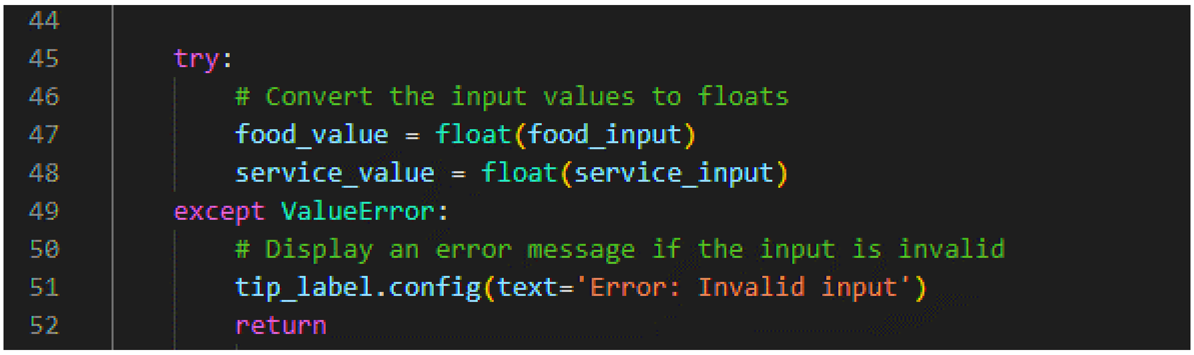
❖ Define the "calculate_tip" function :

This defines the calculate_tip function, which is called when the user clicks the 'Calculate Tip' button. It retrieves the food and service quality input values from the Entry widgets and stores them in '**food_input**' and '**service_input**', respectively. These values are still strings at this point.


```python
def calculate_tip():
    # Get the input values from the Entry widgets
    food_input = food_entry.get()
    service_input = service_entry.get()
```

❖ Convert input values to floats and handle invalid input :

This block attempts to convert the ‘**food_input’** and ‘**service_input’** values to floats using the **float()** function. If the input is not a valid float (e.g. a non-numeric value is entered), a ‘**ValueError’** will be raised and caught by the ‘**except’** clause. In this case, the tip label widget's text is updated to display an error message and the function returns without executing the rest of the code.

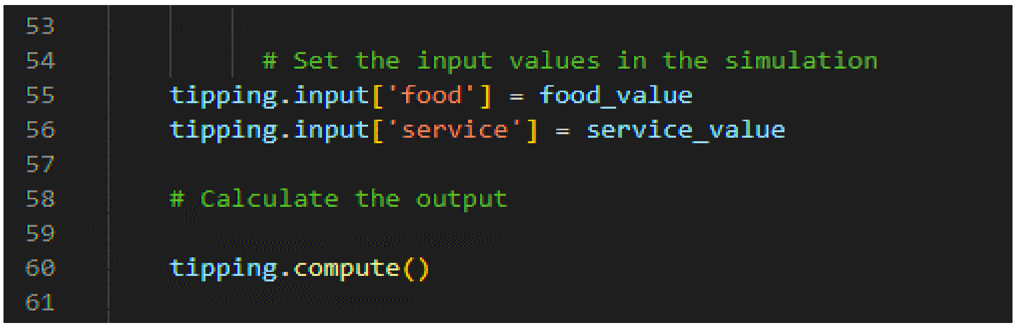

```python

    try:
        # Convert the input values to floats
        food_value = float(food_input)
        service_value = float(service_input)
    except ValueError:
        # Display an error message if the input is invalid
        tip_label.config(text='Error: Invalid input')
        return
```

the **try** and **except** statements are used to handle exceptions. Exceptions are errors that occur at runtime (when a program is running).

❖ Set the input values and calculate the output :

This sets the ‘**food’** and ‘**service’** input values in the **tipping** object.

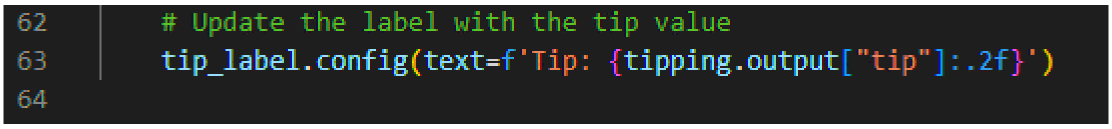
Tipping is the simulation of the control system (already explained in Creates a fuzzy control system and a simulation of that control system)


```python

            # Set the input values in the simulation
    tipping.input['food'] = food_value
    tipping.input['service'] = service_value

    # Calculate the output

    tipping.compute()

```

The **compute** method of the simulation object applies the rules of the control system to determine the output of the system based on the current values of the input variables.

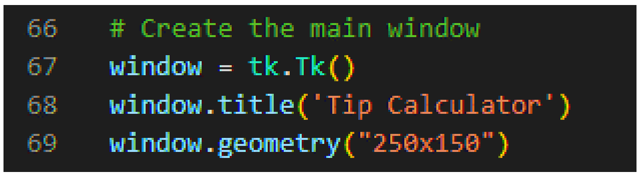
❖ Update the label with the tip value :

This updates the text of the tip label widget to display the calculated tip value, which is retrieved from the ‘**output’** attribute of the ‘**tipping’** object and formatted to two decimal places.


```python
    # Update the label with the tip value
    tip_label.config(text=f'Tip: {tipping.output["tip"]:.2f}')

```

- Create the GUI (main window) :

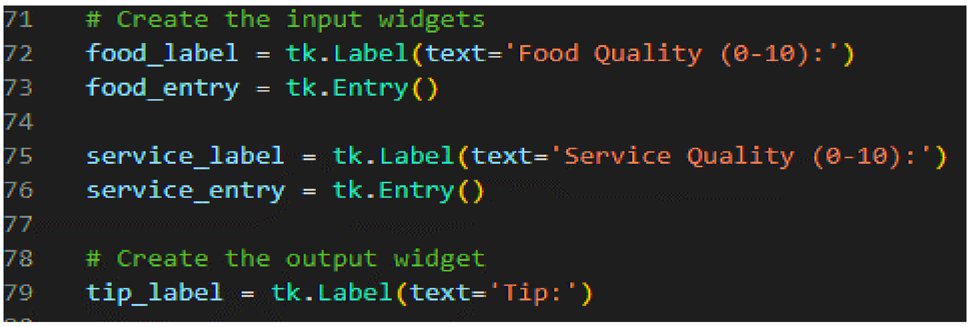
This creates the main window for the GUI and sets its title and dimensions.



```python
# Create the main window
window = tk.Tk()
window.title('Tip Calculator')
window.geometry("250x150")
```

- Create the input and output widgets :

These lines create the food quality label and entry widgets, and the service quality label and entry widgets. The label widgets display the text 'Food Quality (0-10):' and 'Service Quality (0-10):',the entry widgets allow the user to input the food and service quality values.

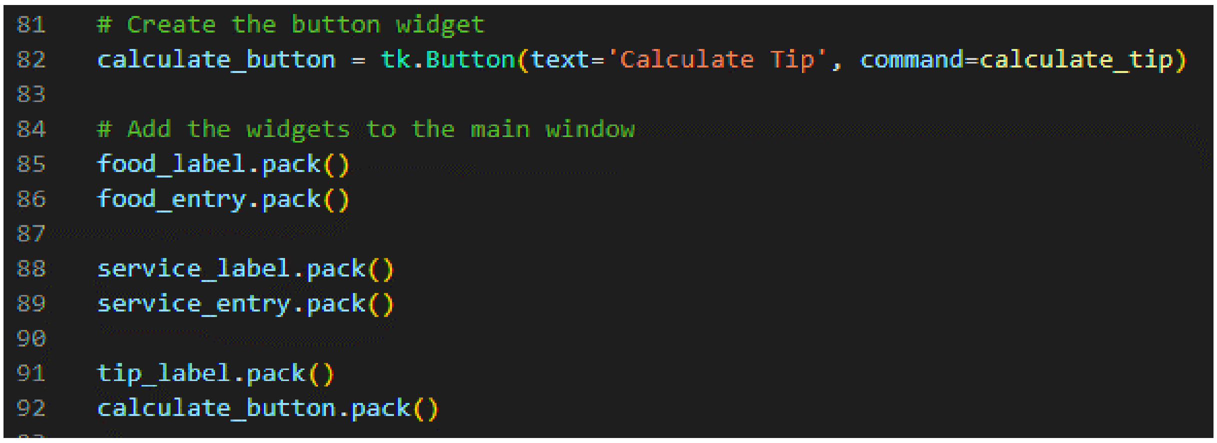


```python
# Create the input widgets
food_label = tk.Label(text='Food Quality (0-10):')
food_entry = tk.Entry()

service_label = tk.Label(text='Service Quality (0-10):')
service_entry = tk.Entry()

# Create the output widget
tip_label = tk.Label(text='Tip:')
```

This creates the tip label widget, which will be used to display the calculated tip value

- Create the button widget and add all the widgets to the main window :

This creates a button widget with the text 'Calculate Tip' and a callback function specified by the '**command'** parameter. When the button is clicked, the' **calculate_tip'** function will be called.

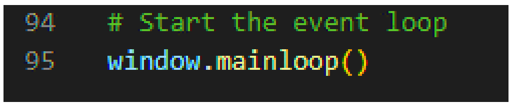


```python
# Create the button widget
calculate_button = tk.Button(text='Calculate Tip', command=calculate_tip)

# Add the widgets to the main window
food_label.pack()
food_entry.pack()

service_label.pack()
service_entry.pack()

tip_label.pack()
calculate_button.pack()
```

These lines add the input, output, and button widgets to the main window using the '**pack()'** method.

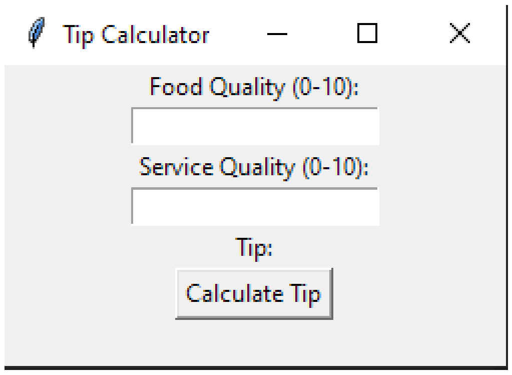
- Start the event loop :

This starts the event loop, which waits for user input and updates the window accordingly. The event loop will run until the window is closed by the user.

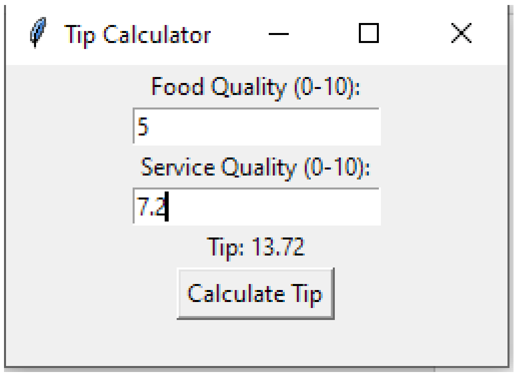


```python
# Start the event loop
window.mainloop()
```

❖ Simulation :



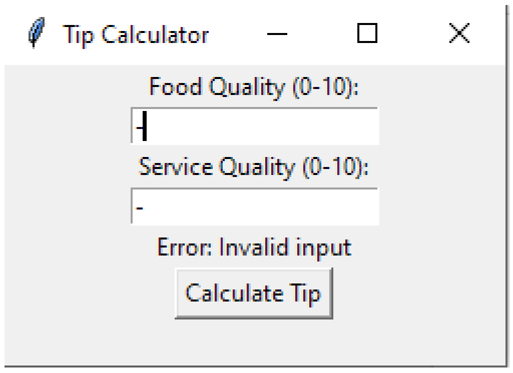
❖ Results :



5

7.2

Tip: 13.72



-

-

Error: Invalid input

Conclusion

In this project, I have developed a fuzzy logic tip calculator using Python that takes into account the subjective ratings of food and service quality at a restaurant to make a recommendation for a tip amount. By using fuzzy logic, I was able to incorporate imprecise or subjective data and make a decision based on a set of rules. The results of our tip calculator show that it is able to make reasonable and logical recommendations for tip amounts based on the input data. It is able to take into account both the food and service quality ratings and provide a recommended tip amount that reflects the overall quality of the dining experience. Overall, this fuzzy logic tip calculator demonstrates the usefulness of fuzzy logic in situations where traditional logic may not be applicable or where subjective data needs to be taken into account. It is a useful tool for making decisions based on imprecise or subjective information and can be easily implemented using Python.