Neuro-Fuzzy Inference System

AIM :

To implement Neuro-Fuzzy-Inference-System using python

PROGRAM :

main.py

```
from visualize import  *

if __name__ == '__main__':
   run = visualize()
```

visualize.py

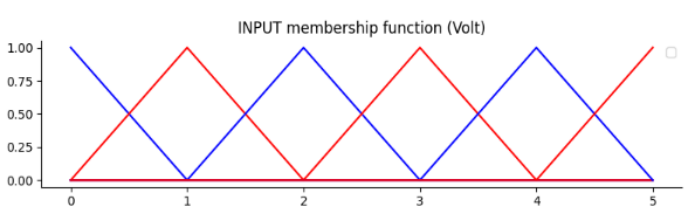
```
import cv2
from tkinter import *
from tkinter import ttk, messagebox
from rpm import motor_rmp
from tip import tip


class visualize():
   def __init__(self):

      self.window = Tk()

      self.window.title("Fuzzy Neural Network Systems")
      self.window.geometry('800x950')

      self.tab_control = ttk.Notebook(self.window)

      self.tab1 = ttk.Frame(self.tab_control)
      self.tab2 = ttk.Frame(self.tab_control)
      self.tab3 = ttk.Frame(self.tab_control)

      self.tab_control.add(self.tab1, text='RPM Calculate')
      self.tab_control.add(self.tab2, text='TIP Calculate')
      self.tab_control.add(self.tab3, text='Info')

      self.introframe = ttk.Labelframe(self.tab1, )
      self.info_1 = ttk.Labelframe(self.tab3,text= "Submited to")
      self.info_2 = ttk.Labelframe(self.tab3,text= "Submited by")
      self.l_frame = ttk.Labelframe(self.tab1, width=100, height=100)
```

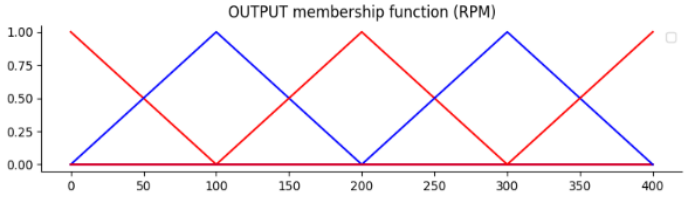
```
        self.v_frame = ttk.Labelframe(self.tab1,text='visualize')

        heading = Label(self.introframe, text="Fuzzy Neural Network Systems\n Motor Control Using
Mandani Method",font=("Arial", 12))
        heading.grid(column=0,row=0,sticky=N)

        submited_to = Label(self.info_1, text=" Prof. Young Im Cho\n Gachon University, South Korea",
                    font=("Arial", 12))
        submited_to.grid(column=0, row=1, sticky=N)

        submited_by = Label(self.info_2, text="Soikat Hasan Ahemd\n ID: 202040110 \nGachon University,
South Korea",
                        font=("Arial", 12))
        submited_by.grid(column=0, row=1, sticky=N)

        Result_txt = Label(self.l_frame, text="Calculated RPM : ",font=("Arial", 14))
        Result_txt.grid(column=0, row=4,sticky=N)

        self.Result = Label(self.l_frame, text="N/A",font=("Arial", 16))
        self.Result.grid(column=1, row=4,sticky=N)

        input_txt = Label(self.l_frame, text="Insert Voltage\n (0~5) : ",font=("Arial", 14))
        input_txt.grid(column=0, row=2,sticky=N)
        self.txt = Entry(self.l_frame,width=10)
        self.txt.grid(column=1, row=2,sticky=N)
        btn = Button(self.l_frame, text="Calculate RPM", command=self.clicked)
        btn.grid(column=3, row=2,sticky=N)

        self.canvas = Canvas(self.v_frame, width=700, height=800)
        # canvas.pack()

        self.canvas.grid( row=0)

        #.....................................................................................

        self.introframe2 = ttk.Labelframe(self.tab2, )
        self.l_frame2 = ttk.Labelframe(self.tab2, width=100, height=100)
        self.v_frame2 = ttk.Labelframe(self.tab2, text='visualize')

        heading2 = Label(self.introframe2, text="Fuzzy Neural Network Systems\n TIP Calculation Using
Mandani Method",
                    font=("Arial", 12))
        heading2.grid(column=0, row=0, sticky=N)

        Result_txt2 = Label(self.l_frame2, text="Calculated TIP : ", font=("Arial", 14))
        Result_txt2.grid(column=0, row=4, sticky=N)

        self.Result2 = Label(self.l_frame2, text="N/A", font=("Arial", 16))
```

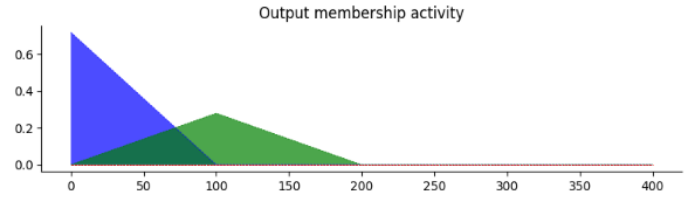
```
    self.Result2.grid(column=1, row=4, sticky=N)

    input_txt1 = Label(self.l_frame2, text="Insert service\n (0~10) : ", font=("Arial", 14))
    input_txt1.grid(column=0, row=2, sticky=N)
    self.txt11 = Entry(self.l_frame2, width=10)
    self.txt11.grid(column=1, row=2, sticky=N)

    input_txt_2 = Label(self.l_frame2, text="Insert Food Quality\n (0~10) : ", font=("Arial", 14))
    input_txt_2.grid(column=0, row=3, sticky=N)


    self.txt12 = Entry(self.l_frame2, width=10)
    self.txt12.grid(column=1, row=3, sticky=N)
    btn2 = Button(self.l_frame2, text="Calculate TIP", command=self.clickedtip)
    btn2.grid(column=3, row=3, sticky=N)

    self.canvas2 = Canvas(self.v_frame2, width=700, height=800)
    # canvas.pack()

    self.canvas2.grid(row=0)

    self.tab_control.pack(expand=1, fill='both')

    self.introframe.pack()
    self.introframe2.pack()
    self.info_1.pack()
    self.info_2.pack()

    self.l_frame.pack()
    self.l_frame2.pack()

    self.v_frame.pack()
    self.v_frame2.pack()


    self.window.mainloop()

  def clicked(self):

    input_txt = self.txt.get()
    if len(input_txt) == 0:
       messagebox.showwarning('Input Error', 'Please Input A Voltage (0~5)')
    else:
       try:
          value = float(input_txt)

          if value < 0.0 or value > 5.0:
             messagebox.showwarning('Input Error', 'value Out of range.\n Input should be 0 ~ 5')
```

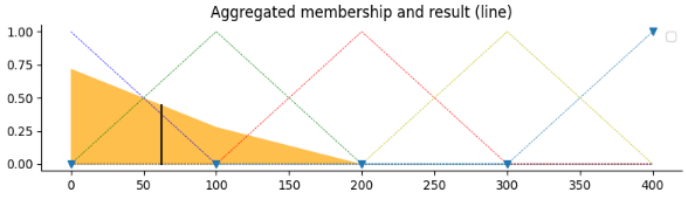
```
            else:
                out = motor_rmp(value)
                self.Result.configure(text='{:02f}'.format(out))
                img = cv2.imread('output/out.png')
                resize = cv2.resize(img,(600,700))
                cv2.imwrite('output/out.png', resize)
                self.img = PhotoImage(file="output/out.png")

                self.canvas.create_image(20, 20, anchor=NW, image=self.img)
                # self.canvas.configure(image=self.img)

        except:
            messagebox.showwarning('Input Error', 'Input Should be Integer or Float Value')

def clickedtip(self):

    input_txt1 = self.txt11.get()
    input_txt2 = self.txt12.get()
    if len(input_txt1) == 0 or len(input_txt2) == 0 :
        messagebox.showwarning('Input Error', 'Input can not be empty')
    else:
        try:
            service = float(input_txt1)
            quality = float(input_txt2)

            if service < 0.0 or service > 10.0 or quality < 0.0 or quality > 10.0:
                messagebox.showwarning('Input Error', 'value Out of range.\n Input should be 0 ~ 10')
            else:
                out = tip(quality,service)
                self.Result2.configure(text='{:02f}'.format(out))
                img2 = cv2.imread('output/out2.png')
                resize2 = cv2.resize(img2, (600, 700))
                cv2.imwrite('output/out2.png', resize2)
                self.img2 = PhotoImage(file="output/out2.png")

                self.canvas2.create_image(20, 20, anchor=NW, image=self.img2)


        except:
            messagebox.showwarning('Input Error', 'Input Should be Integer or Float Value')
```

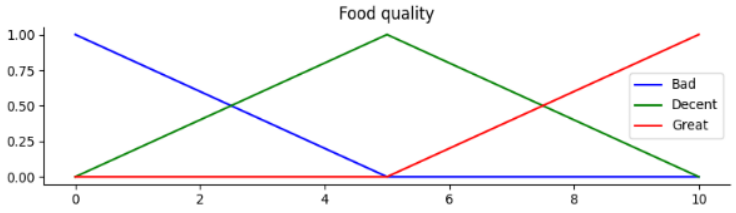
OUTPUT :

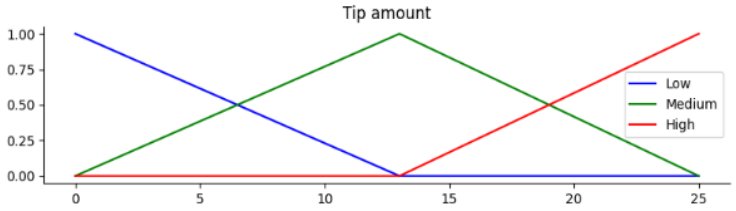


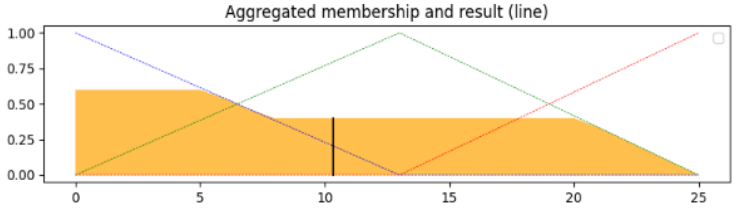












RESULT :

Thus the Neuro-fuzzy Inference System has been implemented using python