**STUDENT NAME**

**COURSE NAME**

**STUDENT NUMBER**

**TOPIC: FUZZY LOGIC IMPLEMENTATION WITH PYTHON PROGRAMMING**

**Question One:**

Rather than spew out random 0s and 1s, computers can also be trained to make judgements to a certain degree of truth. These have led to programmable machines that almost represent human beings. They are called artificial intelligent agents or smart agents. The inception of smart agents came as a result of the need to automate both simple and complex tasks to meet the final desired user needs.

For instance, with globalisation and industrialisation, so did the technology industry equally grow? Factories and industries were producing goods and commodities in bulk, the labour force that were present at the time could not efficiently handles this need, so machines had to be automated. Computers scientists came up with designs and approaches to meet this, need.

Thereafter, a better way had to be found to meet communication between computers and human beings. What this would require is that the computer had to recognize the language of the human, interpretively and give feedback to the user. This involved one,

* Having a collection of large volumes o data
* Gathering data overtime like say over a few 2-4 years on the same subject gathered across different touch points
* Intensively training the dataset in order to tine it to what would be needed as expected case sceneries by the smart agents.
* Programming the agents to accept inputs and then strong expected output from the machine.
* An A/B test for instance, would be applied on the robotic agent and subjected to different test sceneries, fail and pass test cases done and record, for later re-adjustments and approvals.

**Applications of smart agents:**

Smart agent also known as artificial intelligent agents have been applied in real life scenarios to solve several business problems.

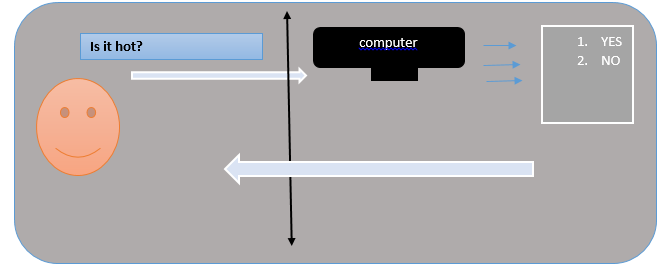
Google has the automated voice search that lets a user search items on the web just by speaking into the microphone. How it does this is that, a series of programmable memories items have been stored into the database that whenever a similar input is detected, searches for that particle memory file and executes the expected outcome which is the presented to the user. Another example is Amazon Alexa that works in a similar manner.

There are several output responses that are store within amazon services, and using the programmable memory of the machine, these devices are trained to accept user input in terms of voice and sound and then proceed to execute the command. A command for instance would include, play music, solve a math problem or predict tomorrow’s weather forecast. Alongside smart AI agents, we have fuzzy applications. There is not much a difference between the two except that fuzzy applications perform simpler activities than smart agents.

Their applications have been used in security, college entry admissions, ticket bookings and ATM Financial services. They are trained to think and behave like human beings. This is based on their abilities to recognise particular inputs, validate them and based on the written algorithms and then pass a feedback to the user. Based on the particular input supplied by their user, the required responses is produced and if an invalid entry occurs, the application informs user of so.

With time, these robots have been programmed to almost make truthful judgements based on certain inputs from a human interactions perspective. The interesting bit is that today fuzzy applications have advanced and have been incorporated in fields such as medicine, to help patients self-diagnose treatments from the comfort of their homes, agriculture to help famers know the correct amount of manure and fertiliser to apply to achieve a given amount of crop yield, motor vehicle industry to help drivers navigate maps and detect traffic jams along the roads.

The hotel and hospitality industry use fuzzy applications to help customers book rooms, by supplying their intended check in dates and then recommending some of the best and affordable rooms and hotels based on inputs,. Traditional computers would select Boolean responses as output with no further declarations. Look at the figurative below:



From the above illustration, the computer will only give an output of Yes or No, based on its binary representation of digits as 0s and 1s. However, with fuzzy logic, this is taken a step further and implemented in smart agents, so instead of just giving a single handed answer, the computer also goes ahead to give full details of temperature and recommended clothes to put on. Whereas fuzzy applications can be seen as complex input output machines, behind the scenes are a lot of algorithms iterating through conditional statements that will hold true as the other conditions execute.

These loops are what gives out an expected answer to the user. For instance, a user would enter pin in an ATM to withdraw money, behind the program, the ATM, checks for the user balance, while still logs in, goes to main account balance, checks if there is sufficient to withdraw and I f there is continue holding the programming so that subtraction of tax and deduction costs can be made, then pass the balance value required for withdraw.

Moreover, we can also use fuzzies, to simulate an interaction between students and the university, this can automate for instance a student college recommendation processes and from there be able to help students make decisions on what to do next after the recommendation has been made. In such a case, the student seeking admission will enter a given input based on the prompts that appear in the screen , and based on the inputs given, the computer will be able to provide instant feedback to the user as to whether they would accept the institution or not.

Such a program can be developed in Python and executed within any Python interpreter such as the Python idle, windows CMD or an online interpreter such as <https://www.onlinegdb.com/online_python_interpreter> . For this activity , the code given below generates a fuzzy logic that will pick the selection entered, prompt an input and then give an interactive answer to the student as to whether they should pick the institution or not.

#define an new function to execute code

def fuzzy\_check():

    n = int(input("Enter Number of Institutes: ")) #promt user for the nunber of institutes

def check\_float(f1):

    try:#in case we have bugs, use the try syntax to handle

        float(f1)

        return True

    except:

        return False

    while n:# start our loop condition, validate the inputs and check for return statements

        faith = input("Enter Faith Value of Institute: ")

        if not check\_float(faith):

            print("\tInvalid Value...")

        else:

            faith = float(faith)

            if 0<=faith<0.3:

                print("\t--> Don't take admission")

            elif 0.3<=faith<0.5:

                print("\t--> Not recommended. But you may consider.")

            elif 0.5<=faith<0.8:

                print("\t--> Good institute to take admission.")

            elif 0.8<=faith<=1:

                print("\t--> Excellent institute. Highly recommended.")

            else:

                print("\tFuzzy Value must be between 0 and 1")

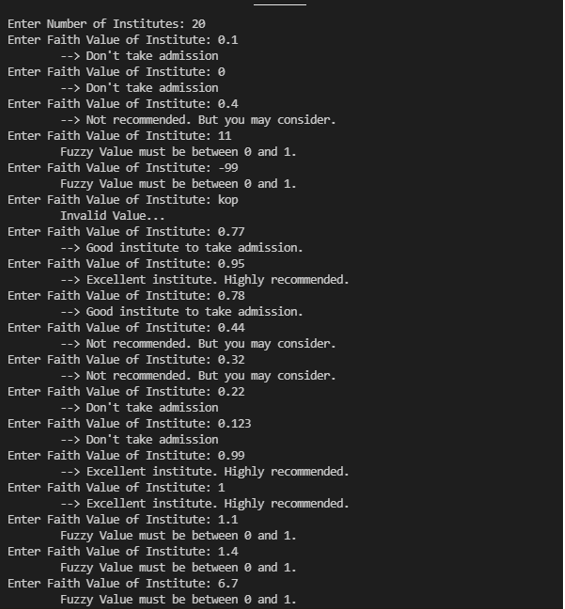
                n-=1

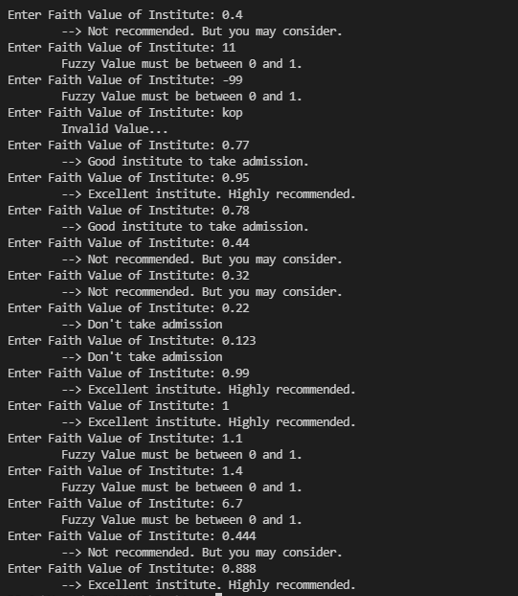
            return fuzzy\_check()

Once the code runs and executes smoothly on a python environment, the variable “faith” is taken and stored by the program then put a side in hold, meanwhile a loop statement starts and verifies classifications of inputs as 0-03, 0.3-0.5, 0.5-0.8 and finally 0.8 -1. These inputs shall hold true as long as the condition is still true, if a value out of 0 and 1, then an error is thrown and user has to enter a valid memory stack value between 0-1.

What this fuzzy logic has done in human Natural Language processing is to build a faith and confidence value in particular institutions by ranking them, where therefore proceed to recommend these institutes to potential students who would like to join them.

Based on the above, successful output of the code is as shown below:





We have run the test cases against a population of say 37 students with each giving a possibly different input and the corresponding output shown on the screens as long as the conditional statements hold true. If the student enters a value out of the scope of 0-1 then an “invalid value” error is thrown and user has to do it again.

**REFERENCES**

Grosan, C. & Abraham, A. (2011). Intelligent systems: A modern approach (Vol. 17). Berlin: Springer. doi:10.1007/978-3-642-21004-4.

* Chapter 8: “Managing Uncertainty in Rule Based Expert Systems” (pp. 187–217).

Alcala-Fdez, J., & Alonso, J. M. (2016). A survey of fuzzy systems software: Taxonomy, current research trends, and prospects. IEEE Transactions on Fuzzy Systems, 24(1), 40–56. doi:10.1109/tfuzz.2015.2426212.

Belohlavek, R., Kruse, R., & Moewes, C. (2011). Fuzzy logic for computer science. Retrieved from