Application of fuzzy algorithms to find possible cases of COVID-19

Leonardo Juárez-Morales
Engineering Division, Campus Irapuato - Salamanca, University of Guanajuato.
l.juarezmorales@ugto.mx

Abstract — This document will analyze and explain the necessary process for the creation and modeling of a system based on fuzzy algorithms used for the analysis of symptoms for possible cases of COVID-19.

I. Introduction

The system that we built was thought about the current situation in the world, a preventive system to identify possible cases of this disease from different symptoms related to the virus. It is not an official diagnosis of contracting the disease, but it is an alert to prevent and invite the public to take the test. At present we can know many things about the virus, the different symptoms that we can have and the danger of these to a greater or lesser extent, having an idea of the risk of our actions with society and the probability of contracting the disease.

II. Content Development

The main objective of the project will be to model ta system using fuzzy algorithms to find a percent in accordance with specific data range of a possible infection.

For this we will focus on an initial basic structure and with it we will follow a fuzzy algorithm adding some rules and comparative data to find a percent relation of the possible disease.

A. *Essential facts.*

For a better application of the method used, we collect some data related to the COVID-19 disease caused by SARS-COV-2, which appear as various symptoms that may indicate the presence of said disease.

The data collected, as well as the information used to structure the purpose of this Project have been obtained from specialized sources in the field of health.

Before getting into the algorithm, we must know a little about the subject in question.

COVID-19 is an infectious disease caused by **SARS-COV-2**, better known as coronavirus. Both were virtually unknown until before the outbreak in Wuhan (China).

The disease is transmitted mainly from person to person through droplets that come out of the nose or mouth of an infected person when coughing, sneezing, or talking. These drops are relatively heavy, do not go very far, and fall quickly to the ground. A person can get COVID-19 if they breathe in these particles from a person infected with the virus.

If someone contracts the disease, they may have different symptoms, some more dangerous than others but all the same importance to find a positive case. *[1]*

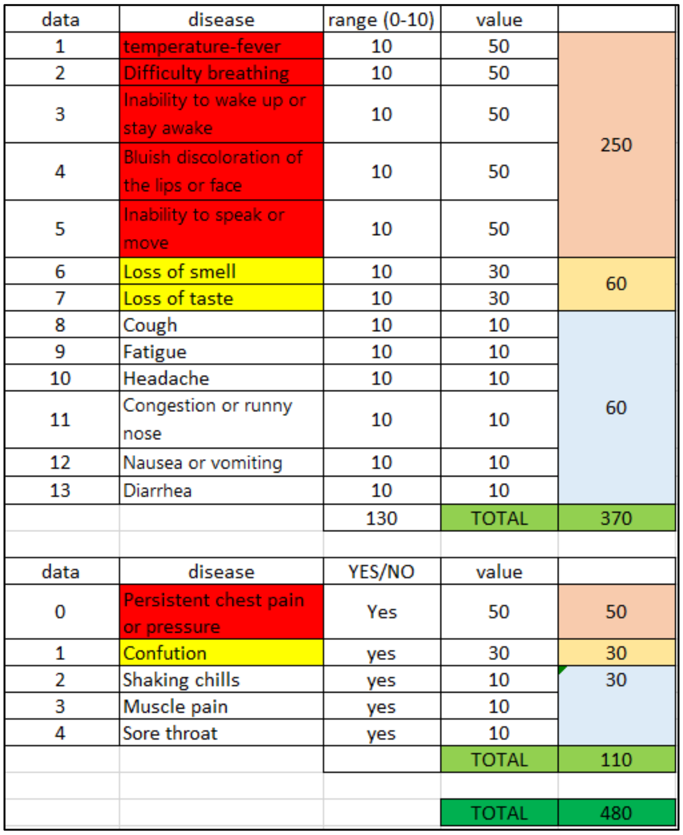
The symptoms that people with this disease present can appear between two and fourteen days after had contact with the virus, this can vary greatly and there may even be cases in which there are no symptoms, but the virus is still in the system, the list of related symptoms corresponds to:

- Fever or chills
- Cough
- Nausea or vomiting
- Diarrhea

- Difficulty breathing
- Fatigue
- Muscle and body aches
- Headache
- Recent loss of smell or taste
- Nausea or vomiting
- Diarrhea
- Recent loss of smell or taste

The list presented above corresponds only to a part of all the symptoms that people with this disease may present, for the realization of this project they have been considerate selectively through a range of risk. *[2]*

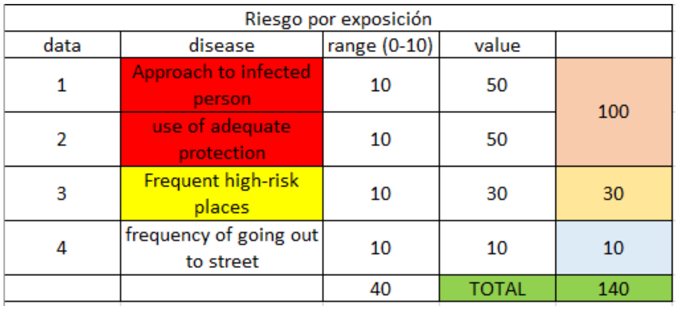
The following table shows the existing symptoms, present in patients who have contracted the virus.



data	disease	range (0-10)	value	
1	temperature-fever	10	50	250
2	Difficulty breathing	10	50	
3	Inability to wake up or stay awake	10	50	
4	Bluish discoloration of the lips or face	10	50	
5	Inability to speak or move	10	50	
6	Loss of smell	10	30	60
7	Loss of taste	10	30	
8	Cough	10	10	60
9	Fatigue	10	10	
10	Headache	10	10	
11	Congestion or runny nose	10	10	
12	Nausea or vomiting	10	10	
13	Diarrhea	10	10	
		130	TOTAL	370
data	disease	YES/NO	value	
0	Persistent chest pain or pressure	Yes	50	50
1	Confution	yes	30	30
2	Shaking chills	yes	10	30
3	Muscle pain	yes	10	
4	Sore throat	yes	10	
			TOTAL	110
			TOTAL	480

Fig. 1 Classification of known symptoms.

The data shown has been divided by its type of presence, either by a range that goes from 0 to 10 or simply by the presence of the symptom, considering a binary response such as "Yes" or "No". They have also been classified by their level of risk (low, medium or high), according to the considerations made by health institutions and published in various media.



Riesgo por exposición				
data	disease	range (0-10)	value	
1	Approach to infected person	10	50	100
2	use of adequate protection	10	50	
3	Frequent high-risk places	10	30	30
4	frequency of going out to street	10	10	10
		40	TOTAL	140

Fig. 2 Risk factor's.

Various risk factors have been included, based on a range from 0 to 10 depending on the frequency of the acts that are normally carried out. Likewise, a risk categorization of activities has been considered given their probability of contagion.

As can be seen, each of these classifications has a final sum to the score of the symptoms or risk factors, these being:

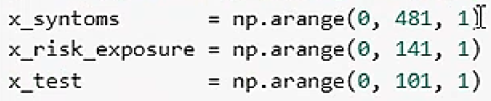
final sum of symptoms	**= 480**
final sum of risk factors	**= 140**

B. *Methodology*

As already mentioned, for this project fuzzy algorithms will be used to find relationships between the cases given by the symptoms present in different subjects to obtain a coherent result to know if said subject is a possible carrier of the disease.

The team has taken the "tipping problem" as a basic example to develop a logic that follows different rules from fuzzy methods to obtain a complex result. *[3]*

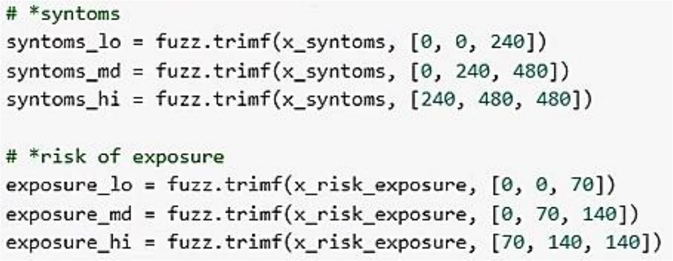
To begin with the implementation of the algorithm, the input variables are proposed to generate the ranges that will be considered as membership functions.



```
x_syntoms          = np.arange(0, 481, 1)
x_risk_exposure = np.arange(0, 141, 1)
x_test             = np.arange(0, 101, 1)
```

Fig. 3 Input variables.

These membership functions are generated from ranges obtained from the total sum of each of the symptoms, grouping them by risk according to the score limit obtained in the sum.



```
# *syntoms
syntoms_lo = fuzz.trimf(x_syntoms, [0, 0, 240])
syntoms_md = fuzz.trimf(x_syntoms, [0, 240, 480])
syntoms_hi = fuzz.trimf(x_syntoms, [240, 480, 480])

# *risk of exposure
exposure_lo = fuzz.trimf(x_risk_exposure, [0, 0, 70])
exposure_md = fuzz.trimf(x_risk_exposure, [0, 70, 140])
exposure_hi = fuzz.trimf(x_risk_exposure, [70, 140, 140])
```

Fig. 4 Membership functions.

To graph these functions, the ranges assigned to these functions have been considered. In the case of the function belonging to symptoms, the subcategories by level of risk have been considered according to the input score obtained, the limit being 480 points.

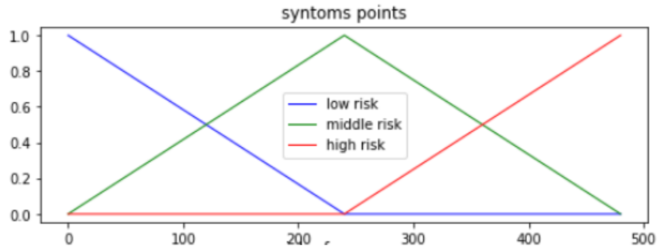


Fig. 5 Symptom function.

As in the previous graph, the function of the risk factors is plotted according to the ranges assigned to the subcategories, the limit being 140 points.

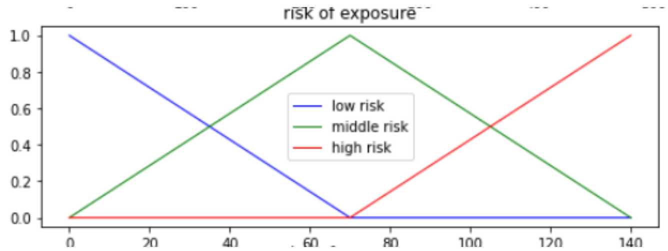


Fig. 6 Risk factors function.

Once these membership functions have been established, an output function called "test" is created in which we can view the results obtained in detail and be able to categorize the risk level in a higher way after adding them.

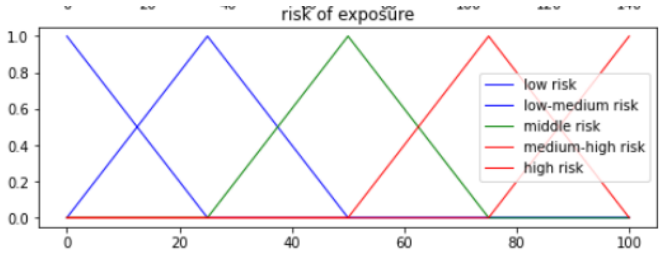


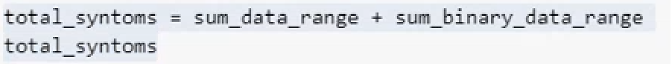
Fig. 7 Output function.

Once the initial structure has been built, we move on to reading external data, where the user will provide various responses to the symptoms that it presents, this will be read from an external document and later with the help of specialized functions the data mapping will be carried out, said mapping will help to obtain the sum of the data stored in a previously defined array.

The stored data is evaluated according to its risk level as follows:

Low risk (x1)
Medium risk (x3)
High risk (x5)

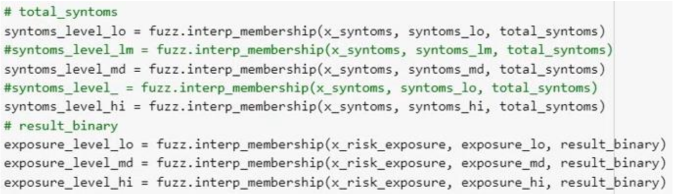
It should be noted that the stored data will be divided into **data by range** and **binary data**, in such a way that the evaluation of the data is carried out separately and repeating the process depending on its level of risk. Finally, the total symptom score is obtained by adding the two data sets.



```
total_syntoms = sum_data_range + sum_binary_data_range
total_syntoms
```

Fig. 8 final sum.

Once the total results of the symptoms and risk factors have been obtained, these results are analyzed with the help of a specialized function *[3]* in such a way that they are interpreted and adjusted to a subcategory of membership functions, in this way we can give the explicit conditions and obtain an approximate result in relation to the amount of data.



```
# total_syntoms
syntoms_level_lo = fuzz.interp_membership(x_syntoms, syntoms_lo, total_syntoms)
#syntoms_level_lm = fuzz.interp_membership(x_syntoms, syntoms_lm, total_syntoms)
syntoms_level_md = fuzz.interp_membership(x_syntoms, syntoms_md, total_syntoms)
#syntoms_level_ = fuzz.interp_membership(x_syntoms, syntoms_lo, total_syntoms)
syntoms_level_hi = fuzz.interp_membership(x_syntoms, syntoms_hi, total_syntoms)
# result_binary
exposure_level_lo = fuzz.interp_membership(x_risk_exposure, exposure_lo, result_binary)
exposure_level_md = fuzz.interp_membership(x_risk_exposure, exposure_md, result_binary)
exposure_level_hi = fuzz.interp_membership(x_risk_exposure, exposure_hi, result_binary)
```

Fig. 9 Data interpretation.

After the comparison and application of specific rules, the method uses defuzzification to convert the data interpreted by the algorithm into a simple real number for easy use.

This is accomplished with the Data System centroid or gravity point defuzzification method.

This method calculates the y coordinates of the center of gravity of the area under the fuzzy set B':

$$y' = centroide(B') = \frac{\sum_{j=1}^{m} \mu_{B'}(y) \cdot y_j}{\sum_{j=1}^{m} \mu_{B'}(y)}$$

Fig. 10 Defuzzification method.

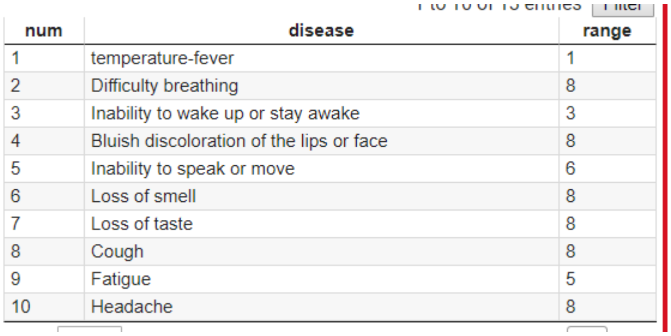
where m is the number of elements and J in Y. *[4]*

III. Tests.

After having the structure and the system ready, several tests were carried out with different data obtained at random to verify the functionality and stability of the system in question.

These data were aggregated in .CSV files that, separately and being three files in total, gave us a more expanded view for data analysis.

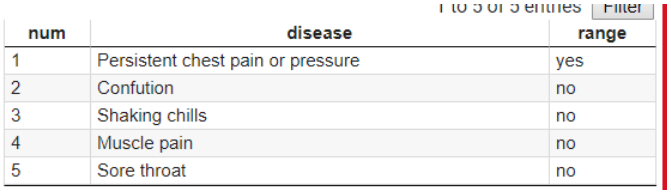
The first batch of data corresponds to person No. 5, who presents different symptoms, but not all, and in a different range of affectation. Below you can see the different data entered for that person.



num	disease	range
1	temperature-fever	1
2	Difficulty breathing	8
3	Inability to wake up or stay awake	3
4	Bluish discoloration of the lips or face	8
5	Inability to speak or move	6
6	Loss of smell	8
7	Loss of taste	8
8	Cough	8
9	Fatigue	5
10	Headache	8

Fig. 10 Symptoms by rank for person 5.

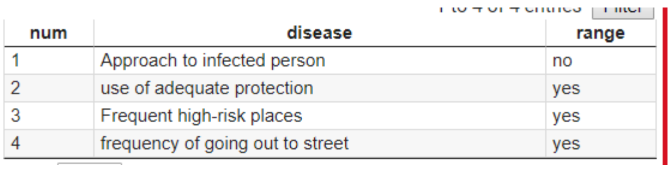
Data of person 5, the symptoms that he presents with his range of affectation.



num	disease	range
1	Persistent chest pain or pressure	yes
2	Confution	no
3	Shaking chills	no
4	Muscle pain	no
5	Sore throat	no

Fig. 11 Symptoms by binary data range of person 5.

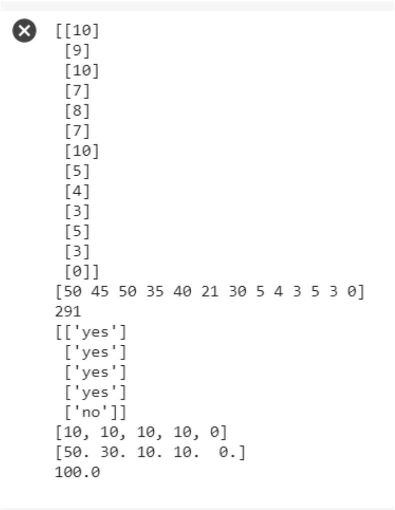
These data cannot have a defined range since there is no way to evaluate them, for this reason it has been chosen to assign binary values for "yes" or "no".



num	disease	range
1	Approach to infected person	no
2	use of adequate protection	yes
3	Frequent high-risk places	yes
4	frequency of going out to street	yes

Fig. 12 Risk factors by binary data range of person 5.

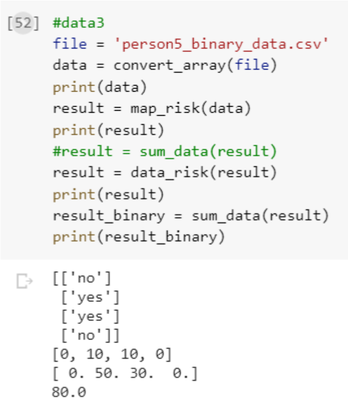
Once the data has been entered into the system, it maps them and displays them in an orderly way and performs the sums of all the data by symptoms.



```
[[10]
 [9]
 [10]
 [7]
 [8]
 [7]
 [10]
 [5]
 [4]
 [3]
 [5]
 [3]
 [0]]
[50 45 50 35 40 21 30 5 4 3 5 3 0]
291
[['yes']
 ['yes']
 ['yes']
 ['yes']
 ['no']]
[10, 10, 10, 10, 0]
[50. 30. 10. 10.  0.]
100.0
```

Fig. 13 Ordered data.

The program analyzes the data presented by risk factors, interprets them and returns the sum of these.



```
[52] #data3
     file = 'person5_binary_data.csv'
     data = convert_array(file)
     print(data)
     result = map_risk(data)
     print(result)
     #result = sum_data(result)
     result = data_risk(result)
     print(result)
     result_binary = sum_data(result)
     print(result_binary)
```

```
[['no']
 ['yes']
 ['yes']
 ['no']]
[0, 10, 10, 0]
[ 0. 50. 30.  0.]
80.0
```

Fig. 14 Ordered data by risk factors.

Once the System makes the sum of these data, it proceeds to apply the rules to verify the subcategory of the same and subsequently, it interprets them and applies a defuzzification method so that a coherent result is thrown for a clear understanding.

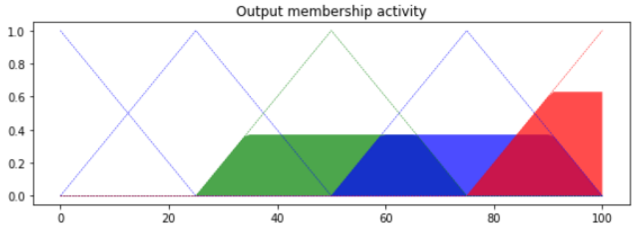


Fig. 15 Final concentration of plotted data.

Subsequently, we proceed to graph and visually show a data concentrated in the areas of greatest concentration, leading to the final part.

IV. RESULTS

The System gives us satisfactory results, giving a clear sample of the scope of the data entered and the percentage of success it presents in relation to what was entered.

```
79.87817381966815
```

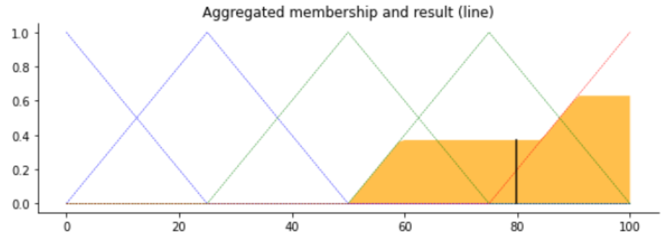


Fig. 16 Final results.

In the previous figure we can see how the System gives us an estimated percentage of effectiveness, indicating the probability that the user may be a carrier of the virus, this is reflected in the graph, in the form of a signal set in the area with the highest concentration of data or matches found as analyzed.

V. CONCLUSION.

The Project has given us a clear and precise idea of the scope of fuzzy algorithms, as well as their handling of data and resolution of problems that other types of methods cannot solve, this has led to various questions, such as the different applications in the industry and how is it that these types of algorithms can go unnoticed by most people, being something fantastic and fascinating for the study of artificial intelligence.

VI. REFERENCES.

[1] Preguntas y respuestas sobre la enfermedad por coronavirus (COVID-19). (s. f.). Recuperado 7 de diciembre de 2020, de

https://www.who.int/es/emergencies/diseases/novel-coronavirus-2019/advice-for-public/q-a-coronaviruses#:%7E:text=sintomas

[2] CDC. (2020). Síntomas de la enfermedad del coronavirus. december 11th 2020, de CDC website:

https://espanol.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html#nav-group-62e05

[3] scikit-fuzzy development team. (-). Fuzzy Logic Toolbox for Python.. 11 de diciembre de 2020, de scikit-fuzzy development team website:

https://pythonhosted.org/scikit-fuzzy/api/skfuzzy.html?fbclid=IwAR1PS4kRuwsWDsMjHveVKacwDbUjhO62B7-tgwEViY5PtIw3eReht0BHE2M

[4] Juan Pedro Azcona. (2014). Modelo Fuzzy de determinación del valor unitario de edificación destinada a vivienda con fines catastrales. 11 de diciembre de 2020, de Catastro website:

http://www.catastro.meh.es/documentos/publicaciones/ct/ct82/1.pdf