**(7135CEM)**

**Modelling and Optimization under Uncertainty**

**Vaccine Myths using Topic Modelling LSA and LDA Techniques**

**Abstract:**

The importance of vaccines in maintaining community health today makes it crucial to understand the types of misconceptions that surround them and their sources. Such myths may be categorised into many categories to assist us comprehend the myth's genesis. In order to successfully combat such misconceptions and persuade those who believe them to be false, it is crucial for individuals to comprehend the scientific and medical community. A statistical modelling technique called topic modelling could be used to identify the general "themes" that appear in a group of texts. A topic model such as Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) is used to categorise text in a document to a certain subject.

**Introduction:**

Vaccinations are one of the most significant advancements in public health in recent history. Around the globe, they have saved and are still saving millions of lives. Because of their effectiveness, few patients, parents, and healthcare professionals in high-income environments have first-hand knowledge of the devastation caused by many of the illnesses they prevent. Vaccine hesitancy has become a significant public health issue in recent years, contributing to measles epidemics and other contagious diseases. Childhood vaccinations are a tradition, but not only for children. Even older persons must prepare themselves for infections that may be avoided, especially those like influenza and pneumonia. The government's advised immunisation regimen isn't being followed by many individuals, however. According to a CDC assessment published in January, adult people 65 and older had alarmingly low immunisation rates. Pneumococcal vaccination rates among seniors were only 62%, tetanus vaccination rates were little over 50%, and shingles vaccination rates were just 15%.

Vaccines do not really result in an illness, but they do help your immune system identify and fight infections. A genetic material thread known as messenger RNA, or mRNA, is present in both the Moderna and Pfizer COVID-19 vaccines. Your cells are instructed to produce a portion of the "spike" protein, which is found on the coronavirus that causes COVID-19, when the mRNA reaches your cells. Although those protein fragments aren't directly harmful to your body, they do cause your immune system to respond by mounting an attack toward them off. The DNA for the spike protein is given by a different, harmless form of virus in the Johnson & Johnson COVID-19 vaccination. After receiving the vaccination, individual might have weariness, muscular pains, a headache, or a fever. This is typical after any vaccination and indicates that our immune system is working properly.

**Dataset:**

I have taken reddit\_vm.csv file from Kaggle website. People could debate numerous vaccine myths in the Reddit community Vaccine Myths. The postings were not filtered, thus there may be a tiny percentage of coarse language in the data. Reddit postings from the Vaccine Myths subreddit, obtained utilizing from

<https://www.kaggle.com/code/renjithrrkj/topic-modeling-vaccine-myths-lda-and-lsa/data>

Posts and comments are both included in data. Score for the title, which is pertinent to the postings, is based on the significance and the volume of comments.

**Methods:**

**Latent Dirichlet Allocation (LDA)**

We sorting our materials into subjects using the Latent Dirichlet Allocation (LDA). A topic model such as Latent Dirichlet Allocation (LDA) is used to categorise text in a document to a certain subject. It creates a Dirichlet distribution-modelled topic per document and words per subject model.

It employs techniques for data analysis that are applicable to machine learning or natural language processing. This method takes into account the fact that papers are a compilation of several themes and that topics are just collections of words. LDA's main objective is to find such topics in the publications. If the user enters, for instance, 10, and LDA searches the text to find those subjects from the collection of words, every meaningful word from this subject may be used to determine its associated topic.

Documents over latent subjects and topics over words have the same probabilistic models. In order to choose a word from the multivariate statistical topic word distributions for a given topic, it first selects a problem from the document’s topic distribution. LDA employs the reverse process of creation when analysing the document themes. Later, it begins by giving each word a random subject assignment before going through multiple rounds to improve the topic assignments.

**Latent Semantic Analysis (LSA):**

LSA, a similar topic modelling method to LDA, counts the number of times a word appears in a text and makes the assumption that texts with similar content have the same distribution. In phrases like LDA, the semantic meaning of the term is disregarded because to its position as a bag of words. LDA and LSA are different in that LDA assumes the topic distribution and the topic distribution inside a document, while LSA just represents subjects and documents generally.

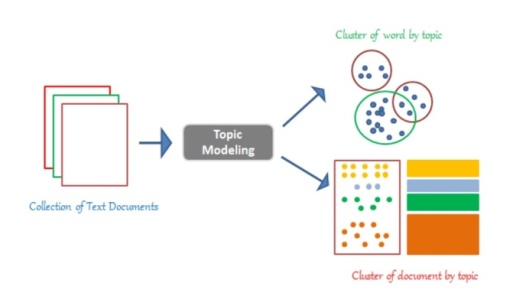


Figure 1: LSA sample view

**Experimental Setup:**

**Pre-process:**

This pre-processing is done before utilising the method to examine the data. The process begins with importing some of the libraries that will provide easier access. This analysis method imports pandas and is dependent on the string in the str property. This will make the procedure's processing simpler. After the module has been imported, the training data set is subsequently assigned to the data frame. The training dataset's training data frame's head is given the following details.

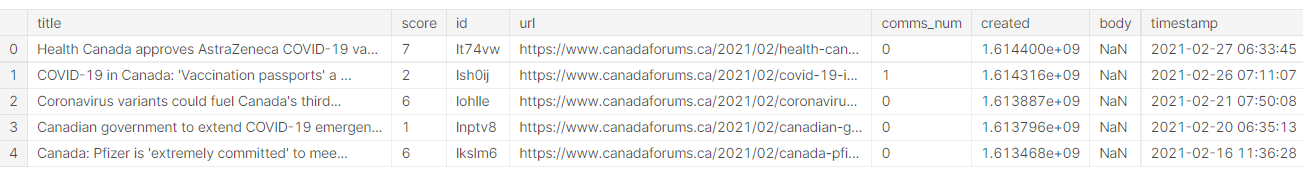


Figure 2: Data header rows

The five rows and 8 columns has been shown.

Next, we have to check the shape of the data using shape function.

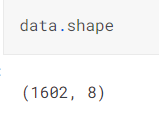


Figure 3: Data shape

We can see that there are 8 columns and 1602 rows overall.

Since it is obvious from the data that many postings are missing their bodies, let's attempt to identify all the missing data.

Tokenization: Sentences are broken up into words

Stop words are eliminated

Lemmatization: Present tense is used for past and future tenses.

Stemming: Words are shortened into their root form by stemming.

After applying the all-pre-processing steps successfully, we get the below result:

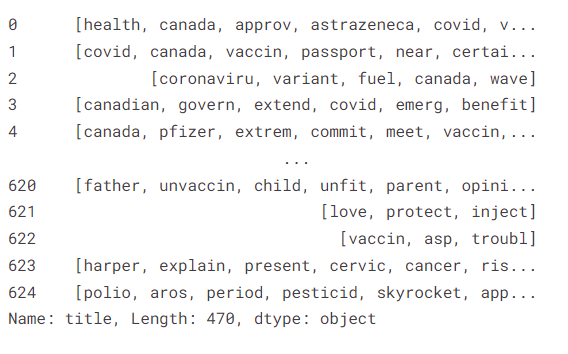


Figure 4: After pre-processing data

Stop words are should remove before to production in order to prevent unexpected outcomes, So after removing all stop words we get the graph that is produced after the top words from the dataset are generated provides us with some further information about the vocabulary used.

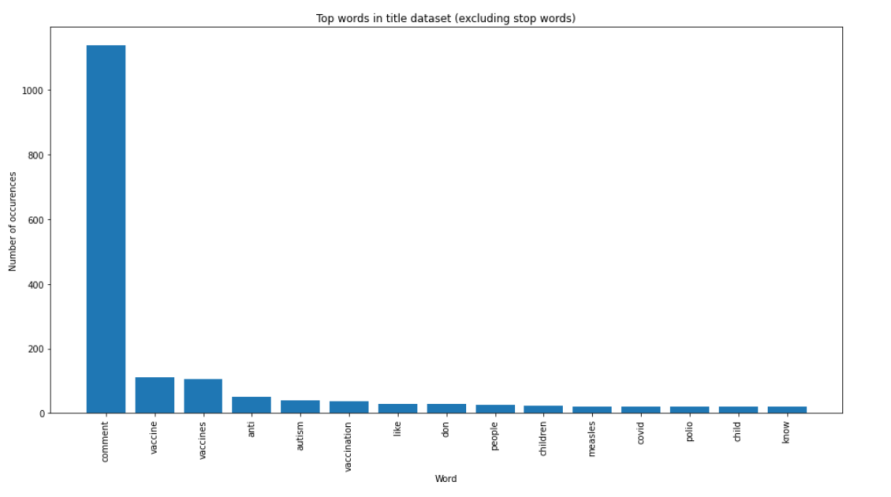


Figure 5: Top words in the dataset

The total number of words: 8991

Mean number of words per tweet: 5.61235955

Next parts of speech tagging for title corpus

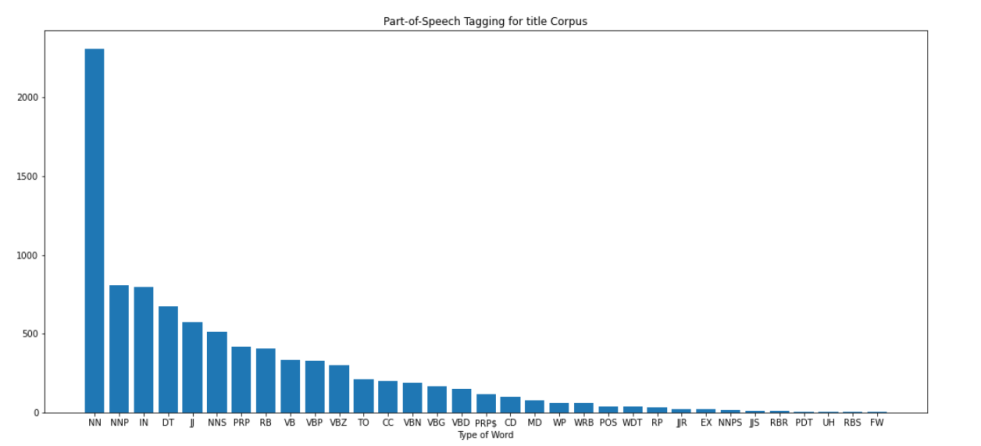


Figure 6: Part of the speech for title corpus

**Topic Modelling:**

Randomly chosen data are collected and processed into a feature space before being sent to our model. Every item of data is converted into an arithmetical vector in order to do this using the Count Vectorizer object from SKlearn. A n\*k doc-term matrix is performed by this entity, where k is the number of unique words around the sample data.

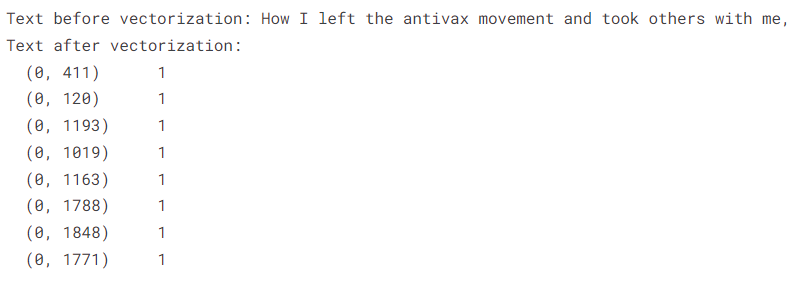


Figure 7: Number of unique words around the sample data

From the graphic above, we can see that our training data is rare and highly ranked. Since doc term matrices could be utilised to build clustering algorithms, LDA and LSA techniques are employed for topic modelling. Both methods utilise our doc word matrices as input and produce n\*N topic matrices, where N is the number of subjects that the user specifies.

Finding the terms that emerge the most before analysing the problem is preferred since it is common for the many themes to lose their significance after they are combined.

**LSA (latent semantic analysis):**

Scikit-Learn, its most powerful and useful machine learning library in Python, is used to provide LSA analysis from a deconstruction of its own library. Since it is not necessary to stay focused the relevant data prior to forecasting the disintegration, sparse matrix multiplication could be effectively managed, and it generally appears to work with vectorizer outcomes like term counts, truncated SVD is executed to our data in order to execute linear measurement of the feature trimming.

It supports two approaches; in this case, randomised is employed, and a quick SVD solver and count vectorizer are used in place of Tfidvectorizer. Utilizing a count vectorizer will result in unique words that are unequally sampled, which is advantageous for the topic distribution.

The following plot shows the some of the LSA topics.

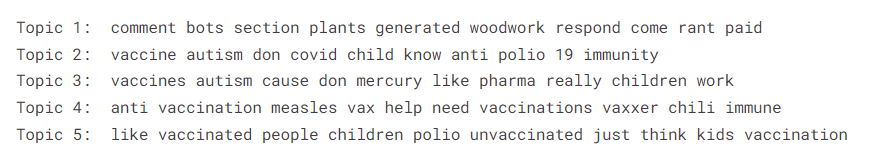


Figure 8: LSA Topics

Starting out, n=5 inputs were given, resulting in 5 different individuals.

Word clouds are used to analyse and visualise the most prevalent terms in each subject to provide a clearer picture.

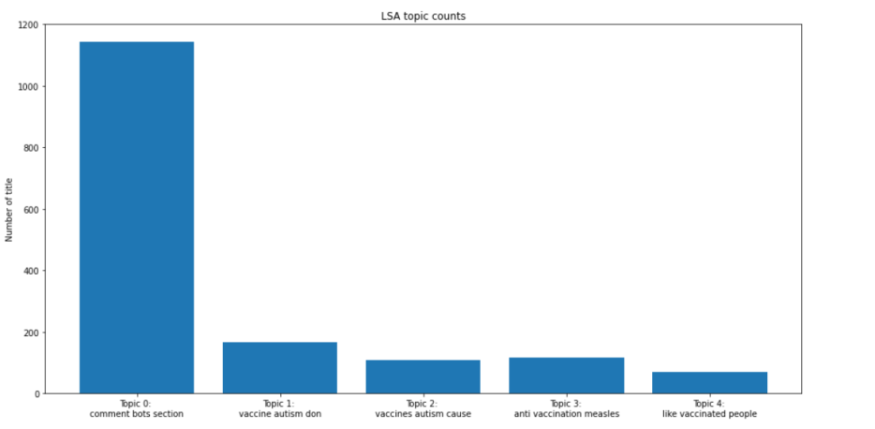


Figure 9: LSA Topic Counts

**LDA topics (latent Dirichlet allocation):**

Both LSA and LDA studies are carried out using the Python software scikit-learn, and our data analysis does actually make advantage of this module's deconstruction latent Dirichlet allocation. If the training set is large, the operation is performed using online finite difference iterative techniques, which are much faster than batch systems. Each cycle utilises a very small batch to train the data. From the data set we have supplied; it is helpful for picking themes. In this scenario, count vectorizer is utilised instead of Tfidvectorizer since LDA is based on phrase count and document count. Digitization has a positive impact on the final topic distribution since it generates unique words that are disproportionately impacting the sample. The random state parameter should ideally be kept to a limited value since LDA is a classification technique.

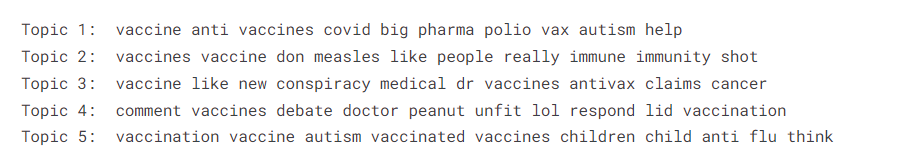


Figure 10: LDA Topics

Starting out, n=5 inputs were given, resulting in 5 different individuals.

Word clouds are used to analyse and visualise the most prevalent terms in each subject to provide a clearer picture.

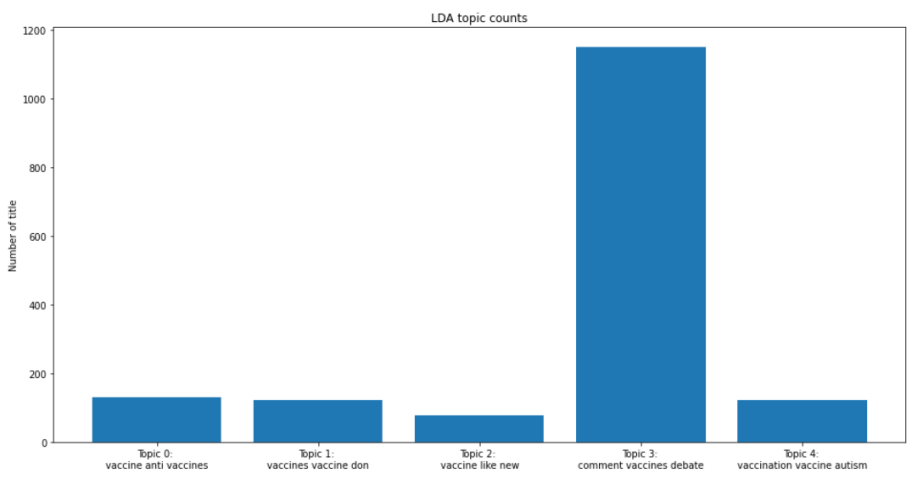


Figure 11: LDA Topic Counts

Visualizing LDA results

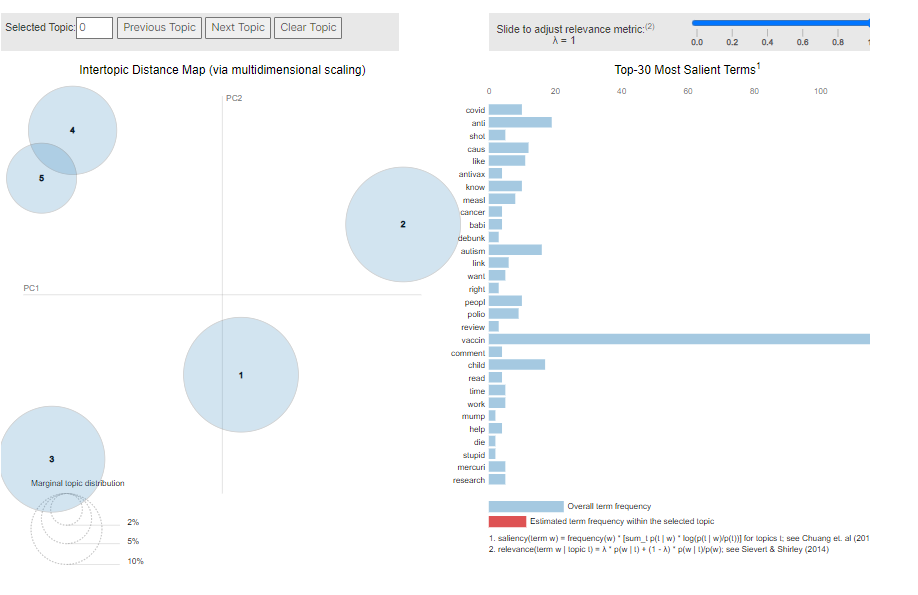


Figure 12: LDA Visualisation

It is evident that the LDA model was effective in quickly constructing each of the 5 clusters.

Clusters 1 and 2 barely slightly overlap one another.

By just examining the themes, it is difficult to compare the outcomes of both LSA and LDA models. Visualizations aid in comparison, which is why T-SNE clustering was used. Below, both models' clusters are shown, and their respective outcomes are compared.

t-SNE clustering of both LSA and LDA 5 topics are shown below:

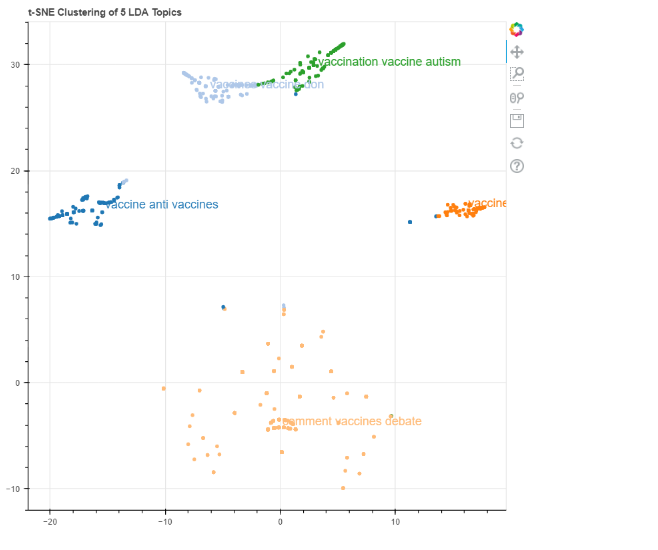
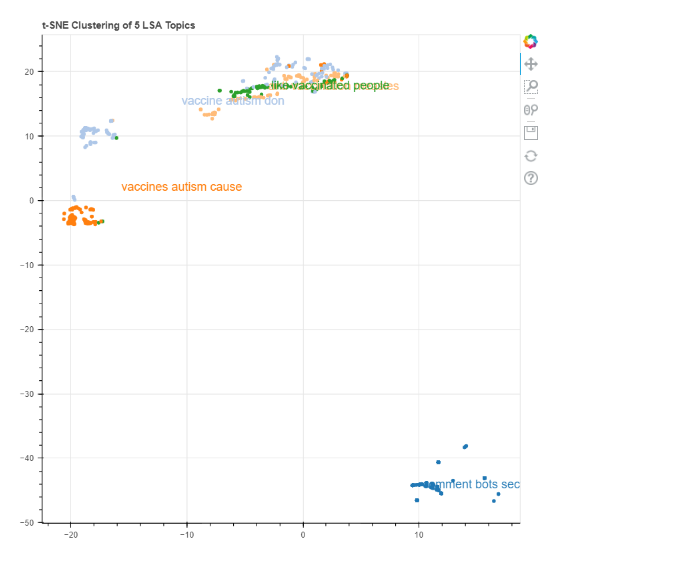


Figure 13: t-SNE clustering of both LSA and LDA

**Social, ethical, legal and professional considerations:**

The dataset was obtained from a publically accessible source, Kaggle, thus all social, ethical, legal, and professional considerations are taken into account. No information, such as a userid or name, that belongs to a specific user is utilised. All of these details were disregarded and not taken into account while creating the model.

**Conclusion:**

Findings show that LDA performs in topic and word differentiation a little bit better than LSA. The emphasis of Topic 1 is on comment bots’ section while Topic 5 focuses on like vaccinated people, vaccinated measles etc. For the benefit of vaccine myths who is vaccinated, these topics enable us to categorise all of the vaccine myths and comments by genre. The results will differ for various types of datasets and demands since both procedures make different assumptions and take different approaches to solutions; the procedure is outlined in the preceding sections. As previously stated, I originally suggested the dataset used for this exercise to the instructor, and I was told that it may be used again but in a different method.

**Task 2**

**Fuzzy Logic Optimized Controller (FLC) for Care Environment**

**Abstract:**

In this project, the fuzzy logic controller for the Assistive Clinical Area is built by giving the regulators the proper input and providing feedback to the inference mechanism. Mamdani fuzzy is used in this case, and mathematical formalism is used to create the rules.

**Introduction:**

In this project, our environment is a resident apartment where persons with disabilities will be living. This is known as an intellectually appropriate care environment. In order to care for these handicapped individuals, there are numerous residential care organisations nowadays, and a significant amount of money is spent on these services. As we all know, disabled persons have trouble arranging their possessions when living in their apartment. The need for human capital is decreasing as technology develops on a daily basis.

To build flat intelligence, researchers need to maintain some logical thinking, hence in this situation, a fuzzification controller may be used. As two separate types of extrapolations used by this system, Sugano fuzzy systems and Mamdani are used; Mamdani is implemented and carried out using the fuzzified toolkit MATLAB in this work.

**Background Analysis:**

There are several approaches to build this environment. To increase the accuracy of the results, several techniques could be utilised and the inputs, outputs, and rules can all be modified. Due to resource constraints and desired results, structures may sometimes be difficult since they do not always follow a clear plan. A water heater, air conditioner, kitchen appliances, fire detectors, motion detectors, auto lighting, presence sensors, physical sensors, and many other items that the normal person needs to live in a home may all now be made utilising gadgets. Electronic sensors can operate all of these gadgets since they are all powered by electricity, which is quite beneficial, particularly for those who are disabled.

**Assistive Care Environment Fuzzy design:**

We can use all of the electrical components that are linked with each other in an utilizing a variety to operate sensibly for the comfort of people with disabilities, but this is still reliant on requirement specification. Let's take the case of a resident who has certain requirements for his environment: Motor moisture management equipment that dehumidify if there is an abundant supply of water content inside the apartment, auto air conditioning units that include heating systems and cooling systems depending on the temperature of the flat, and perhaps most importantly, since residents are disabled people, it is crucial to check their own health issues like hypertension, oxygen level, and heartbeat. Any peculiar events affecting residents should be brought to the attention of medication control.

The MATLAB fuzzy logic - based system toolbox is used in this project to run an optimization method. This fuzzy controller method evaluates the data and transmits it to fuzzy logic implication, which yields outcomes. Each input is sent to fuzzy implication, which processes it in accordance with pre-established rules. Inference is followed by fuzzy rules to provide the desired results.

Extrapolations comprises rules for merging input and output, and computational complexity also contains model parameters for the output. The portion of the portfolios of a controller using fuzzy logic has membership values that are independently of all other inputs.

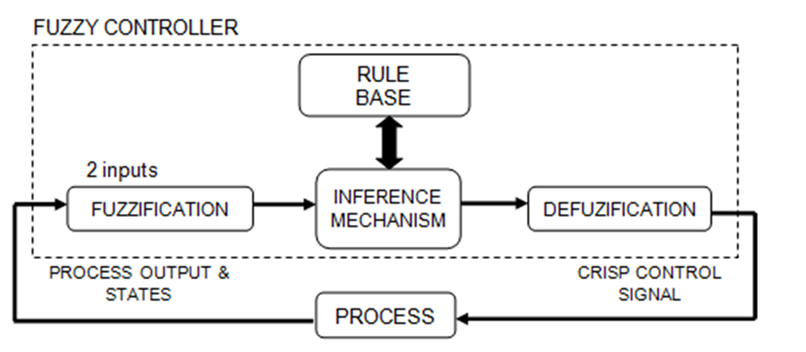


Figure 1: Fuzzy Controller

**Temperature:**

All people need a stable temperature in their homes, thus regardless of whether a heater or air conditioner is functioning; our thermometers monitor the ambient temperature. The perfect temperature in this scenario is between 11.7 and 17.3 degrees Celsius. The ideal temperature varies based on the person and their place of origin. This leads to the identification of the VL, L, M, H, and VH membership functions, which are then represented as triangular and trapezoidal forms.

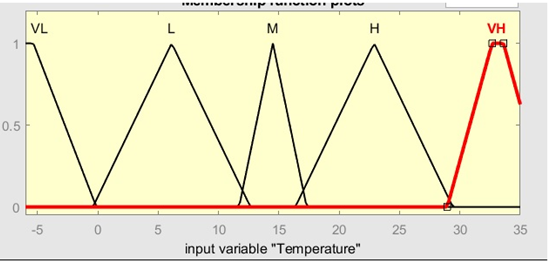
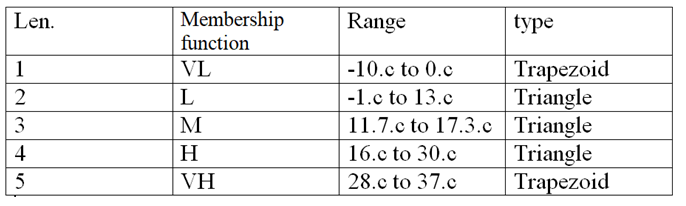


Figure 2: Temperature input variable



**Humidity:**

If we live close to the ocean or in a region with high humidity, we could use a dehumidifier because these circumstances make it difficult for individuals to reside there, destroy objects like furniture and wallpaper, and also encourage the spread of microbes.

Membership events are planned with an usual humidity range of 20% to 50% in mind. Extreme, very high, high, and average. This is represented by trapezoidal and triangle functions.

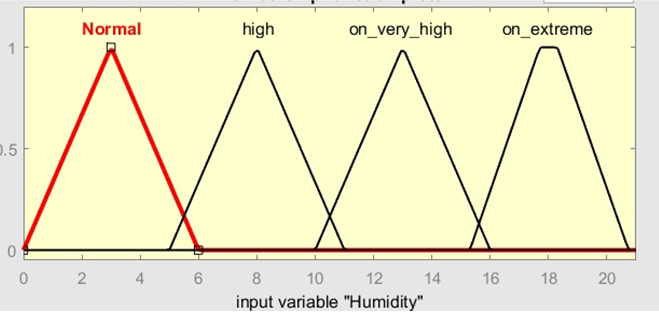
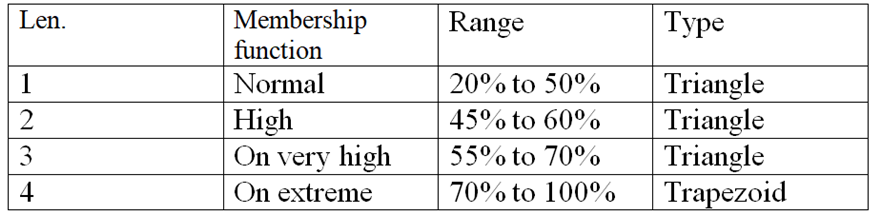


Figure 3: Humidity input variable



**Light:**

In intelligent workplaces, auto light bulbs are preferred since they are practical for all sorts of individuals. The lights are turned on and the brightness is adjusted based on how much light is present within the apartment. Its five membership values, denoted by a triangle and a triangular member function, are VL, L, M, H, and VH.

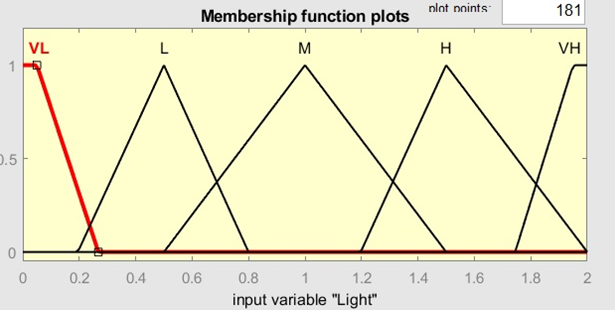
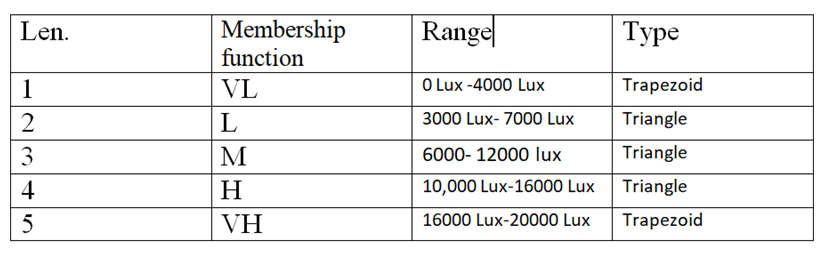


Figure 4: Light input variable



**Sensor:**

Since it is ideal to switch off some of the home appliances while no one is home to save on energy expenditures, a presence sensor must be included when building technological applications. Here, just the triangle membership function and the missing and presence membership functions are used.

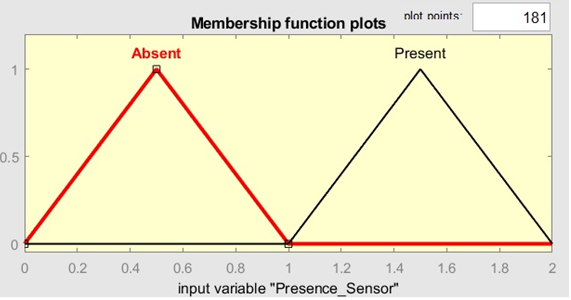
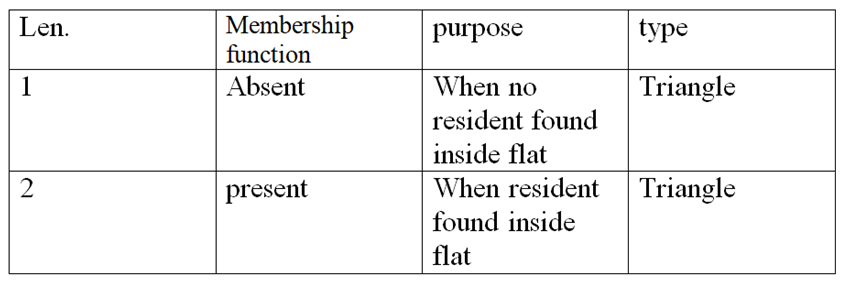


Figure 5: Presence Sensor input variable



**Blood Pressure:**

This is an extra input for healthcare apps, allowing us to immediately alert them if anything is wrong. This interface uses a trapezoidal and a triangular activation function to represent its three membership functions, L, M, and H.

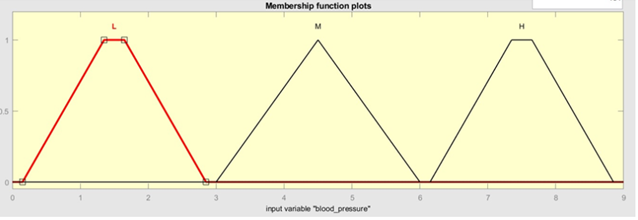
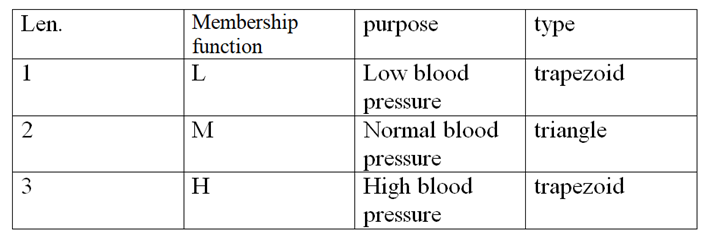


Figure 6: Blood pressure input variable



**Fuzzy Interface System:**

Our system's principal makes use of the following, Mandani inference, which was discussed before, was designed with this in mind. The key indications should be carefully considered while creating a successful fuzzy logic system's input and output.

The original application of this reasoning, which was developed by EBhasim Mamdani in 1975, was to examine the operation of a steam engine and its boilers.

Stages of development for our activity:

1. Induction rules are created in the first step.

2. Inputs will be converted to fuzzy inputs by applying degree of membership.

3. Analysing the efficacy of rules from unclear sources using fuzzy sets

4. Analysing the outcome using result participation and rule intensity approaches.

5. Analyse output distributions using the findings of the rule effectiveness and membership function.

6. The last step is to get defuzzied results.

The following MATLAB screen demonstrates how the fuzzy toolbox in MATLAB was used to complete this Mamdani Fuzzy inference:

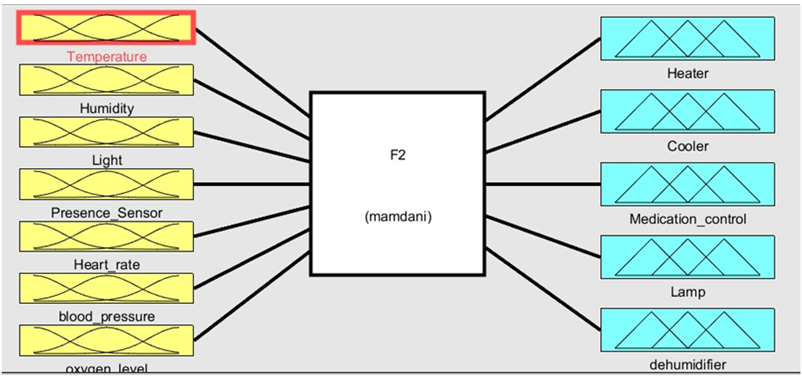


Figure 7: Fuzzy toolbox in MATLAB

**Outcomes:**

**Heater:**

The heaters are also not configured to warm the space if the temperature is normal, they are programmed to do so if the temperature is low.

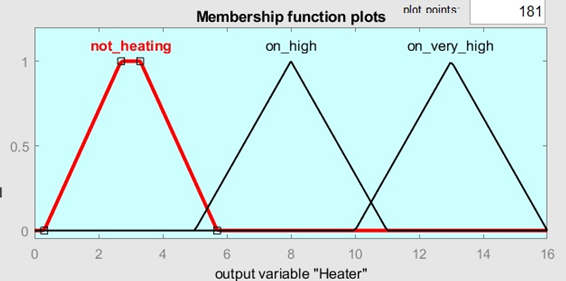
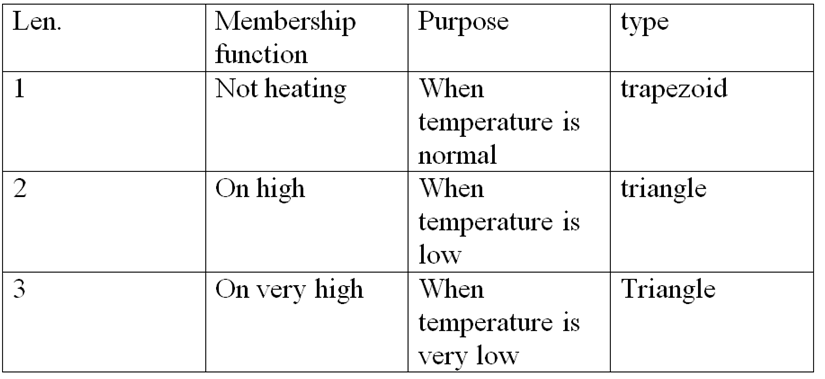


Figure 8: Heater output variable



**Cooler:**

Coolers might well be compared using the similarity metrics of not cooling, on high, and on extremely high. The cooler is configured to not cool while the ambient temperature is normal; when it is high, it is changed suitably.

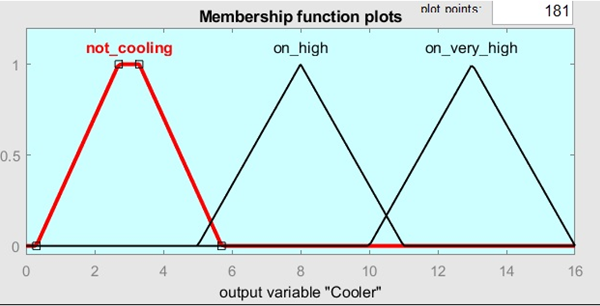
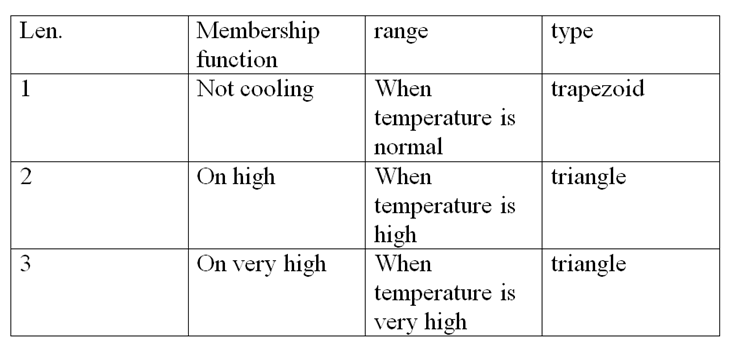


Figure 9: Cooler Output variable



**Dehumidifier:**

The auto dehumidifier operates based on inputs of humidity; the measures used here include ordinary, high, very high, and extreme.

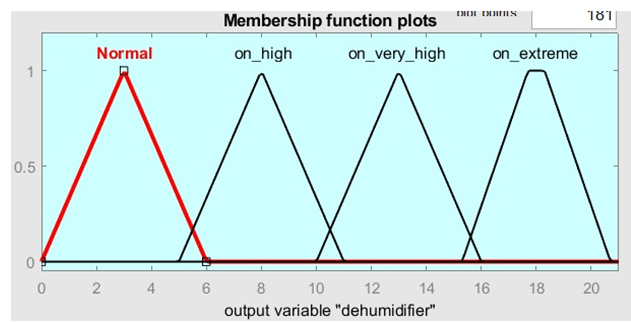
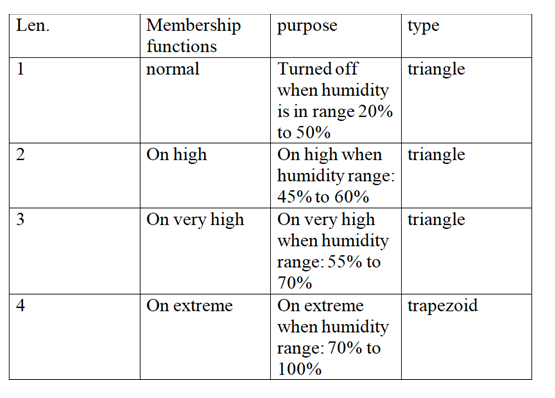


Figure 10: De-humidifier output variable

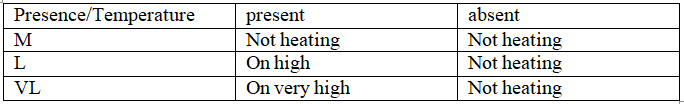


**Fuzzy Rules:**

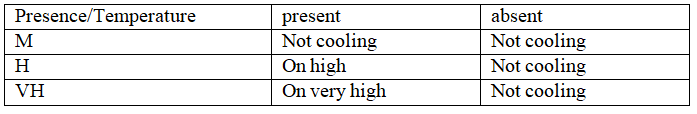
These rules are put into action by communicating the needs of the occupants of the apartment, and they are intended to achieve the appropriate outcomes based on one or more circumstances.

They may perform several operations using a variety of variables because they have t norms or t-conorms, where t-norms utilise AND operators and t-conorms use OR operations.

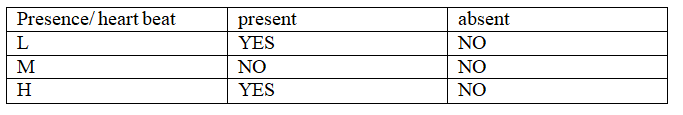
Heater and Temperature sensor:



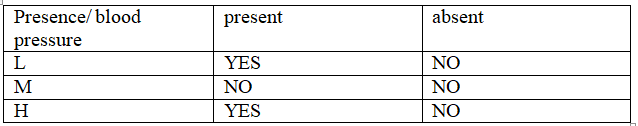
For cooling, a temperature and presence sensor



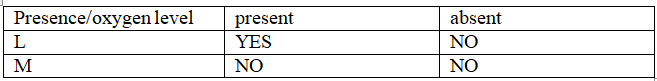
Medication control from heart beat and presence sensor:



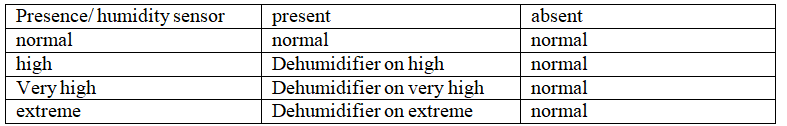
Medication control from blood pressure and presence sensor:



Medication control from oxygen level and presence sensor:



Dehumidifier from humidity sensor and presence sensor:

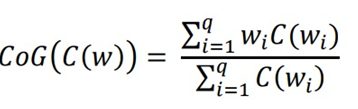


**Aggregation:**

When combining all rules, all outputs are cropped, and if the compressed membership function has two components, the highest power is taken into consideration.

**Defuzzification:**

The process of producing outputs from restrictions and inputs is known as defuzzification. To achieve this, a variety of tactics are used, including the Center of Area, Min of Progression of the disease, Adaptive Integration, Center of Gravity, and others. The formulation of the Center of Area approach is as follows:



In order to use this formula, parameters must be established and weighting must be taken into account. Q stands for the discontinuous points, and w for the amount of w at ipoint.

The following graphic shows it after defuzzification:



Figure 11: After De-fuzzification

The following are the graphs relating input and output:

Heater vs. Temperature, Presence

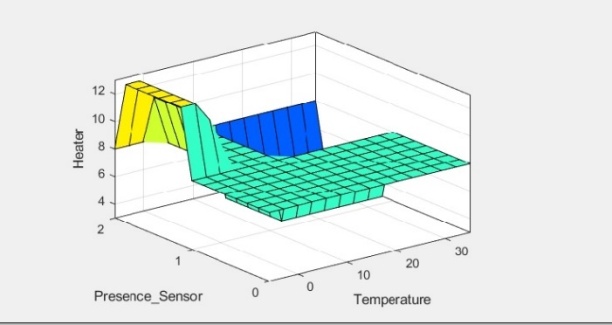


Figure 12: Heater vs. Temperature, Presence

Heart rate, presence, and medication management

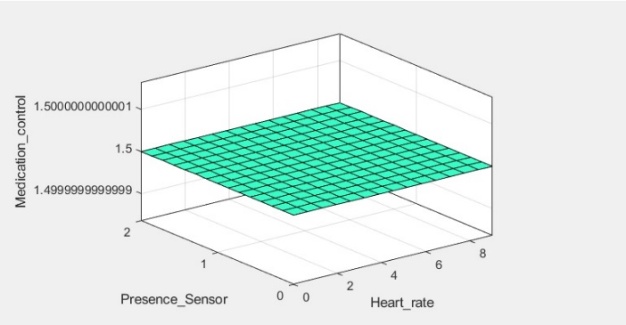


Figure 13: Heart rate, presence, and medication management

Humidity, presence sensor vs dehumidifier

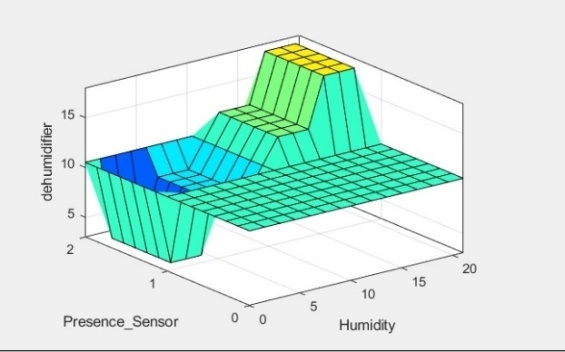


Figure 14: Humidity, presence sensor vs dehumidifier

Light sensor, presence sensor vs lamp:

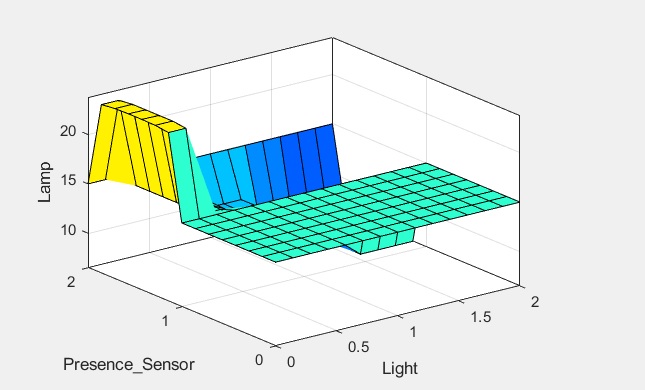


Figure 15: Light sensor, presence sensor vs lamp:

**Part 2: Optimizing the FLC developed for Part 1:**

It is possible to improve the fuzzy logic controller created in Part 1 using genetic algorithms.All of the membership function arguments must first be converted to binary form.

Every digit 0 or 1 represents a gene, and all together they make up a chromosome. Populations of chromosomes are referred to collectively.

In comparison to the triangle membership function, the trapezoidal membership function has four parameters. There are four functions for input membership and five for output membership in section 1.

Thus, there are [(4+3+3+3+4) + (4+3+3+3+4) + (4+3+3+4) + (4+3+3+4)]= 65 chromosomes in aggregate for the inputs.

Therefore, there are 58 chromosomes in total for the outputs: [(4+3+3+4) + (4+3+3+4) + (4+3+4) + (4+3+4) + (4+4)].

The fitness value is then employed to determine how well-suited for mating each chromosome is. The ability to make children will be determined by the fitness score each chromosome receives from this function.

Choosing the stopping criterion is the next step. At this stage, all the parameters, including crossover rate, mutation rate, and population size, should be chosen in order to carry out genetic operations including selection, crossover, and mutation.

The chromosomes with the highest fitness scores will be chosen using the roulette wheel approach, depending on the fitness score. It will be possible to create new chromosomes using these chromosomes.

After crossover, the process ends with mutation, which involves changing any gene that is randomly chosen from the chromosome. Until optimised results are produced or a halting condition is met, the algorithm continues to run.

**Comparison:**

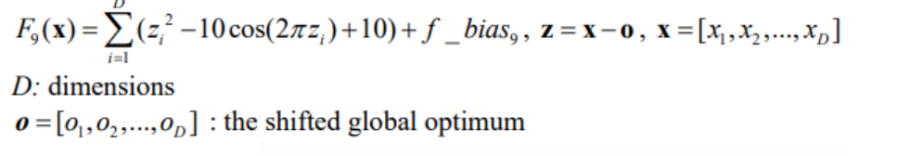
Sugeno is computationally more efficient, performs well together with linear approaches like PID control, and is ideally suited to mathematical analysis, but Mamdani's presentation is user-friendly, well-suited to human input, and extensively utilised.

Since Sugeno Induction is less regulation than Mamdani Inference, it is less subject to interpretation. As a result, Mamdani was chosen as the interaction process.

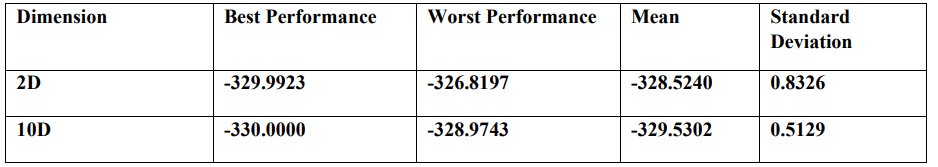
**Part-3: Comparing different optimization techniques on CEC’2005 functions**

In this section, the fundamental CEC'2005 benchmark functions F9: Started shifting Rastrigin's Function and F10: Shifted Rotated Rastrigin's Function are subjected to two optimization techniques—Genetic Algorithm Optimization and Particle Swarm Optimization. 15 iterations of the optimization procedure are performed. Finding the global minima for parameters is the goal.

**Shifted Rastrigin’s Function:**

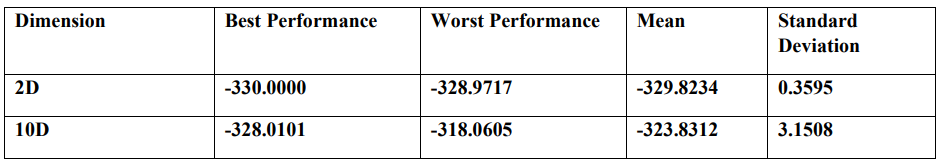


f\_bias9= - 330 Results with Genetic Algorithm:



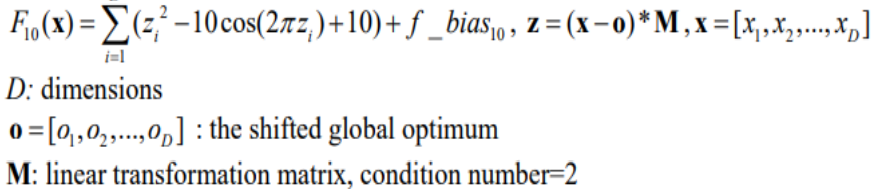
Shifted Rastrigin’s Function Results Summary with Genetic Algorithm

Results with Particle Swarm optimization Algorithm:

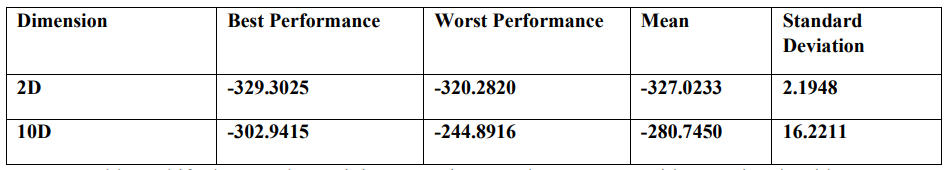


Shifted Rastrigin’s Function Results Summary with PSO

Shifted Rotated Rastrigin’s Function: Formula:

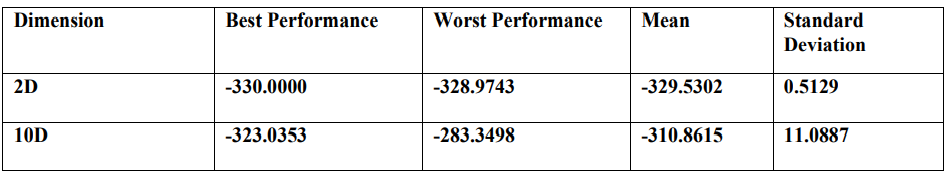


f\_bias10= - 330 Results with Genetic Algorithm:



Shifted Rotated Rastrigin’s Function Results Summary with Genetic Algorithm

Results with Particle Swarm optimization Algorithm:



Shifted Rotated Rastrigin’s Function Results Summary with PSO

It is evident from the preceding tables that when there are just two dimensions, PSO outperforms GA for the Shifted Rastrigin's Function. However, when the dimensions are 10, the genetic algorithm has a lower standard deviation. The intended global minimum is getting closer.

When the number of dimensions is 2 or 10, the performance of the particle swarm optimization algorithm is greater for the function F10—Shifted Rotated Rastrigin's Function.

From the aforementioned findings, it can be inferred that these two algorithms perform really well since their outputs are almost identical to the target values for these functions given in the literature, indicating that both algorithms are appropriate for solving any optimization problem that arises in the real world.

**References:**

Weng, J. (2021, December 11). *NLP Text Preprocessing: A Practical Guide and Template*. Medium. <https://towardsdatascience.com/nlp-text-preprocessing-a-practical-guide-and-template-d80874676e79#:%7E:text=Text%20preprocessing%20is%20traditionally,learning%20algorithms%20can%20perform%20better>

Foong, N. W. (2021, December 17). *Introduction to Topic Modeling using Scikit-Learn - Towards Data Science*. Medium. <https://towardsdatascience.com/introduction-to-topic-modeling-using-scikit-learn-4c3f3290f5b9>

Kapadia, S. (2019). Topic Modeling in Python: Latent Dirichlet Allocation (LDA). Medium. Retrieved 13 December 2021, from

[https://towardsdatascience.com/end-toend-topic-modeling-in-python-latent-dirichlet-allocation-lda-35ce4ed6b3e0](https://towardsdatascience.com/end-to%20end-topic-modeling-in-python-latent-dirichlet-allocation-lda-35ce4ed6b3e0)

Li, S. (2018). Topic Modeling and Latent Dirichlet Allocation (LDA) in Python. Medium. Retrieved 13 December 2021, from [https://towardsdatascience.com/topic-modelingand-latent-dirichlet-allocation-in-python-9bf156893c24](https://towardsdatascience.com/topic-modeling%20and-latent-dirichlet-allocation-in-python-9bf156893c24)

**Appendix:**

**Data Set Link:**

<https://www.kaggle.com/code/renjithrrkj/topic-modeling-vaccine-myths-lda-and-lsa/data>

**Task 1:**

**Source Code Link:**

**Task 2 Code:**

[System]

Name='Fuzz2'

Type='mamdani'

Version=2.0

NumInputs=7

NumOutputs=5

NumRules=20

AndMethod='min'

OrMethod='max'

I\_Method='min'

Agrgat\_Method='max'

Dfuz\_Method='centroid'

[Input1]

Name='Temperature\_control'

Range=[-6 45]

NumMFs=4

mf1='VL':'trimf',[-10.66 -6 0.5227]

mf2='L':'trimf',[-0.4091 6.114 12.64]

mf3='M':'trimf',[11.7 14.5 17.3]

mf4='H':'trimf',[16.36 22.89 29.41]

[Input2]

Name='Humidity'

Range=[0 22]

NumMFs=3

mf1='Normal':'trimf',[0 3 6]

mf2='high':'trimf',[5 8 11]

mf3='on\_very\_high':'trimf',[10 13 16]

[Input3]

Name='Light\_cnt'

Range=[0 2]

NumMFs=5

MF1='VL':'trapmf',[-0.45 -0.05 0.05 0.267540478026215]

MF2='L':'trimf',[0.195 0.5 0.799537393986122]

MF3='M':'trimf',[0.5 1 1.5]

MF4='H':'trimf',[1.19737856592136 1.5 2]

MF5='VH':'trapmf',[1.74479568234387 1.95 2.05 2.45]

[Input4]

Name='Pr\_sense'

Range=[0 2]

NumMFs=2

MF1='Absent':'trimf',[0 0.5 1]

MF2='Present':'trimf',[1 1.5 2]

[Input5]

Name='Heart\_rate'

Range=[0 10]

NumMFs=2

MF1='L':'trapmf',[0.15 1.35 1.65 2.85]

MF2='M':'trimf',[3 4.5 6]

[Input6]

Name='blood\_pr\_check'

Range=[0 10]

NumMFs=2

MF1='L':'trapmf',[0.15 1.35 1.65 2.85]

MF2='M':'trimf',[3 4.5 6]

[Input7]

Name='oxgn\_lvl'

Range=[0 9]

NumMFs=2

MF1='normal':'trimf',[0 3 6]

MF2='low':'trimf',[4 7 10]

[Output1]

Name='Heater'

Range=[0 16]

NumMFs=3

MF1='not\_heating':'trapmf',[0.3 2.7 3.3 5.7]

MF2='on\_high':'trimf',[5 8 11]

MF3='on\_very\_high':'trimf',[10 13 16]

[Output2]

Name='Coolr'

Range=[0 16]

NumMFs=3

MF1='not\_cooling':'trapmf',[0.3 2.7 3.3 5.7]

MF2='on\_high':'trimf',[5 8 11]

MF3='on\_very\_high':'trimf',[10 13 16]

[Output3]

Name='Medication\_control'

Range=[0 2]

NumMFs=2

MF1='No':'trimf',[0 0.5 1]

MF2='Yes':'trapmf',[1.05 1.45 1.55 1.95]

[Output5]

Name='dehumidifier'

Range=[0 21]

NumMFs=4

MF1='Normal':'trimf',[0 3 6]

MF2='on\_high':'trimf',[5 8 11]

MF3='on\_very\_high':'trimf',[10 13 16]

MF4='on\_extreme':'trapmf',[15.3 17.7 18.3 20.7]

[Rules]

1 0 0 2 0 0 0, 3 1 0 0 0 (1) : 1

2 0 0 2 0 0 0, 2 1 0 0 0 (1) : 1

3 0 0 2 0 0 0, 1 1 0 0 0 (1) : 1

4 0 0 2 0 0 0, 1 2 0 0 0 (1) : 1

5 0 0 2 0 0 0, 1 3 0 0 0 (1) : 1

0 1 0 2 0 0 0, 0 0 0 0 1 (1) : 1

0 2 0 2 0 0 0, 0 0 0 0 2 (1) : 1

0 3 0 2 0 0 0, 0 0 0 0 3 (1) : 1

0 4 0 2 0 0 0, 0 0 0 0 4 (1) : 1

0 0 1 2 0 0 0, 0 0 0 4 0 (1) : 1

0 0 2 2 0 0 0, 0 0 0 3 0 (1) : 1

0 0 3 2 0 0 0, 0 0 0 2 0 (1) : 1

0 0 4 2 0 0 0, 0 0 0 1 0 (1) : 1

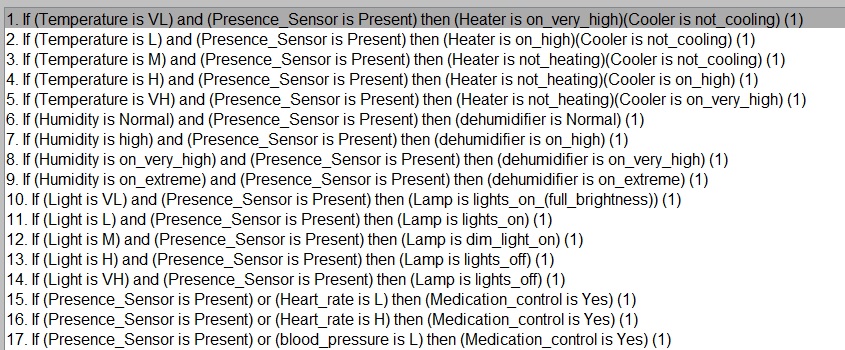
0 0 5 2 0 0 0, 0 0 0 1 0 (1) : 1

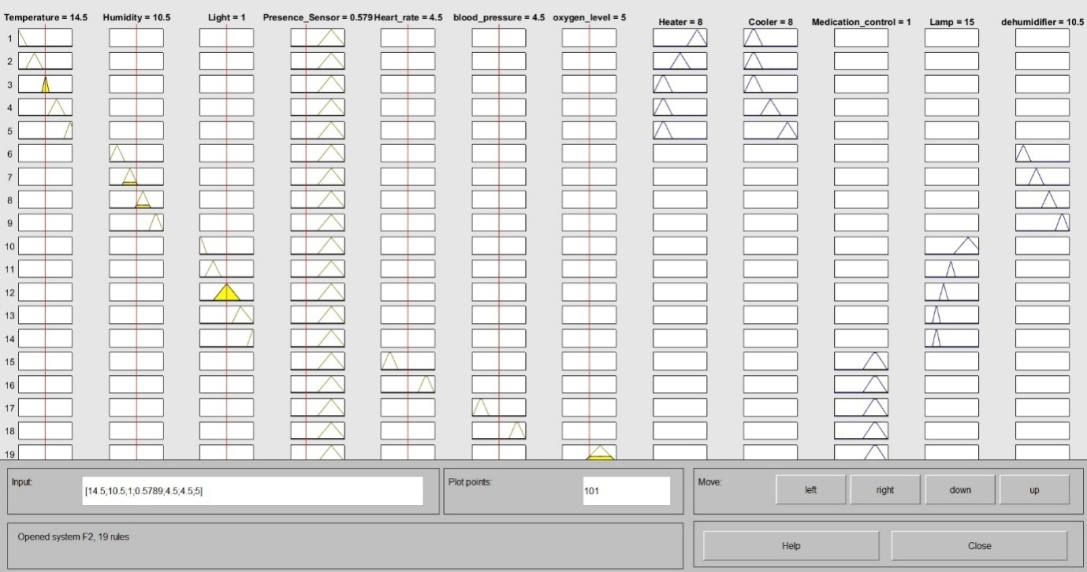
0 0 0 2 1 0 0, 0 0 2 0 0 (1) : 1

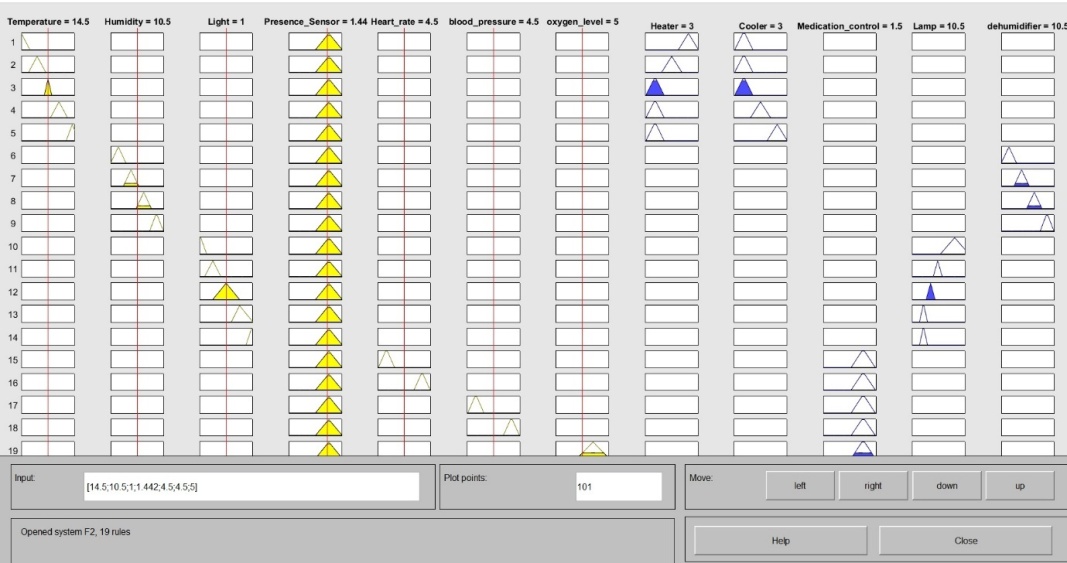
0 0 0 2 3 0 0, 0 0 2 0 0 (1) : 1

0 0 0 2 0 1 0, 0 0 2 0 0 (1) : 1

**Snapshots:**

****

****

****