***COVID-19 Symptoms Diagnosis***

**ABSTRACT Corona virus, rather known as COVID-19, has taken the lives of many people all around the globe. With an increase in demand for methods to identify the virus, numerous scientists and doctors have been researching how to discover the virus in humans, as a result, a tremendous amount of data has been collected regarding everything about the virus since 2020. This encouraged software engineers and computer scientists to training the data on Machine Learning and Deep Learning algorithms, which has produced programs and applications able to predict the infection status of individuals. This study focused on using an algorithm to predict the infection status of individuals who seek immediate medical advice through simple chatbot communication.**

**I. Introduction**

The COVID-19 pandemic has had a significant impact on the world, leading to numerous deaths and illnesses. To better understand this virus, a study has been conducted of the various factors related to COVID-19, including the symptoms associated with it. One approach is the use of machine learning algorithms to help identify individuals who may have COVID-19 based on their symptoms.

One of the main goals of this study was to develop a Machine Learning (ML) model that can accurately classify individuals as either having COVID-19 or not based on several symptoms, such as a sore throat, breathing problems, dry cough, and more. To achieve this goal, a dataset was collected involving information about individuals who have tested positive for COVID-19 as well as those who haven’t. In addition to the model, a chatbot was coded with the capability to interact with a patient and offer useful information related to the virus, as well as informing the patient whether they are likely to have corona virus or not.

Another goal of this study was to prevent the fast spread of the virus by using the chatbot as it could serve as an invaluable tool for healthcare professionals in identifying potential cases of the virus accurately through their symptoms, or to encourage seeking medical assistance as soon as possible in the event of an uncertain case.

In the following section, all related work to this study is unveiled. Section 3 has detailed information about the dataset, the ML model and the chatbot along with all of its functionalities. Section 4 reveals the results acquired and test cases of the chatbot. Finally, the paper is concluded in Section 5.

**II. Literature Review**

Salah Sammari performed training on the same dataset using a different algorithm, Logistic Regression. The author chose to train on 4348 rows and test on 1087. A very good accuracy of 96.5% was achieved in this case. [1]

Salah Sammari used another algorithm as well for training, The random forest regressor. This algorithm had the same training and testing size as the previous one but with a slightly less accurate result. An accuracy of 89.6% was achieved here. [1]

One more algorithm that Salah Sammari used was the Naïve Bayes algorithm. With the same training and testing sizes, the algorithm didn’t exceed an accuracy of 77.4%. [1]

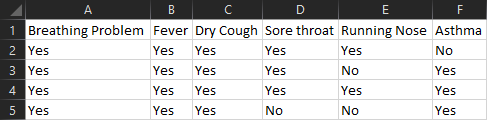
**III. Proposed Technique**

**1. Overview**

The project was essentially several stages of work implemented into one program: the chatbot. The chatbot was developed from an expert system which was created from the decision tree constructed from our dataset. The chatbot has the ability to answer COVID-related questions and ask about symptoms to predict if a patient is infected or not.

**2. Dataset**

**2.1 Data**

The dataset was composed of 5434 binary cases of patients with varying symptoms and conditions such as breathing issues, sore throats and asthma. In total, it added up to 20 symptoms and 1 column as the label for whether patients were infected or not with COVID.

**Figure 1: COVID Dataset Snippet**

**2.2 Preprocessing**

Using Microsoft’s application Power BI, some preprocessing was applied to the dataset to be then applied onto the decision tree algorithm using information gain as the main method for calculations.

The dataset was practically perfect upon acquisition, therefore the only preprocessing accomplished was changing the header rows and identifying the label column.

**3. Decision Tree Algorithm**

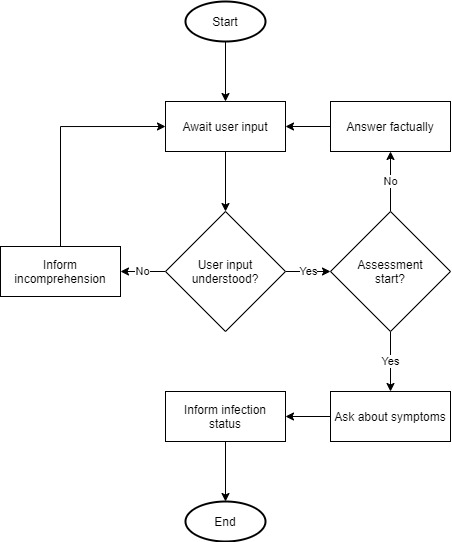
The information gain represents the amount of entropy removed during splitting of a node, the higher the information gain, the more entropy was removed, this criterion was the decided method for splitting the decision tree as it proposed a more compelling result compared to the gain ratio method. The used program for executing the algorithm was RapidMiner. The tree was then used to create an ES-Builder Web expert system which could be accessed at any time.

**4. Chatbot Program**

After the decision tree was extracted from RapidMiner and used in an expert system, it was converted to a string format to be easily read and used in a programming language to run the chatbot.

The chatbot’s program was written purely in Python programming language to ensure both speed and simplicity of execution. The program contained dictionaries of possibly asked questions with their respective answers. Phrases carrying the same meaning were taken into consideration as well, meaning that the chatbot could understand if the user replied with simply a “no” or if they had written the more slang version as “nah”. This ensures that users can chat with the bot feeling as if it were a human being, reducing the possible stress or anxiety induced from communicating with non-humans.

A Graphical User Interface (GUI) was developed using Tkinter (Tk Interface), minimizing complexity to prevent drawing the user’s attention away from their chat. This was important to keep in mind due to the possibility of wrong answers or misunderstood questions, so keeping the user’s attention on the chat was essential for accurate results.



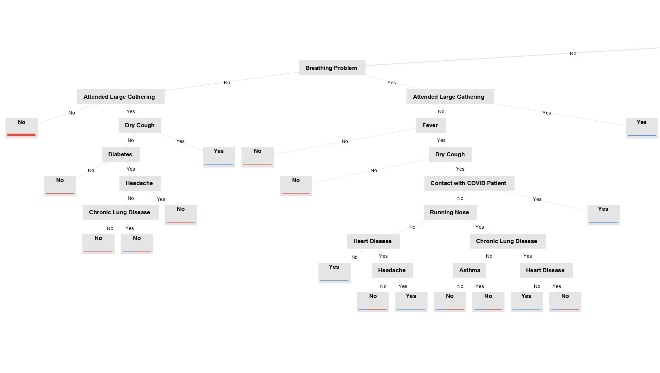
**Figure 2: Chatbot Flowchart**

The chatbot started by waiting for any user input to start the conversation. Once the user greeted the bot or asked a question, it would check if the question was part of the known facts or not, assuming it was, it would then check if the statement was requesting an assessment start, in the condition that it was, it would go over the decision tree and ask questions similar to the expert system and inform the user of the infection status at the end of the assessment.

**IV. Results and Discussions**

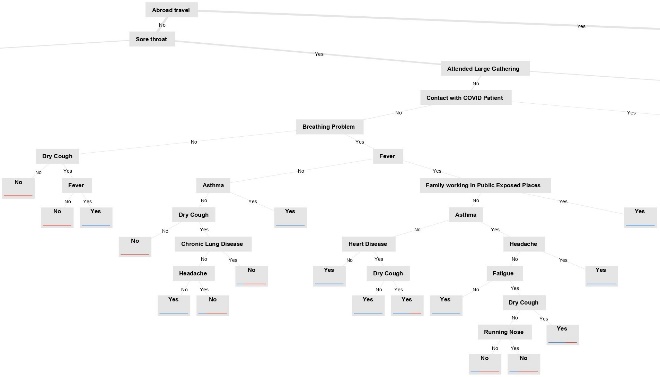
**1. Decision Tree**

The resulted decision tree from RapidMiner was computed using the information gain criterion as it was a more compelling result compared to the gain ratio method.



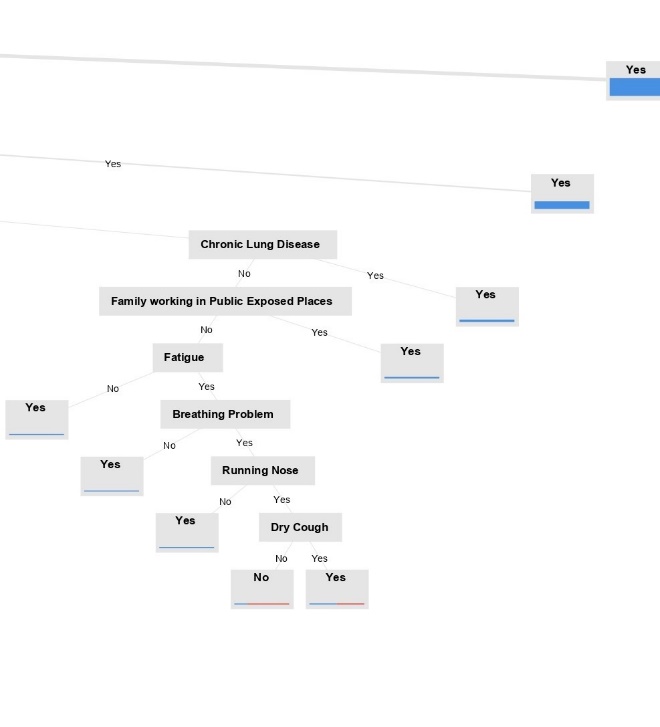
**Figure 3: Decision Tree Part 1**

The first part of the decision tree focused on the possible results in case of not having a sore throat and not having travelled abroad recently.

****

**Figure 4: Decision Tree Part 2**

The second part considered the patient having a sore throat and once again, not having travelled abroad in recent times. This part of the tree had the deepest nodes as most symptoms in this part could relate to many other diseases and infections.

****

**Figure 5: Decision Tree Part 3**

Finally, the third part included mainly the case of having a sore throat, having contact with another COVID-infected patient but not travelling abroad nor attending large gatherings.

All in all, the tree was considerably less complicated in comparison to the tree resulted from the gain ratio criterion. This was considered an advantage to the rest of the study since the more complicated the tree would be, the higher chance of inaccuracy and faulty conditions in the chatbot.

**2. Power BI Statistics**

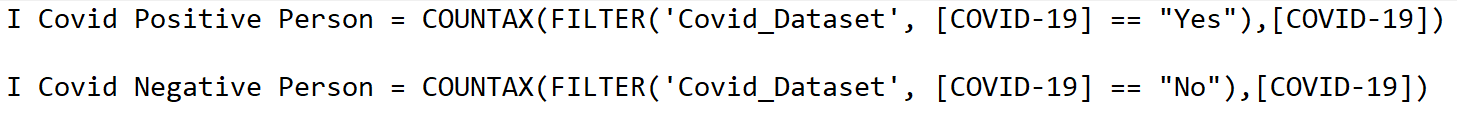
Using Power BI, it was possible to generate graphical representations of comparative data within the dataset.



**Figure 6: Statistical Data in Power BI**

Each of the shapes represented a comparison between the positively infected and negatively infected people with varying conditions.

The first graph was calculated with two Data Analysis Expressions (DAX) commands, which were written to count the total number of infected people compared to the total number of uninfected people in the dataset.



**Figure 7: DAX Functions in Power BI**

**3. Chatbot**

The bot was written in Python programming language for complexity reduction, in addition to simplifying the coding process itself. The chatbot was able to answer COVID-related questions as well as identify the infection status with haste.