DiagnoBot: A Medical Chatbot

Cheyenne Nixon
Amazon Web Services
ms.nixon14@yahoo.com

Benjamin O'Barr
LabWare
obarbarian@gmail.com

Keugmo Gu
keugmo@gmail.com

Abstract

Many people live without access to healthcare or delayed care due to inconvenience, work, cost, living in rural areas, or social/medical fears Gertz et al., 2022, Golembiewski et al., 2022. Medical chatbots have emanated as a potential solution to healthcare access and to promote self-care. We aim to provide medical information through conversations with those who may otherwise delay seeking care. A Rasa chatbot is created using our Disease Prediction System, which utilizes machine learning algorithms i.e., Decision Trees, Gradient Boosting, Support Vector Machine (SVM), and Naïve Bayes to guide users to a sensible diagnosis, so they may opt for self-care at home or seek medical attention. In this paper, a sample of 4920 patient records with 41 disorders are analyzed. The Recursive Feature Elimination algorithm enhances 95 out of the 132 symptom features. Our system achieved 97-100 percent accuracy.

Keywords: Chatbot, machine learning, disease prediction.

1. Introduction

A medical chatbot is a software program that uses artificial intelligence and natural language processing technologies to provide medical assistance and advice to users. Medical chatbots can help users to identify symptoms, suggest possible conditions, and provide information on treatments and modifications, and offer general health and wellness advice. They can also help users to find nearby healthcare providers or schedule appointments.

Medical chatbots can be accessed through various platforms, such as messaging apps, social media, and websites. They are designed to be user-friendly and interactive, providing a conversational interface that mimics human interaction. General trends of medical chatbots include enhancing healthcare services and improving patient experiences. Existing chatbots offer services such as scheduling appointments, locating healthcare facilities, and encouraging clinical trials Luba, 2020. They can interact with patients, ask relevant questions about their symptoms, provide preliminary guidance on potential diagnoses, and introduce the next stop such as making an appointment with healthcare professionals. For example, a chatbot named "About Ada: Personal Health Companion", 2023, guides users to a diagnosis based on their symptoms and provides them with solutions to take next.

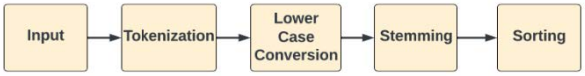

Figure 1. Flow Diagram for Input Processing in Typical NLP Application.

Medical chatbots can learn from user data through machine learning algorithms and adjust their responses to individual preferences. By using AI and NLP technologies, chatbots can become better at understanding and generating human-like language, and it could enable the chatbots to comprehend and respond to user queries more effectively.

The following are the main phases of NLP generally used in chatbot implementation, as indicated in Figure 1.

In the first step, Tokenization, the character stream is converted into a collection of tokens consisting of unique identifiers, words, numerals, or special characters. The words are then converted to lowercase letters. After the lowercase conversion Stemming is

URI: https://hdl.handle.net/10125/107011
978-0-9981331-7-1

applied, which truncates words into simpler root words by using a language dictionary to remove prefixes and suffixes. Finally, sorting will systematically arrange items by categorizing and grouping them by shared properties and then ordering them in a sequence determined by some criterion.

We intend to use a conversational medical chatbot to collect symptoms given by the user to classify a disease and provide relevant medical information about the disease. However, it is important to recognize that medical chatbots are not a substitute for professional medical advice but are instead designed to provide general guidance and information. They should not be used as a substitute for a medical provider nor relied upon for medical diagnosis or treatment. Consultation with a healthcare professional is advised for the evaluation, diagnosis, and treatment of any disease.

2. Related Works

2.1. Medical Chatbots

Several studies have been conducted on multi-class disease-symptoms data sets in the past to predict a user's disease based on their symptoms. Included in many of these studies is the use of the venerable Random Forests algorithm which uses multiple decision trees to vote on a result. For example, in research performed by Swarupa et al., 2021, a medical diagnosis system using Random Forest for disease prediction achieved an accuracy score of 95%. Feature selection was applied to the data set to retain 95 out of 132 features.

In another study by Gupta and Gupta, 2022, a disease prediction system was created using a Decision Tree, Random Forest, K-nearest neighbors, and Naïve Bayes algorithms. A backward selection method was used to perform feature selection on the disease-symptom data set. The accuracy score ranged from 93% to 95% with Decision Tree and Random Forest performing the lowest and K-Nearest Neighbors performing the highest. A Tkinter GUI is used to allow users to select symptoms from a drop-down list and see their predicted diseases.

Jothi et al., 2022, created an AI chatbot to answer common medical questions relating to heart disease. A data set was created by storing many query patterns and their related responses in a JSON file. NLP methods were then used to preprocess the data set into human-like natural language. An Artificial Neural Network (ANN) model was then trained on the processed data and by using 1000 epochs a 99% accuracy was acquired.

Srivastava and Singh, 2020, developed a diagnosis bot to engage patients in conversation for medical queries and to provide an individualized diagnosis. The diagnosis chatbot can identify symptoms from user inputs with a standard precision of 65%. Extracted symptoms were identified to have a recall of 65% and a precision of 71%. Artificial Intelligence Mark-up Language (AIML) based on Extensible Mark-up Language (XML) was used to process a user's message and extract relevant patterns to identify potential diseases.

A paper by Mathew et al., 2019, created a disease prediction chatbot system that uses Python's Natural Language Toolkit (NLTK) to process texts using natural language processing (NLP). NLP allows a chatbot to understand the natural language spoken by humans and return appropriate responses. A user inserts their symptom into a text box. The chatbot then processes the text and shows a result of symptoms that are like the user's input. The user confirms which symptom in the list of symptoms returned by the chatbot accurately matches their symptom. After predicting the disease, the chatbot outputs the result in a similar text box page.

A health chatbot using NLP for disease prediction was also created in a paper by Prayitno et al., 2021. However, cosine similarity is used to compare the user's input with the available symptoms in a database. The most similar symptom is then used in an ID3 decision tree algorithm to find the user's disease. The data set used in this paper is generated from the Alodoketer official website and it contains numerous disease data, including the symptoms and treatments. The chatbot can predict a user's disease based on their symptoms with an accuracy of 87.5%. For future work, P. I. Prayitno would like to increase the accuracy of the model and add additional creatures such as buying medicine with the chatbot application.

2.2. Disease Prediction

The subject of disease prediction classification using machine learning algorithms is widely studied using many data sets and techniques. The following data sets are found in numerous disease prediction papers: the Wisconsin Breast Cancer data set, the Pima Indians Diabetes data set, the UCI Heart Disease data set, and the UCI Thyroid Disease data set. In this section, we will categorize the classification techniques used for each data set.

Random Forest is a popular machine-learning method that is used to detect various diseases. In a paper by Mridha, 2021, the Random Forest technique is applied to the Wisconsin Breast Cancer data set for early prediction of breast cancer. K. Mridha found that the Random Forest model returns the best accuracy

of 98.83%, and K-nearest Neighbors returns the worst accuracy of 91.22%. The Random Forest technique also returns high accuracy scores in other disease data sets. For example, in a paper by Peya et al., 2022, the Random Forest model returned the highest accuracy rate of 99.58% with the UCI Thyroid Disease data set.

Support Vector Machines are another machine-learning method that can be used to detect various diseases. It is shown to return high accuracy rates with the UCI Thyroid Disease data set. In a paper by Tyagi et al., 2018, the SVM model returned an accuracy score of 99.63%.

In a paper by Sisodia and Agrawal, 2020, a method was created for diabetes diagnosis using the PIMA Indians Diabetes data set that includes four main steps, namely missing value handling, instance reduction through k-means clustering, dimension reduction, and classification. Case Deletion or Ignore Missing (IM) and K-means Clustering Imputation (KMI) were found to be the best methods for handling missive values in the Pima Indians Diabetes data set. In the Case Deletion imputed data set after applying PCA, AdaBoost, Bagging, and Multi-layer Perceptron returned the highest accuracy scores of 98.9967%. In the KMI imputed dataset after applying PCA, AdaBoost and Multi-layer Perceptron returned the highest accuracy scores of 99.2767%. Naive Bayes returned the lowest accuracy scores for each data set at 94.3144% and 93.1284% respectively.

Like AdaBoost, Gradient Boost also combines multiple models to improve the performance of a machine learning algorithm. In a paper by Kumar and Reddy, 2021, a heart disease detection system was created using the UCI Heart Disease data set and the Gradient Boost technique. The model returned a higher accuracy score of 97.10% compared to previous techniques such as Naive Bayes, Logistic Regression, and SVM.

3. Proposed DiagnoBot Application

The scope of our chatbot named DiagnoBot is defined below:

- DiagnoBot should be able to understand if the user wants to know their predicted disease based on their symptoms.
- DiagnoBot should be able to understand if the user is inputting their symptoms and reply with follow-up questions to check if the user has any other symptoms as well.
- DiagnoBot should be able to predict the disease based on the user's symptoms and reply with a predicted disease.

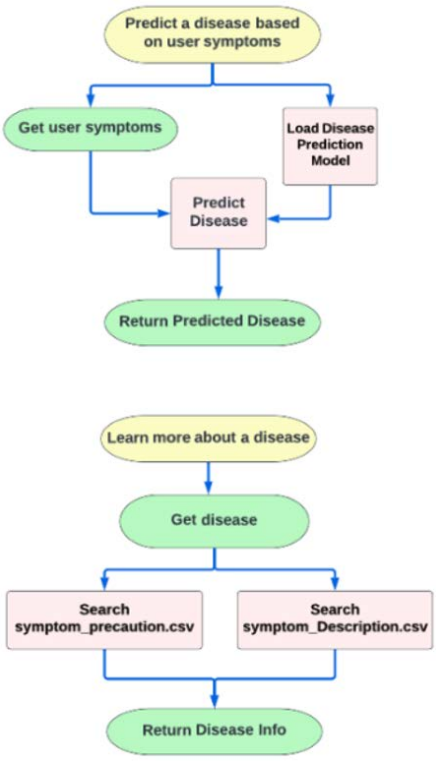

Figure 2. Flowchart for DiagnoBot core functions.

- DiagnoBot should be able to understand if the user wants to know more information about a disease.

3.1. Building the DiagnoBot

In our project, we use Rasa NLU and Rasa Core to help build and train a medical chatbot application named DiagnoBot. Rasa is an open-source NLP library that is commonly used for intent classification and entity extraction in chatbots. Rasa NLU is used to understand a user's intent and extract any entities from their text. To perform intent classification and entity extraction, Rasa NLU uses NLP methods, such as the methods shown in Figure 1, to train a specified machine learning model. After a user's intent and entities are identified, Rasa Core takes the information and tries to build a probability model that will decide which actions the chatbot will perform Terry-Jack, 2019.

DiagnoBot has two core functions as shown in Figure 2: predict a disease based on the user's symptoms and search for information about a disease. We create custom actions in our actions.py file to perform each action when needed.

To predict a disease based on a user's symptoms, DiagnoBot will load a pre-trained disease prediction

Disease	Precaution_1	Precaution_2	Precaution_3	Precaution_4
Drug Reaction	Stop irritation	Consult nearest hospital	Stop taking drug	Follow up
Malaria	Consult nearest hospital	Avoid oily food	Avoid non-veg food	Keep mosquitos out
Allergy	Apply calamine	Cover area with bandage		Use ice to compress itching

Table 1. A snippet of Symptom Description data set.

Disease	Description
Drug Reaction	An adverse drug reaction (ADR) is an injury caused by taking medication. ADRs may occur following a single dose or prolonged administration of a drug or result from the combination of two or more drugs.
Malaria	An infectious disease caused by protozoan parasites from the Plasmodium family that can be transmitted by the bite of the Anopheles mosquito or by a contaminated needle or transfusion. Falciparum malaria is the deadliest type.
Allergy	An allergy is an immune system response to a foreign substance that's not typically harmful to your body. They can include certain foods, pollen, or pet dander. Your immune system's job is to keep you healthy by fighting harmful pathogens.

Table 2. A snippet of Symptom Precaution data set.

machine learning model into our actions.py and pass the user's symptoms as a numerical vector to the model. DiagnoBot then returns the predicted disease class to the user.

To return information about a disease to the user, we use the symptom_Description.csv data set shown in Table 1 and the symptom_precaution.csv data set shown in Table 2. Each data set holds information about diseases from the Disease-Symptom data set.

3.2. Creating the Disease-Prediction Model

The steps proposed in Figure 3 allow our medical chatbot to predict diseases based on a user's symptoms. We will use the Disease-Symptoms data set from Kaggle, which has close to 132 types of symptoms (columns) and close to 42 types of diseases which gives more flexibility in finding a higher accuracy for a machine-learning model. The proposed method is broken down into two general steps: preprocess data set and apply machine learning algorithms. The model that performs the best out of the three will be used in our DiagnoBot.

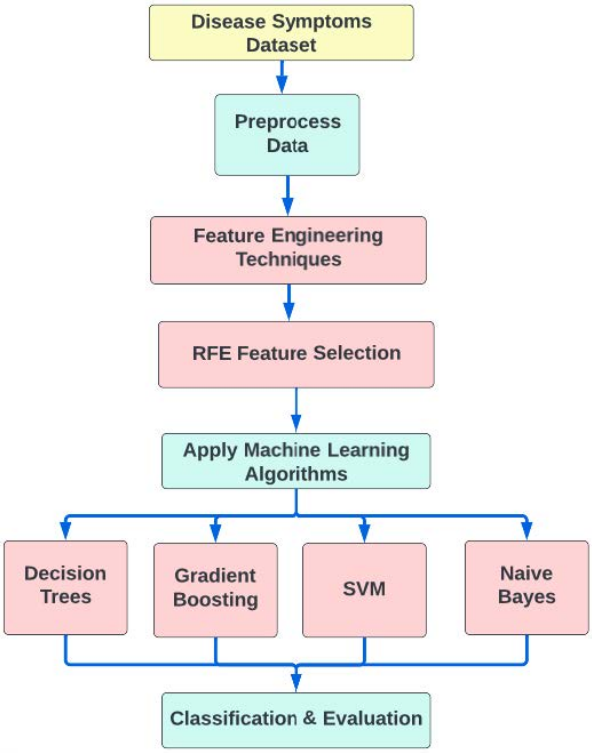

Figure 3. Flow Diagram for Disease-Prediction Model used by DiagnoBot.

The preprocessing step is divided into two parts, Feature Engineering and Feature Selection. In Feature Engineering, we increase the quality of the data by removing null values, dropping columns (attributes) with no data, and converting categorical data into numerical data using the label encoding method.

In Feature Selection, we try to identify the correlation between the features and the target variable and select only the important features that fit into a model so that it can be trained well and results with good accuracy. With fewer features to process, our machine-learning models should run more efficiently by reducing computation times. We will use Recursive Feature Elimination (RFE), which is a feature selection algorithm that can be used for both classification and regression problems. Recursive Feature Elimination recursively removes features from a data set until a desired number of features is reached. The desired number of features is selected "by fitting the given machine learning algorithm used in the core of the model, ranking features by importance, discarding the least important features, and re-fitting the model." Brownlee, 2020. Random Forests can be used to evaluate the importance of features and return their scores. Therefore, Random Forests will be used as the

core estimator for our Recursive Feature Elimination algorithm.

In our method, machine learning algorithms are applied to handle multi-class classification problems, such as Decision Trees, Naïve Bayes, Gradient Boosting, and Support Vector Machines.

Decision Trees make predictions by splitting data based on a selected feature. They are easy to interpret and can handle complex relationships between variables. In Gradient Boosting, weak machine learning models are combined in an ensemble to create a final model that can have a higher predictive power. The ensembles are usually built with decision trees. However, "the number of trees in gradient boosting decision trees is very critical in terms of over-fitting. Adding too many trees will cause over-fitting so it is important to stop adding trees at some point." Yildirim, 2020. Support Vector Machine (SVM) is an algorithm that can be used with small and complex data sets. It works to find hyper-planes that best separate data into different classes. SVM is also effective in high dimensional spaces where there are many features such as the 132 features in our Disease-Symptoms data set.

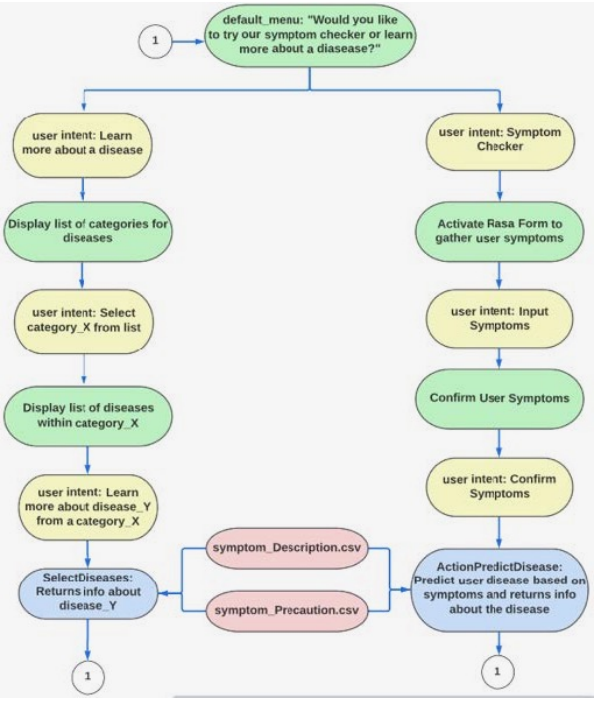

Figure 4. Conversational flowchart for DiagnoBot

4. Implementation

4.1. DiagnoBot

As displayed in Figure 4, a user can perform two main actions in DiagnoBot: receive a predicted diagnosis based on their symptoms and receive information about a disease. DiagnoBot uses two custom actions, SelectDiseases and ActionPredictDisease, to respond to user requests.

If the user's intent is to receive a predicted disease based on their symptoms using the Symptom Checker, a Rasa Form is triggered to ask users a series of questions that collect information about the symptoms experienced by users. Rasa Forms are used to define form fields to fill out slots, validate user inputs, and handle a user's response in a conversational manner. After a user inserts their symptoms, DiagnoBot stores them within a symptoms slot to be used within the ActionPredictDisease custom action.

The Rasa custom action, ActionPredictDisease, takes the user's symptoms and returns a prediction of their ailment. This is accomplished by retrieving the symptoms that have been slotted for storage and then checking the number of entries to see if the user has input enough symptoms to make a prediction. If the user has entered the requisite number of symptoms, the trained Support Vector Machine model is called to predict the user's disease. The predicted disease result is given to the user and information about the disease is obtained by looping through the symptoms_Description.csv and symptoms_precautions.csv files which contain descriptions of the disease and precautions the afflicted may need to consider as shown in Figure 5. The action ends with the user being given the option to enter a new set of symptoms, learn about another disease, or enter a custom message.

If the user's intent is to learn more about a disease, DiagnoBot returns a list of disease categories a user can choose from such as infectious diseases, cardiovascular diseases, and so forth. The Rasa custom action, SelectDiseases, is triggered when a user opts to select a disease from within a category. The SelectDiseases custom action will return information about the selected disease from the symptom_Description.csv and symptom_precaution.csv files as shown in Figure 6. The user is then prompted to select whether they wish to try the symptom checker, search for a disease, or create a custom message.

4.2. Disease Prediction Models

DiagnoBot can predict a user's disease based on their symptoms by using the Disease Prediction model displayed in Figure 7. The Disease Prediction model gathers the Disease Symptoms data set and performs

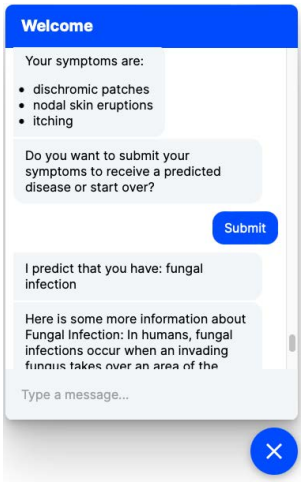

Figure 5. A user receiving a disease prediction based on their symptoms provided to DiagnoBot.

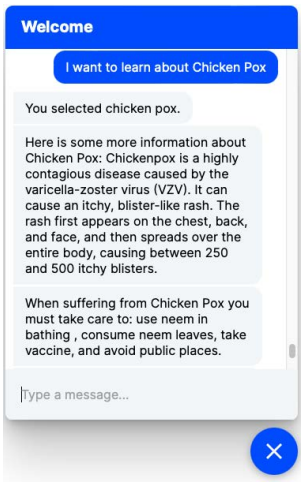

Figure 6. A user receiving information about a requested disease from DiagnoBot.

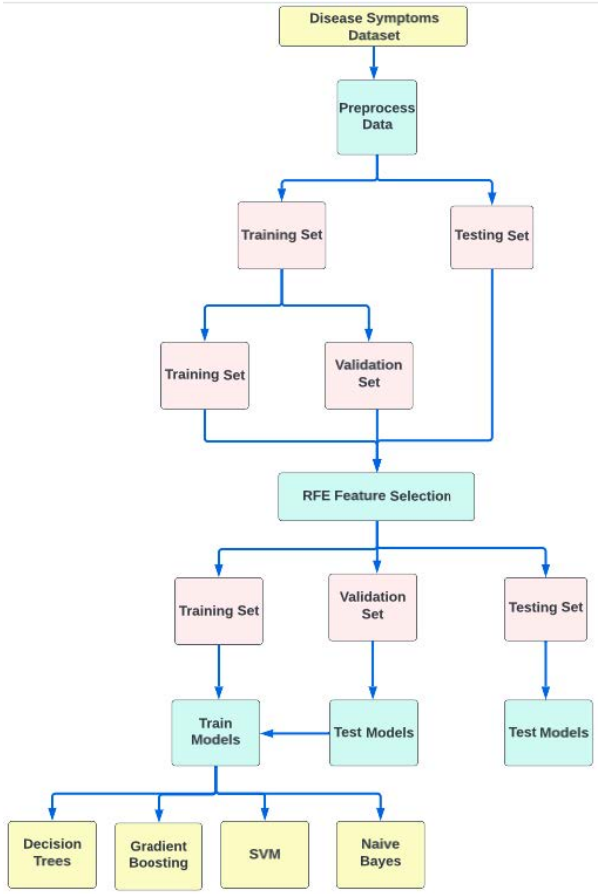

Figure 7. Flow Diagram for Disease-Prediction Model used by DiagnoBot.

multiple machine learning steps to train four models: Decision Trees, Gradient Boosting, Support Vector Machines, and Naïve Bayes.

Data preprocessing is an essential step in machine learning and involves cleaning and transforming raw data into a format suitable for modeling.

In the first step of data preprocessing, we cleaned our data which involves eliminating unnecessary or redundant information, fixing errors or inconsistencies, and resolving missing or null values. After unnecessary information is eliminated, missing or null values are handled. Null values can be handled by imputing them with a specific value, such as the mean or median of the column. Next, categorical data is converted to numerical data since certain machine learning models cannot handle categorical data. This is done by using the label encoding technique. Outliers are extreme values that can affect the accuracy of a model. Outliers can be detected and handled by removing them from the data set, transforming them, or replacing them with a more representative value such as the median.

After cleaning our data, the Disease-Symptoms data set is split into 50-50, which means that 50% of the data is used for the training data set, and 50% of the data is used to test each model's performance. The testing data set is used to evaluate the final trained models on data that has not been seen during training. The training data set is further split 70-30, which means 70% of the training data is used for training each model, and 30% of the training data is used as a validation data set. The validation data set is used to configure hyperparameters for each model and compare models during training. The use of a validation data set allowed our testing data set to remain unseen by our models during training. As previously mentioned, Recursive Feature Elimination (RFE) is a feature selection method in Python that helps to select the best features from a given data set. By iteratively eliminating less important features, RFE can help to improve model performance by reducing over-fitting and increasing generalization. In our implementation, we configured an RFE model with a Random Forest classifier as an estimator due to its ability to select the most important features. Out of 132 features, we have chosen the top 95 features based on their importance in the Random Forest model.

After we trained our RFE model using our training data set, our training, validation, and testing data sets were transformed to only show the selected feature columns. The transformed training data set is then used to train the Decision Tree, Gradient Boosting, Support Vector Machine, and Naïve Bayes models. Lastly, our trained models are evaluated on the transformed validation data set to check for possible hyperparameter configurations that avoid over-fitting and noise interference.

5. Results

Upon the collection of training data, a chatbot was developed based on Rasa. Once the chatbot application development process was completed, various tests were conducted using a range of user inputs. The output of the experiments is displayed in Table 3 and Table 4.

5.1. Performance of Algorithms on Testing Data

Table 3 shows the accuracy score of our trained models on unseen testing data. All the machine learning models show high accuracy scores when using the Recursive Feature Elimination algorithm. Of the four models that we have chosen, the Support Vector Machine shows the highest accuracy score of 100%. The Decision Tree model returned the lowest accuracy score of 97.93%.

Algorithms Used	Accuracy score
Decision Tree	97.93%
Gradient Boosting	99.63%
Naïve Bayes	99.84%
Support Vector Machine	100%

Table 3. Accuracy scores of trained models using testing data.

Algorithm Used	Predicted	Actual
Decision Tree	Dengue	Fungal infection
Gradient Boosting	Impetigo	Fungal infection
Naïve Bayes	Fungal infection	Fungal infection
Support Vector Machine	Fungal infection	Fungal infection

Table 4. Trained model predictions for unseen Fungal infection sample.

We evaluated our trained models on a previously unseen sample that contained a more diverse set of features for the Fungal infection class, namely dyschromic patches, skin rash, nodal skin eruptions, and itching. Our new sample for the Fungal infection class contains the following set of features: itching, skin rash, red spots over the body, family history, and red sore around the nose.

As shown in Table 4, the Decision Tree and Gradient Boosting models did not successfully predict the new sample as a Fungal infection despite their high performance on testing data. This suggests potential over-fitting of the training data even though RFE feature selection was implemented. In contrast, the Support Vector Machine and Naïve Bayes models successfully predicted the new sample as a Fungal infection. The performance of the Support Vector Machine and Naïve Bayes models on unseen data shows that both models can identify underlying patterns in our data set.

6. Future Work and Discussion

DiagnoBot was designed to search for information about a disease and predict a disease based on the user's symptoms using Rasa NLP and machine learning algorithms. Rasa is an open-source NLP library that is commonly used for intent classification and entity extraction in chatbots based on the trained machine learning model. Rasa NLU allows DiagnoBot to classify intents and extract entities from a user's response. Rasa Core uses the information gathered from Rasa NLU to predict what action DiagnoBot should take next. Rasa Forms are used in our chatbot to gather symptoms from a user and predict their disease.

We have trained and evaluated the following four

models to predict diseases based on a set of symptoms: Decision Trees, Gradient Boosting, Naïve Bayes, and Support Vector Machine. Our Support Vector Machine model with RFE feature selection returned a high accuracy score of 100% on testing data, which is better than the existing system that uses Backward Feature Selection. The Decision Tree returned the lowest accuracy score of 97.93%. We observed that the Decision Tree and Gradient Boosting models were prone to overfitting our training data, as they were unable to accurately predict a sample containing previously unseen features associated with the Fungal infection class. Due to its high performance, the Support Vector Machine model is used in DiagnoBot to predict a user's symptoms and provide accurate predictions based on the symptoms provided.

Several improvements to the chatbot platform could be expected in future development. First, NLU improvements could be used to create a more intuitive user experience. This could be accomplished by expanding the nlu.yml, rules.yml, and stories.yml files to contain more conversation options. Expanding the range of diseases available for diagnosis could increase the chatbot's pertinence and usefulness. This would especially hold true if recently relevant maladies such as COVID-19 were included. This task would be complex and would require the gathering of accurate symptom/disease information to be added to the symptom_description.csv and symptom_precaution.csv files, as well as updating actions.py, nlu.yml, stories.yml, and the machine learning test/train data and models. Giving the user the ability to challenge their result would be another favorable component to develop in the future. The user could voice their disagreement with the result and the chatbot could provide an alternative diagnosis by outputting the second-highest-ranked result. These potential improvements could increase the accuracy and relevance of our chatbot allowing it to become a valuable tool for providing reliable medical advice to people around the world.

To enhance the accuracy of disease prediction models, it may be beneficial to increase the size and diversity of the training data, which can be achieved through the utilization of advanced machine learning techniques such as deep learning and transfer learning. Additionally, the incorporation of other types of data, such as genetic, lifestyle, or environmental factors, may also contribute to training the model for improved accuracy. Furthermore, to address the constantly changing disease patterns, symptoms, and treatment options in real-time situations, it is recommended to train the model with up-to-date data collected from various sources to continually enhance the chatbot's accuracy.

References

About ada: Personal health companion [Retrieved May 10, 2023]. (2023). https://ada.com/about/

Brownlee, J. (2020, May). *Recursive feature elimination (rfe) for feature selection in python* [Retrieved April 19, 2023]. Machine Learning Mastery. https : / / machinelearningmastery . com / rfe - feature-selection-in-python/

Gertz, A. H., Pollack, C. C., Schultheiss, M. D., & Brownstein, J. S. (2022). Delayed medical care and underlying health in the united states during the COVID-19 pandemic: A cross-sectional study. *Preventive Medicine Reports*, *28*.

Golembiewski, E. H., Gravholt, D. L., Torres Roldan, V. D., Lincango Naranjo, E. P., Vallejo, S., Bautista, A. G., & ... Boehmer, K. R. (2022). Rural patient experiences of accessing care for chronic conditions: A systematic review and thematic synthesis of qualitative studies. *Annals of Family Medicine*, *20*(3), 266–272.

Gupta, A., & Gupta, M. (2022). Prediction of diseases using different machine learning approaches. *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)*, 712–717.

Jothi, J. N., Poongodi, S., Chinnammal, V., Kannagi, L., Panneerselvam, M., & Prabu, R. T. (2022). Ai-based humanoid chatbot for medical application. *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)*, 1135–1140.

Kumar, K. L., & Reddy, B. E. (2021). Heart disease detection system using gradient boosting technique. *2021 International Conference on Computing Sciences (ICCS)*, 228–233.

Luba, P. (2020, May). *Healthcare chatbots can help with the pandemic* [Retrieved May 10, 2023]. Towards Data Science. https : / / towardsdatascience.com/healthcare-chatbots-can-help-with-the-pandemic-bcc07fc606c9

Mathew, R. B., Varghese, S., Joy, S. E., & Alex, S. S. (2019). Chatbot for disease prediction and treatment recommendation using machine learning. *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, 851–856.

Mridha, K. (2021). Early prediction of breast cancer by using artificial neural network and

machine learning techniques. *2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)*, 582–587.

Peya, Z. J., Chumki, M., & Shymon Islam, M. (2022). Thyroid disease prediction based on feature selection and machine learning. *2022 25th International Conference on Computer and Information Technology (ICCIT)*, 495–500.

Prayitno, P. I., Pujo Leksono, R. P., Chai, F., Aldy, R., & Budiharto, W. (2021). Health chatbot using natural language processing for disease prediction and treatment. *2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI)*, 62–67.

Sisodia, D. S., & Agrawal, R. (2020). Data imputation-based learning models for prediction of diabetes. *2020 International Conference on Decision Aid Sciences and Application (DASA)*, 966–970.

Srivastava, P., & Singh, N. (2020). Automatized medical chatbot (medibot). *2020 International Conference on Power Electronics IoT Applications in Renewable Energy and its Control (PARC)*, 351–354.

Swarupa, A., Sree, V., Nookambika, S., Kishore, Y., & Teja, U. (2021). Disease prediction: Smart disease prediction system using random forest algorithm. *2021 IEEE International Conference on Intelligent Systems, Smart and Green Technologies (ICISSGT)*, 48–51.

Terry-Jack, M. (2019, September). *Let's learn rasa* [Retrieved April 13, 2023]. Medium. https://medium.com/@b.terryjack/lets-learn-rasa-c8a3744fa781

Tyagi, A., Mehra, R., & Saxena, A. (2018). Interactive thyroid disease prediction system using machine learning technique. *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 689–693.

Yildirim, S. (2020, February). *Gradient boosted decision trees-explained* [Retrieved April 19, 2023]. Towards Data Science. https://towardsdatascience.com/gradient-boosted-decision-trees-explained-9259bd8205af