

Implementation of intelligent system to support remote telemedicine services using
chatbots technology

Maria Vasiliou

AM 1806

A Thesis in the Field of Natural Language Processing in Telemedicine
for the Degree of Master of Big Data and Analytics

University of Piraeus

02 2020

Supervisory Committee

Dr. Ilias Maglogiannis

Abstract

The objective of the current thesis is the implementation, integration and training of the “MV Health Bot” by leveraging Natural Language Processing (NLP) and Machine Learning (ML) algorithms. MV Healthbot’s role is to integrate features of a virtual assistant and bridge the gap between patients and health professionals. The chatbot provides a friendly interface to the patient in order to collect the demographic data and the health symptoms. The data are normalized and passed to the well trained artificial intelligence models for health predictions. The outcome of the prediction is shared with the patient and is stored for further analyses by the healthcare professionals.

MV HealthBot is using human–computer interaction technologies based on natural language conversations or trivial selection flows. It provides predictive health services by using AI models and NLP for identifying Covid-19 , diagnose diseases based on symptoms and alert the patient in case of disorders. MvHealthBot also simulates the process of fetching data from a public health API using the patient's medical ID and provides diagnosis via the trained AI model that has been built for the purposes of the app.

Keywords: Chatbot, AI, machine learning, digital health, telemedicine, NLP

Frontispiece



Author's Biographical Sketch

Pioneering technologist with 15+ years of executive level experience. Deep knowledge on designing and implementing heavy transactional platforms mainly focused on the Mobile Advertising area. Strong global experience in delivering strategic technology solutions for Tier 1 Telecoms and Media Agencies. Entrepreneurial-spirited, with a powerful blend of technology vision and business acumen. Strong experience in software development, architecture/infrastructure design, operation management, full project life cycle management and process plan implementation.

Acknowledgments

I would like to express my gratitude to my supervisor, Ilias Magklogiannis for the continued support and guidance throughout the thesis period.

Table of Contents

Abstract	4
Frontispiece	5
Author’s Biographical Sketch	6
Acknowledgments	7
List of Figures	12
Chapter I.	
Introduction	14
Motivation	15
Contributions	16
Thesis Outline	17
Chapter II.	
Background and Related Work	18
Historical Background	18
Related Work - Literature Survey	22
Conversational Agents Role in Telemedicine & Healthcare Support For Home-Living Elderly Individuals	22
A Self-Diagnosis Medical Chatbot Using Artificial Intelligence	25
Sanative Chatbot for health seekers	27
Chatbots meet eHealth: automatizing healthcare	28
Chatbot for healthcare system using Artificial Intelligence	31
Using Health Chatbots for Behavior Change: A Mapping Study	32
Automated medical bot	34
Comparison	35
Technologies	37
Machine Learning	38
Types of Machine learning	39

Supervised machine learning	40
Unsupervised learning	46
NLP	49
NLP Techniques and Semantics	49
Algorithms for NLP	51
LSTM - Long short term memory	51
Sequence 2 Sequence	52
Named Entity recognition model	53
Word Embedding	53
Speech Recognition	54
Connectionist temporal classification	56
Neural machine translation and Google NMT	56
Conversational Interfaces, ChatBots	57
Contribution	63
Chapter III. Design and Implementation	63
Proposed Architecture	63
Implementation Details	66
MV Health Bot ML Model	67
Covid-19 ML Model	67
Model Algorithm	67
Framework, storage	68
Heart Disease ML Model	68
Model Algorithm	69
Framework, Storage	69
APIs	70
Covid-19 ML API	71
Heart Disease ML API	72
Hospital API	74

Notification API	75
DialogflowWebhook Api	77
MV HealthBot NLP	77
Flow Diagram	77
Intent Classifier	78
Demographic Intent	79
Patient History Intent	81
Symptoms	83
HealthCondition	84
BookDoctor Intent	85
MV Health bot Notification Alerts	87
Cloud Scheduler	87
Cloud Notification	88
Chapter IV. Experimentation and results	89
Heart Disease ML Model	89
Dataset Pre-process & Analysis	90
Model Creation and Comparison	100
Models comparison	100
Covid-19 ML Model	101
Dataset Pre-process & Analysis	101
Model Creation, Evaluation and Prediction	103
Training Process	104
Evaluation	107
Prediction	108
MV Health Bot ChatBot	109
Demographic Intent Dialog	109
Patient History Intent	114
Symptoms	115

Health Condition	116
Book Doctor's appointment	118
Covid-19 Application	118
Authenticate	119
Menu Selection	120
Covid-19 CheckUp	121
Conclusion	124
[Bibliography/References/Works Cited.]	125

List of Figures

Figure 1a. Turing Test	20
Figure 1b. Patient-Chatbot Activity diagram	24
Figure 1c: Finite state graph	25
Figure 1d. HOLMeS System	29
Figure 2. Machine learning techniques	40
Figure 3. Supervised learning algorithm	40
Figure 4. Regression problem	42
Figure 5. Classification problem	43
Figure 6. The result of a cluster analysis	47
Figure 7. 2D PCA plot from 30 feature dataset	48
Figure 8a. CBOW vs SKIP-GLAM	55
Figure 8b. Speech Recognition process flow	55
Figure 9. Conversational flow using AI chatbot	58
Figure 10. Intent Interaction	62
Figure 11. Fulfillment	63
Figure 12. MV Health Bot Architecture	64
Figure 13. First five observations of the dataset	93
Figure 14. Dataset's statistical data	93
Figure 15. Density diagram	94

Figure 16. Histogram	95
Figure 16a. Age per Target class	96
Figure 16b. Distribution of sex observations	96
Figure 16c. Age versus sex per target class	97
Figure 16d. Scatter plot for disease status per age	98
Figure 17. Correlation matrix	99
Figure 18. Logistic Regression Model API metadata	70
Figure 19. Message Queue Pub/Sub architecture	76
Figure 20. Health Bot Flow diagram	78
Figure 21. Covid-19 dataset	103

Chapter I.

Introduction

The objective of the thesis is to analyze and elaborate on the implementation of an intelligent system that can support telemedicine services using chatbot technology.

Living in the era of the Internet of Things (IOT) and Big Data, telemedicine systems produce valuable health related data that needs to be consumed and analyzed via intelligent platforms using AI and ML technology. In addition, there is a need for the end user to be able to communicate with a sophisticated system, ask questions and get insights from well trained models using NLP technology. Natural Language Processing is the technology used to aid computers to understand human's natural language. MV HealthBot exposes the API interfaces to collect health data from telemedicine systems, exposes a friendly human-like chatbot interface for recording health symptoms using NLP technology and uses regression algorithms to predict health disorders.

What is a ChatBot? A ChatBot is an artificial intelligence (AI) software that can simulate a conversation with a user in the natural language. The interface can be a native application, web pages, PWAs, APIs. The chatbot is trying to mimic the human to human conversations and interaction. The chatbot evolution offered a breakthrough to the legacy questionaire systems by making the interviewing process more user friendly.

MV HealthBot is trained to understand the content of a keyword-based conversation (Intents) and to match the conversation to symptoms. The platform is also

collecting daily health data via APIs. Regression model has been trained to identify if the ingested data are within the accepted ranges otherwise to trigger notifications in case of disorder identification. The ChatBot analyzes the user's request to locate the Intents and extract the Entities. The ability to locate the Intents and extract the Entities of the user's request is the first and most important prerequisite in the ChatBot's kernel.

Motivation

The concept of conversing with a computer machine has been around for a long time. Joseph Weizenbaum developed the first NLP program called ELIZA in the early sixties at MIT. By using 'pattern matching' and subrogation methods , ELIZA could simulate part of the conversation without the intelligence of a contextual framework. MAD-Slip was the programming language that has been used, which enabled ELIZA to analyze user inputs and participate in a talk according to the script rules. The script that was most acknowledged was called “DOCTOR” and was mimicking Carl Roger parroting practises. ELIZA was one of the first chatbot able to test the “Turing” Test.

Nowadays, NLP and AI have advanced to such a degree that user's intentions can be actively predicted. A variety of commercial frameworks have been developed to provide services to define behaviours from chatBots, including IBM's Watson, Google's Dialogflow, Amazon's Alexa.

MV HealthBot NLP implementation is based on DialogFlow.

Contributions

- How does the user interact with the chatbot?

The chatbot is accessible via Native application (Android, iOS) and progressive web app (PWA). The HealthBot NLP engine provides a seamless Medical Interview experience to the end user. The objective is to collect patient's profile data, information about historical diseases, symptoms that the patient is going through and pass this information to the doctor for diagnosis.

- How can I close an appointment with the doctor in case of disorder

In case of disorder the application prompts the user to close an appointment with the appropriate doctor and arranges with the user the data and time of the rendezvous

- How the patient's data are evaluated

There are two logistic regression models that have been trained and tested for the needs of this application. The Covid-19 model, which based on the patient's symptoms provides the risk analysis assessment if the user is positive in Covid-19 and what measures should be taken. The second model is the Heart Disease model, which takes as input a list of attributes related to the blood pressure, glucose, etc. and provides the risk analysis assessment on the probability of heart disease. These data can be fetched from hospitals and clinics via APIs using the patient's medicalId. For the purpose of the application, simulation of such API has been developed (Hospital API). The ML models are constantly re-trained using the input conversational or API data.

- How the chatbot understands the free text input data

The application is using the natural language processing machine learning model that is constantly trained on the input data. The main objective is the input data to be classified to the correct intents according to the keywords that exist in the respective Entities. Therefore, the more training phrases are provided to the NLP model , the better the outcome will be.

Thesis Outline

This chapter described the problem that Mv Healthbot came to solve and the contributions. The rest of this thesis is organized as follows.

Chapter II elaborates on the key technological areas that are related to this thesis and presents relevant literature and work.

Chapter III describes in depth the architecture and implementation of the MV Health Bot. It presents precisely the flow using Flow diagrams and screenshots. It elaborates on the process and the integration points through the architecture and implementation section.

Chapter IV provides insights on the execution of the algorithms and the level of precision in real cases. It provides the models evaluations. It analyzes the sequence flows with real cases and provides screenshots of the application in action.

Chapter II.

Background and Related Work

There is a need for technical solutions to programmatically interact with the patient and the hospitals in order to gather information, monitor health conditions and provide support. Smartphones and various types of digital health devices have been used to telemedicine in order to provide personalization and suggestions, although the real disruption came from the chatbots. Conversational agents or chatbots are playing a catalytic role in the interaction between the patients and the clinicians. This chapter will dive deep into the conversational agent technology by using machine learning, natural language processing and artificial intelligence. Additionally, this chapter will elaborate on similar published papers and literature surveys.

Historical Background

The history of natural language processing officially began by Alan Turing in the 1950s. He issued the paper called “Computing Machinery and Intelligence”^[1] analyzing the Imitation Game based on the words “machine” and “think”, “Can machine think”?

Turing instead of trying to find out if a machine can think, he moved to the approach if the machine can bit a game called the “Imitation Game”^[1]. Three players composed the team, the man (player A), the woman (player B), the interrogator (player C) with the objective to identify the gender of A and B players. Player B is the sincere

opponent, trying to help the interrogator in making the right decision, although Player A is trying for the opposite.

Turing changed the opponents of the game into one computer and two humans. One of the humans is the judge. Both computer and human are talking to the judge via a terminal. If the judge cannot identify correctly if the input is from the computer or the human, then the computer wins. This approach has focused on the performance capabilities of the system that makes thinking possible and how an ordinary system can create them. Turing was trying to find how the computation could lead to generate human cognitive capability[7].

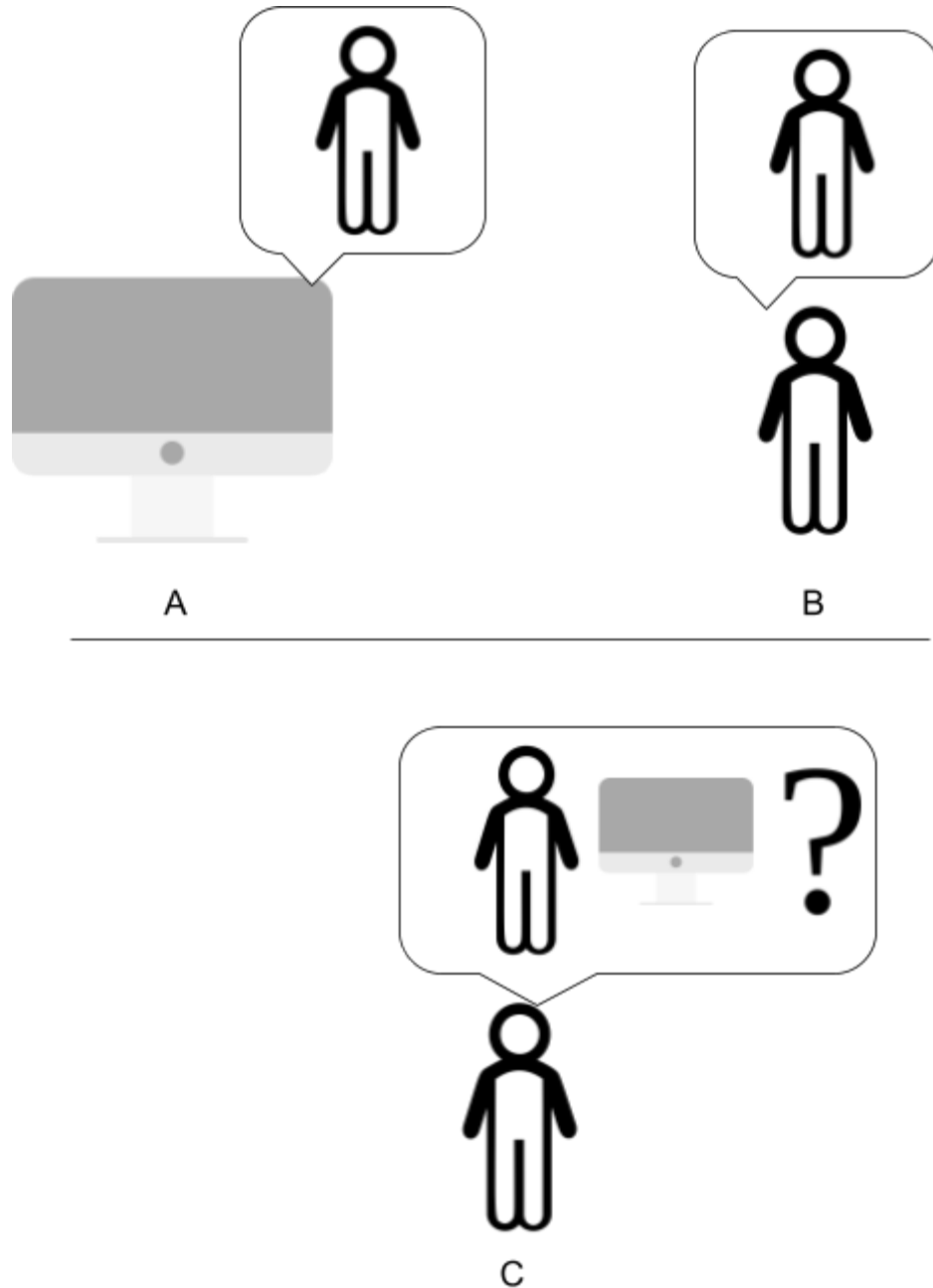


Figure 1a. Turing Test

The Georgetown - IBM experiment ^[2] in 1954 was showing the possibilities of machine translation by automatically translating more than sixty Russian sentences into

English. It consists of two hundred fifty lexical items and six grammar rules. The idea was mostly “lexicographical” and the concept is that specific rules and steps^[3] where determine the connection based on a defined dictionary. Nevertheless, after ten years of research the results had failed to live up to expectations as the process has proved to be extremely slow.

In the 1960s, a new NLP system was developed by Terry Winograd at MIT called SHRDLU^[6] which operated in limited "block worlds" with confined vocabulary.

SHRDLU’s core functionality was based on the concept that the possible combinations of the entire set of objects could be achieved by combining around 50 words. Every object had a unique name which during the SHRDLU dialog^[8] parsing process was memorized by the system in conjunction with the processed past events for contextual identification.

The period between 1980s and mid 1990s was called “statistical revolution”, as natural language processing has moved away from the rule based coding and stemming practices and focused primarily on machine learning algorithms. The hardware acceleration as well as the scientists recognition of the validity and the power of the ML algorithms contributed to the NLP evolution. Hard coded rules started to be replaced by decision trees ML algorithms. The scientists have started to focus on statistical models that produce probabilistic decisions by attaching weights to the features and reevaluating the input data. After 2000, Natural Language processing has been drastically enriched by new machine learning algorithms including but not limited to Neural networks. Neural networks consist of interconnected nodes and edges on multiple layers. Each edge has a relative weight and the network defines the computational rules that should be used to

transmit input data from the network input to the output level. Classification, regression algorithms are applied at the output layers for better results. Learning techniques like word embeddings are used for capturing semantic properties of words. Deep Neural networks brought an evolution to NLP systems design.

Related Work - Literature Survey

The area of chatbots and conversational agents is booming as there is a real need for artificial intelligence and NLP to undertake many manual executed procedures under different sectors, education, bookings services, first level customer support, healthcare etc..

Conversational Agents Role in Telemedicine & Healthcare

Support For Home-Living Elderly Individuals

Ahmed Fadhil[30] is analyzing the role of telemedicine and healthcare support for home-living elderly individuals by using chatbots. The aim of the specific chatbot is to act as healthcare assistance and provide responses to patients' questions about their health issues, recommendations related to diet, medication adherence. The chatbot is watching the patient's health condition and notifies the doctor in case of disorders. Acknowledging that for the elderly people it is more difficult to adapt to the digital world, the design and the implementation of a friendly to use system is a big challenge . The current system is using NLP algorithms for parsing the input data and analyzing the content for it's meaning and the contextual relation with the satellite words. The output is matched with

existing symptoms category clusters that have been setup in order to provide a consolidated response. The response text is either predefined in the chatbot dialog structure or it is retrieved from REST API calls after applying further functional processing. The objective of the chatbot is to provide the right health diagnosis after analyzing and matching the symptoms.

According to Figure 2a, which demonstrates the activity diagram, the user enters the chatbot application via the “Access” activity. Next state is the “Presentation” activity where the chatbot assists with useful insights about the application and the out of the box functionality. During the “Process” state the chatbot serves the relevant information to the patient based on their request. The next step is the “Options” state where a list of options are provided to the user based on their health condition in order to select and proceed further. The flow is trivial according to the user's selection The bot provides suggestions and the user can declare on his satisfaction level. The state is switching to “Satisfying” if everything went in order with the specific activity therefore the state changes to “Conclude”. During the ”Feedback” state, the patient has the possibility to retrieve the respective outcome / feedback of the . In case of “Unsatisfying” state different recommendations will be provided to the user by the chatbot. If the bot cannot identify atall the patient’s request the process is handed to an “external actor” state which will invoke a human resource to the process.

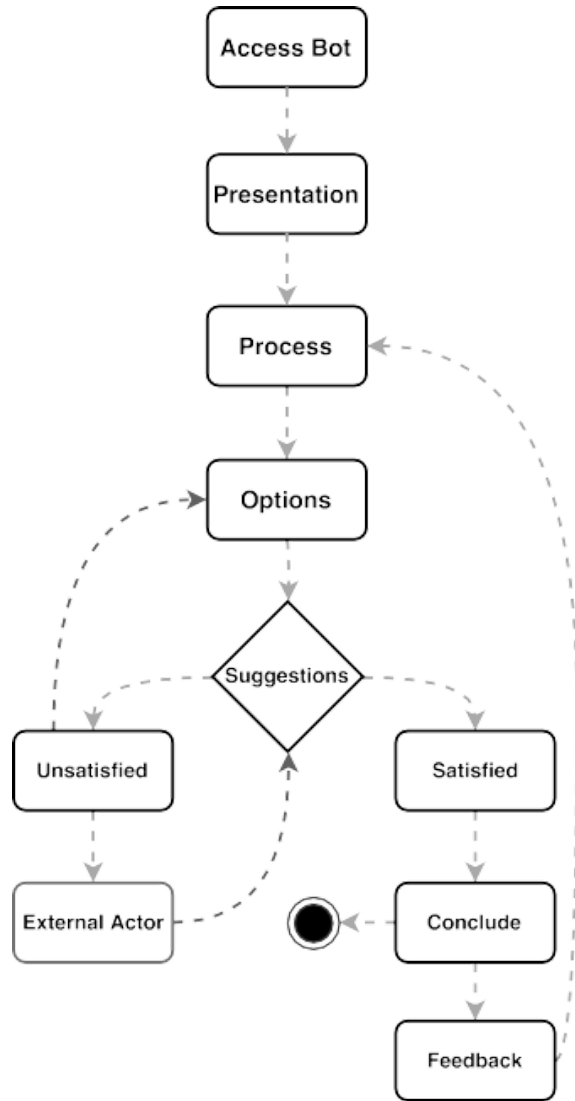


Figure 1b. Activity diagram

The drawback of this study is that the solution has been well designed and analysed but not implemented.

A Self-Diagnosis Medical Chatbot Using Artificial Intelligence

Another interesting work has been conducted by Divya S, et al. [31] related to personalized diagnoses based on symptoms. The AI bot is constantly trying to identify and match the symptoms in order to successfully derive to the diagnosis. The symptoms are classified, characterized as major or minor and if major are passed to the doctor for further processing. “String Searching Algorithm” has been used for the symptoms identification. The identified symptoms are passed back to the user for confirmation. User’s reply will derive the system to shortlist specific diseases. Additional questions are asked for further symptom classification in order to narrow down the estimated disease list.

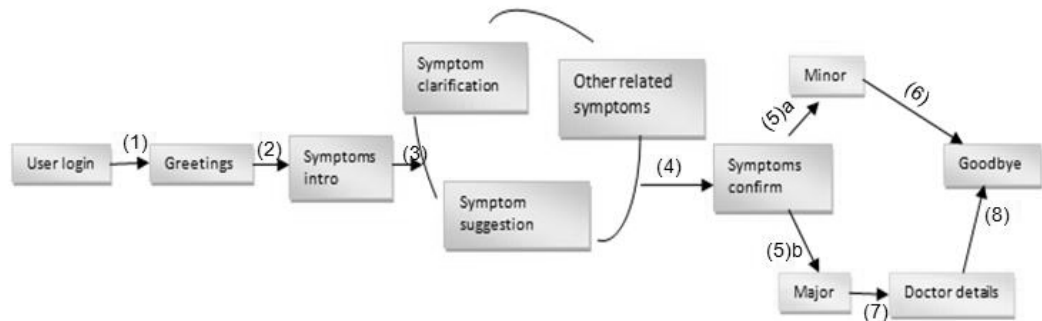


Fig1c: Finite state graph

The bot authorizes the user first and then it asks questions related to the symptoms till it has sufficient information to derive to the correct prognosis . Afterwards, the user re-enters the process to provide symptoms and expect the accurate diagnosis. All these

records can be reviewed in the history section of the app. Text input is filtered and analyzed via the “**String Searching Algorithm**” in order to subtract the symptoms identified in the input text . This process concludes on a list of symptoms that the user is asked to verify and confirm. The next steps are the symptoms classification and the population of a list of questions per symptom that the responses will narrow down the possible disease diagnosis. A constant matching process is taking place where the symptoms that are identified in the user’s input text after the NLP processing, is compared against the disease to symptoms table in the database. The final list of diseases is provided to the end user via the chatbot interface. The result is evaluated based on the severity of the disease according to the conditional rules built in the chatbot. The chatbot’s NLP algorithm is constantly trained using the user’s generated content. If the issue is major the chatbot makes contact with a specialized doctor for this type of disease. If the issue is minor then the chatbot specifies the disease and provides first aid support.

Sanative Chatbot for health seekers

V.Manoj Kumar [38] has designed a search engine mechanism around health context, “Sanative Chatbot for health seekers” as it is called. Based on Pew’s Research Centre National survey in Jan 2013 one to three USA citizens searched on the internet to find answers for their health condition according to their symptoms. The results of the research show that there is a high demand for a technical solution that can communicate using easy plain human language and provide back verified responses to user’s health related questions . The aim of this system is the health providers to provide accurate responses to the patient’s questions without using difficult medical wording. Therefore two identification patterns are used, the “Local mining” and “Global Approached”. “Local mining” has the objective to analyze the input data and extract the health keywords which are matched to external authorized lexicon. “Global learning” aims to learn basic concepts that are missing and spreads accurate terminology between the underlying linked files in a large dataset.

In traditional systems there are various limitations including but not limited to:

- Replies according to doctor’s availability
- Lack of compatibility, which is based on the ability of the doctor to answer the specific question. Not all doctors can answer all questions.
- Payment methods, live chat or communication via telephone are chargeable services.

Auto-generated systems have been developed to overcome these issues and link health seeker and healthcare providers. Rule based and ML algorithms are used for normalizing and cleaning the unrelated row data, and do the matching between the extracted keywords and external health related lexicons using “local mining” and “global learning” techniques. The core modules of this system consist of multiple modules . The row data analysis and cleaning is vital for the feature identification and extraction. Pruning techniques are in place in order to extract noun phrases and eliminate noisy data. “Lexical similarity” technique is in place in order to map the health related words with the input data using lexical similarity by graph-based global learning. After all the steps above the most relevant medical response will be served to the user as an answer to his input query and the user will rate the response for its relevance.

Chatbots meet eHealth: automatizing healthcare

Flora Amato, et al. [39] study elaborates on the effectiveness of the e-health applications between humans and machines. Specifically, what is needed for a chatbot system to be developed and trained in order to communicate in a human being way. The research was performed using real clinical measurements, where chatbot was used by the healthcare system to help patients choose the most appropriate way to fight the disease. Nowadays, there are various sources that produce tremendous amounts of health data like personal health records (PHR), electronic medical records (EMR) systems, mobilized health records (MHR), healthcare monitors, as well as biomedical sensors and smart devices. These data are ideal to be used for providing intelligent recommendation systems using

AI and NLP. The big challenges though in using health care data are elements like data privacy, clinical restrictions, data management, extensible and scalable data enhancements and contextual elaboration. In this work a new medical recommendation system has been designed, called HOLMeS. HOLMeS stands for “Health On-Line Medical Suggestions” and aims to autonomously interact with the patient via a chat interface.

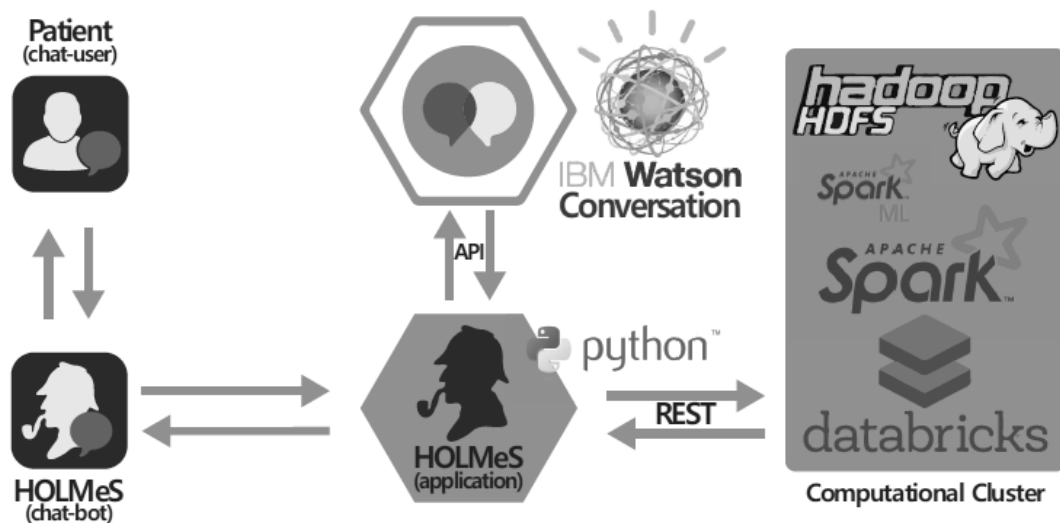


Fig1d: HOLMeS System

The HOLMeS system is composed of the following components:

- HOLMeS Application** is the core of the HOLMeS system. It holds the business logic of the platform. The programming language that has been used is Python and is consuming the Watson Conversation API for the natural language interactions with the user via the chat-bot.

- **HOLMeS ChatBot** is the agent that has been designed to make patients feel more comfortable, by interacting using the chat service. Using deep learning mechanisms, it is capable of understanding different types of communication forms, formal and more informal types. It is used as the first point of interaction with the HOLMeS System for requesting health care services. It collects personal data from the users such as age, height, weight, smoking status etc.
- **IBM Watson** including the Conversation APIs is the platform that is used for applying the context identification via natural language processing and identifies the user intents. The platform is using machine learning and data mining processes as well as NLP and deep learning algorithms.
- **Computational Cluster** holds the business logic of the system and provides the rules to apply the decision taking logic . Apache Spark on Databricks infrastructure cluster has been setup for handling big clinical data scenarios, ensuring sufficient response time and scalability. It uses Spark ML machine learning algorithms and Hadoop HDFS storage. The decision algorithm that has been used is based on Random Forest classification. Spark ML is the ecosystem for the data analysis processing, the Random Forest model training , the classification of the optimal disease prevention path, as well as the provisioning of the histogram of each dataset's disease probability. The results are stored in the database for further analysis, training and processing. The Area Under ROC Curve (AUC) of the Random Forest model has 74.65% accuracy. By injecting

more specific disease features to the dataset, HOLMeS accuracy climbed up to 86.78% achieving better prevention pathway predictions.

Chatbot for healthcare system using Artificial Intelligence

Kavitha B. R. and Dr. Chethana R. Murthy from RV College of Engineering in Bengaluru, Karnataka [40] conducted an interesting analysis regarding the Chatbots in healthcare systems using Artificial Intelligence. The main objective is to extract from the sentences the relevant keywords, apply decision logic according to the matching rules and finally respond accurately to the question. cosine similarity, n-gram and TF-IDF are used for keyword ranking and evaluation of the sentence resemblance. Score analysis for each sentence is applied for all the similar sentences within the query. The chatbot knowledge database which is used to spot the sentence and decide on the answer to the question is stored in the RDBMS database. The input sentence will get the similarity score of input sentences using bigram. The input is collected using the text () function, punctuation is removed using the trim () function and the random () function is used to choose a response from the database. n-gram technique is used for extracting the words from the sentences and applying comparison and deduction logic to the input with case data using Moro phonemes and phonemes as the deciding parameter. Probability analysis for the closest match is performed. Porter algorithm is used to discard unwanted words like suffixes or prefixes. The SVM algorithm is used for input context classification. N-gram TFIDF is used for extraction of the set of keywords and frequency of the

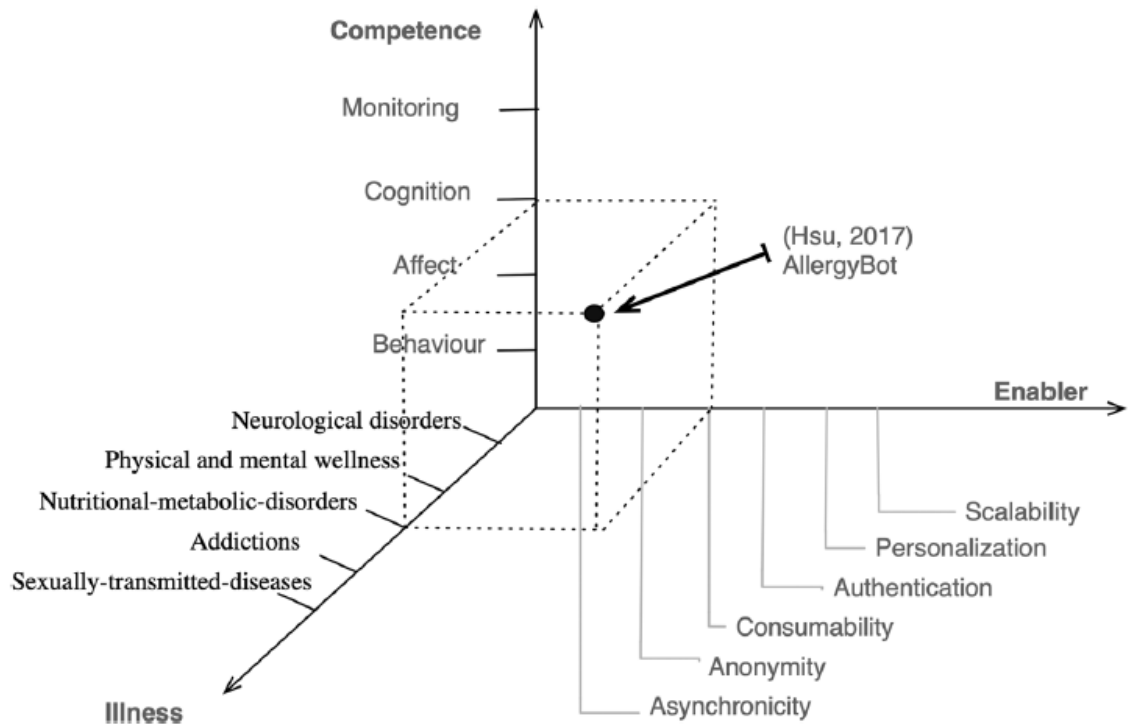
keywords from the input (TF) as well as the computation of the weight of uncommon words over all reports in the input text (IDF). Cosine similarity has been used to check the similarity between the sentences.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

After the above processes the answer for the query is retrieved and displayed to the chatbot UI interface.

Using Health Chatbots for Behavior Change: A Mapping Study

Juanan Pereira and Óscar Díaz analyzed the landscape of health chatbots by focusing on three basic questions, what kind of diseases are the chatbots encountering, what patient's skills do chatbots aim for and who are the most interested chatbot technology providers in the health sector. The paper applies a "Systematic Mapping Study" (SLR) to deal with these questions. The target audience of the chatbot is the patients. The study analysis the triplet $\langle \textit{illness}, \textit{competences}, \textit{technicalEnablers} \rangle$ where "illness" refers to the encountered disease clusters, "competences" state the patient's desired conduct; and "technicalEnablers" pertain the benefits that have been gained from the chatbot systems technology.



The methodology that has been followed in order to tackle the paper’s inquiry questions is using the SLR methodology. The SLR protocol appliance expedites the collection, identification, interpretation and finding from papers related to a specific domain. An SLR defines a structured analysis protocol with clear search methodology, certain principles for include or exclude selection, and collection of valuable information from other primary studies. For the development of the searching and identification mechanism which identifies the keys and the possible overlaps, a collection of key studies with surveys from the closest location areas has been conducted . The main criteria for the search string are related to patient’s population , chatbot usage in health sector, comparison processes among healthcare chatbots contribution, outcome of

patient's competence in using chatbots and contextual interaction of patient's with the chatbot (PICOC = "Population", "Intervention", "Comparison", "Outcomes", "Context")

The matched keywords were: 'patient', 'chatbot' and 'healthcare' as well as 'chatbots', 'conversational agents' and 'virtual agents'. The search string can be explained by the following equation :

$$((^B\text{conversational agent}^{\wedge} \text{OR}^B \text{virtual agent}^{\wedge} \text{OR}^B \text{chatbot}^{\wedge}) \text{AND } (^B\text{health}^{*\wedge})) \text{AND } (^B\text{patient}^{\wedge})$$

Classification of thirty articles following the defined triplet took place and the insights are that mental, physical wellness, nutritional and metabolic disorders are the most popular health related topics. The most searchable terms via the chatbots are the 'affect' and the 'cognition'. The widely mentioned chatbot enabler is 'consumability'.

Automated medical bot

Automated medical bot is the subject of another interesting paper from Krishnendu Rarhi, et al. [32]. The objective of this paper is the design of a medical chatbot that provides diagnosis and measures the seriousness of the diagnosis based on the symptoms. It is using AIML (Artificial Intelligence Mark-up Language) to detect human message patterns [33]. The bot analyzes the input text and breaks it down to symptoms. Each symptom has a seriousness score and if the sum of the score values reaches a specific threshold then "call the doctor" action will be triggered as well as

temporary tips and medications. Doctors and Medical Professionals ingested medical data to the platform so that the Chatbot training can be more efficient and accurate. The Chatbot engine has used JAVA for the implementation and for the AIML platform Pandorobot has been used.

Comparison

The paper “**Conversational Agents Role in Telemedicine & Healthcare**” that is elaborating on the area of telemedicine and healthcare support for home-living elderly individuals by using chatbots is only in theoretical level and **has not been implemented**. It provides guidelines on how it should be implemented but only in theoretical scope.

The paper’s “**A Self-Diagnosis Medical Chatbot Using Artificial Intelligence**” subject is the AI bot that provides personalized diagnoses based on symptoms. The symptoms are collected using the **String Searching Algorithm** and the keyword extraction using **NLP** technology. A lookup table between diseases and symptoms exist in the database and does the **rule base matching** between provided symptoms and predicted diseases.

The paper “**Sanative Chatbot for health seekers**” brings together the healthcare providers and the health seekers. The technical process that the platform is following is the **input gathering and data preprocessing, terminology detection** by dataset pruning

of irrelevant data, mapping of the medical words using **lexical similarity** by graph-based global learning.

The paper “**Chatbots meet eHealth: automatizing healthcare**” provides the most complete implementation of the medical recommendation system. The solution has a process to gather clinical data and use them as training datasets. The trained model using **Random Forest algorithms** has **86.78% Area Under ROC Curve (AUC)** achieving very good predictions on the disease’s probability and likelihood. It is using **Watson Conversation API** for natural language processing (NLP) and contextual analysis. Once the input data are analysed and classified, the platform is using the trained models to provide a **histogram** regarding the probability of each disease in the dataset.

The paper’s “**Chatbot for healthcare system using Artificial Intelligence**” main objective is to identify the keywords from the input sentences via the chatbot application, make a decision for the query and answer the question. The keyword ranking and sentence similarity calculation are found using **n-gram, TF-IDF and cosine similarity**. The input context classification is taking place using **SVM algorithm**.

The paper “**Using Health Chatbots for Behavior Change: A Mapping Study**” is analyzing the landscape of the health chatbots using **Systematic Mapping Study (SLR)**. The triplet <**illness, competences, technicalEnablers**> cover the group of meanings under which the papers out there will be classified and assigned to.

The most relevant and complete paper is the “**Chatbots meet eHealth: automatizing healthcare**” which provides an end to end solution to disease

identification based on symptoms challenge. It is using NLP, Machine Learning Algorithms with high occurrence and a scalability back end system based on Spark ML and hadoop for data analysis and classification.

The “**Automated medical bot**” paper objective is the design of a medical chatbot that provides diagnosis and measures the seriousness of the diagnosis based on the symptoms. The chatbot is in an abstract layer and needs implementation in order to be used.

MV Health Bot provides a more holistic and comprehensive approach on the telemedicine chatbot sector as it has designed and fully implemented the end to end solution. The difference with the studies above is that MV Health Bot covers all areas, from chatbot interface implementation, natural language processing and classification, health disease risk analysis using accurate ml models, integration with external sources for health data collection. More thorough analysis on the implementation is coming to the next chapter.

Technologies

Chatbots by using AI and ML algorithms will heavily contribute to the patient's interviewing process for the collection of health symptoms and can provide self-care smart recommendations using NLP and AI techniques. By using telemedicine

applications like chatbots, patients have the ability to communicate from everywhere , even from rural areas, provide easier symptoms collection through natural language processing techniques and receive improved diagnosis. In addition the healthcare professionals can have access to comprehensive patient data.

Machine Learning

Machine Learning, ML is the new trend for innovative businesses and sectors that want to leverage data assets in order to gain more knowledge on their fields. ML is a subset of AI that empowers the system to be trained using historical data instead of using code with predefined instructions and patterns. By developing and training the appropriate machine learning models, accurate predictions can be extracted.

Compared to biostatistical methods, machine learning has the advantage of scalability, flexibility and speed, key factors for risk analysis and classification processes and predictions. It can combine different types of data like clinical datasets, patient's profiling data, data from surveys and use them for predictions and risk analysis via the ML models.

Machine learning utilizes a variety of algorithms that constantly learn from data to improve, describe data and predict outcomes. The more training data are ingested, the more precise ML models are produced. These trained ML Models can then work independently and respond with their predictions on the input data that they were requested. There are online and offline machine learning models. The online ML model is constantly refined as new data are injected and processed in real time, therefore the

constant training may lead the system to adapt to changing data and associations in data.

Although offline models are trained on existing data , deployed and cannot change on the fly.

Machine learning processes consist of the following processes:

- Data Preparation which includes the cleaning of the data by removing empty values, errors and deviations, normalizing the data, format , short, merge, join if applicable
- Training the algorithm: after the data preparation, the dataset is ready to be the input on the appropriate algorithm that will be used for training in order to produce the ML model
- Predicting and refining: the trained model is eligible to provide determinations or predictions on new sets of data input. The results are first evaluated on the test set, which consists of known outputs, and based on the level of errors e.g RMSE the model is fine tuned.

Types of Machine learning

The main Machine Learning techniques are: supervised learning, which uses known input-output pairs of data to train the model and provide accurate predictions and unsupervised learning that identifies hidden rules/norms/patterns or coalescent structures in input data.

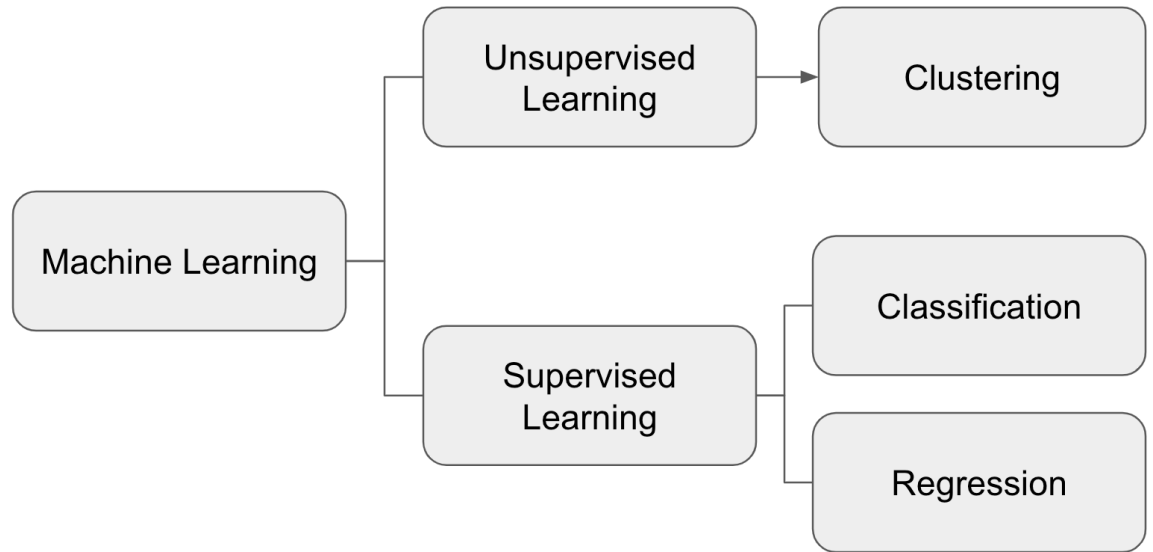


Figure 2. Machine learning techniques

Supervised machine learning

Supervised learning is a method that works with well classified and labeled training data to train an algorithm that can then produce a model to provide predictions to new datasets and determine the class labels of unseen data scenarios.

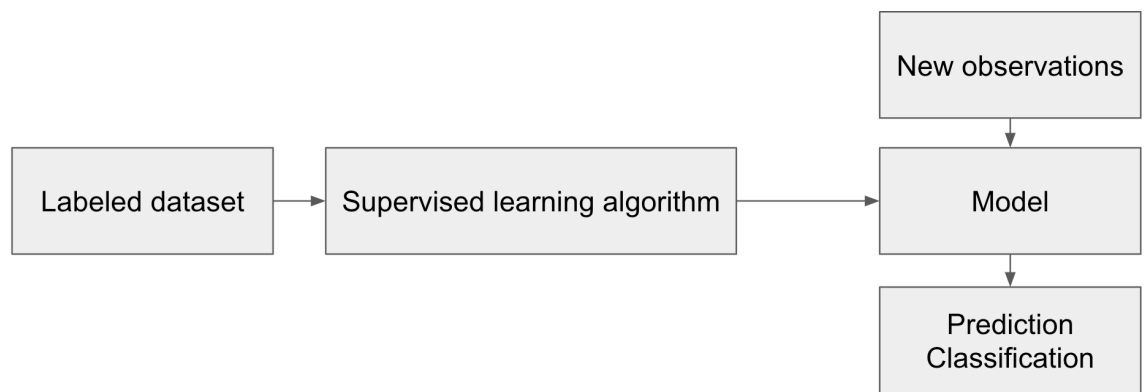


Figure 3. Supervised learning algorithm

The training process generates input - output pairs^[10]. The input object is transformed into a feature vector, which consists of a number of features representative to the object. Because of the “curse of dimensionality”^[11], the number of features should be kept at such a level that secures the accurate prediction of the output and keeps only the features that are important to the decision process. There are various supervised machine learning algorithms available like SVM, decision trees each with strengths and weaknesses. Thorough analysis of the problem should lead to the correct algorithm that should be used. Algorithms, after training and parameter’s adjustments should be validated on their accuracy using cross-validation^[12] techniques and finally the evaluation of the best algorithm should be measured against the test set. Supervised learning is well described [13], [14] with well established algorithmic tools. Classification and Regression techniques are heavily used in Supervised learning for developing predictive models.

Regression techniques are used for predicting continuous responses that may include forecasting and algorithmic trading. The output is a real value. It may provide useful insights in the real estate field for predicting the future house pricing, or weather predictions etc. Linear models are one of the most used regression algorithms including but not limited to decision trees and neural networks.

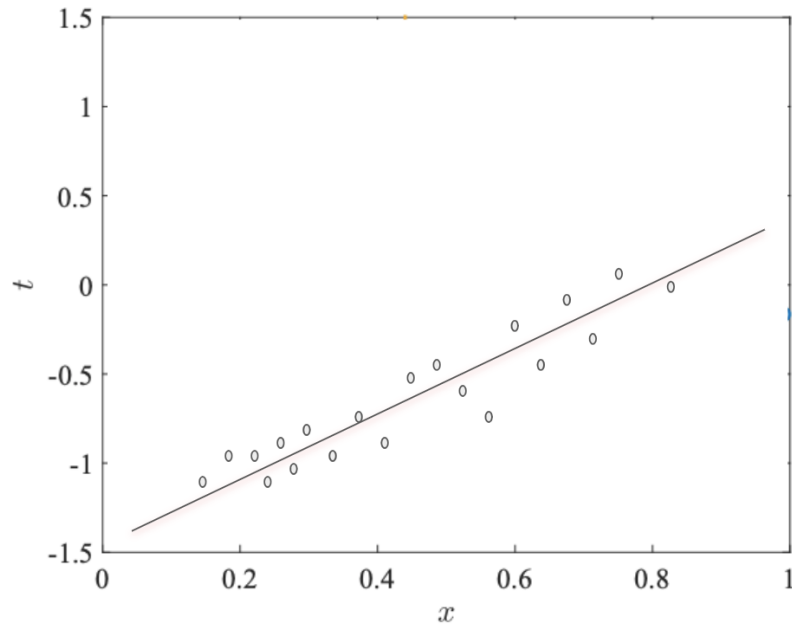


Figure 4. Regression problem

Training dataset D of N training points (x_n, t_n) is provided where $n = 1, \dots, N$. x_n represents the covariates (inputs) and t_n the dependent variables (output) where the regression problem to be solved is the prediction of the output t for every new input x .

Classification techniques are used when the data can be categorized to defined groups. Classification is close to regression with the main alteration that outputs t are distinguished variables that accept a confined number of potential values and defines the class that x belongs to.

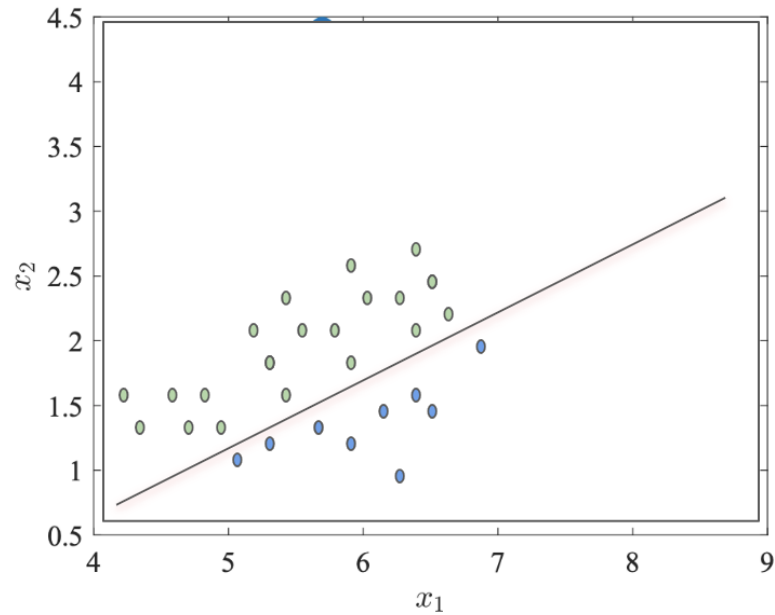


Figure 5. Classification problem

Logistic regression, k nearest neighbor (KNN), support vector machine (SVM) are some of the most ordinary algorithms to perform classification. Based on the supervised learning problem, the correct learning algorithm should be selected. There are **four major topics** to be taken into consideration during the selection:

Bias-variance tradeoff : Biased is a learning algorithm when for a specific input x , after training, the predicted output is constantly incorrect. On the other hand, a learning algorithm has big discrepancy when for a specific input x , the predictions are constantly different if different training sets are used. Bias and variance tradeoff should be adjusted either automatically or via parameters and weights adjustments in order to achieve more accurate predictions[15].

Training data in conjunction with function complexity: This issue is related to the combination of the quantity of training data and the complication of the “true” function (regression or classifier function). The “true” function can be very complex if it implicates complicated interactions among a lot of different input features and reacts differently in different areas of the input. A learning algorithm with high variance and low bias as well as a big amount of training data should be used in case of complex “true” function. In case of a simple “true” function, the training can take place using a small amount of data using algorithms with low variance and high bias and.

High dimension of the input feature version: It is a good practice to identify the features with the higher importance and use this subset for training the algorithm in case of many input feature vectors.

Noise in the output values: Overfitting can be caused in case of noise in the output values as the algorithm will try to learn from incorrect data and engage with a function that exactly matches the training data. The technique that is used to limit the noise is called “early stopping” and can obstruct overfitting by identifying and removing noisy training data before the training. Most common algorithms in supervised learning are SVM, logistic and linear regression, decision trees, Neural Networks, Similarity learning etc.

There are various algorithms that are used in supervised learning.

Logistic Regression is preferred in cases where we have binary classifications and there is a categorical dependent variable.

$$y = e^{(b_0 + b_1 * x)} / (1 + e^{(b_0 + b_1 * x)})$$

x is equivalent to the input values which are linearly combined applying weights / coefficient-values (B) for predicting the output variable y.

Support Vector Machines are used in cases of classification and regression statistical analysis (C-SVC) and (E-SVR) respectively. The data are represented in (n)dimensions where n is the number of the features and the feature's value is a specific coordinate in the space. The categorical classification of the class data point is achieved by using "hyperplanes" in the (n)dimensional space which defines the decision boundaries and the differentiation of classes in the most optimal way.

Naive Bayes is also used in classification tasks as a probabilistic classifier and has the concept that provided that B has happened, A has a probability to happen. It is supposed that the feature's value in the dataset is independent of any other feature's value, after providing the class variable.

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}$$

Decision Tree is included in the predictive models and has a tree base learning decision rules approach for solving classification and regression problems and predicting the target variable. The primary focus in the decision tree development is to define the root and sub-root levels and be careful of overfitting cases, which are very often in the decision tree algorithms.

Random Forest is like the decision tree classification algorithm without the usual problem of overfitting and with the difference of the randomization in finding the root node and distributing the features nodes.

Unsupervised learning

Unsupervised learning algorithms are applied on unlabelled datasets consisting only of inputs x_n , with $n = 1, \dots, N$, and the main objective is to identify previously unknown patterns in data in order to identify common properties of the data. The “principal component”[16] and “cluster analysis”[17] are the two main methods that are used in “unsupervised” learning. .

Cluster analysis identifies commonalities in the unlabelled data, parses each new data entry to find the existence of such commonality and respectively assigns it to the applicable group.

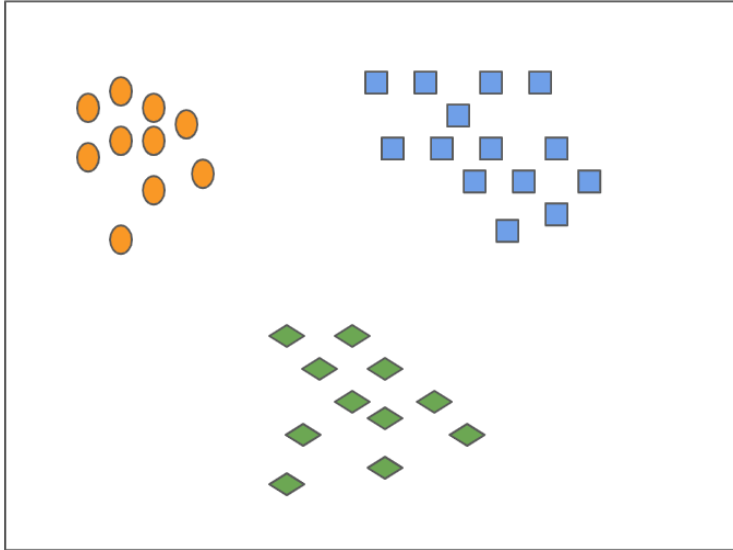


Figure 6. The result of a cluster analysis

Principal component analysis (PCA) aims to convert a set of observations into a set of values of linear unrelated variables called principal components using a rectangular transformation in the statistical process PCA . For Exploratory data analysis[18] and constructing predictive models this method is heavily used. The objective is that after the normalization using mean centering each variance of the variable to be equal to one, Z-score[19]. PCA can be achieved by eigenvalue decomposition[20] of a correlation matrix or singular value decomposition of a data matrix.

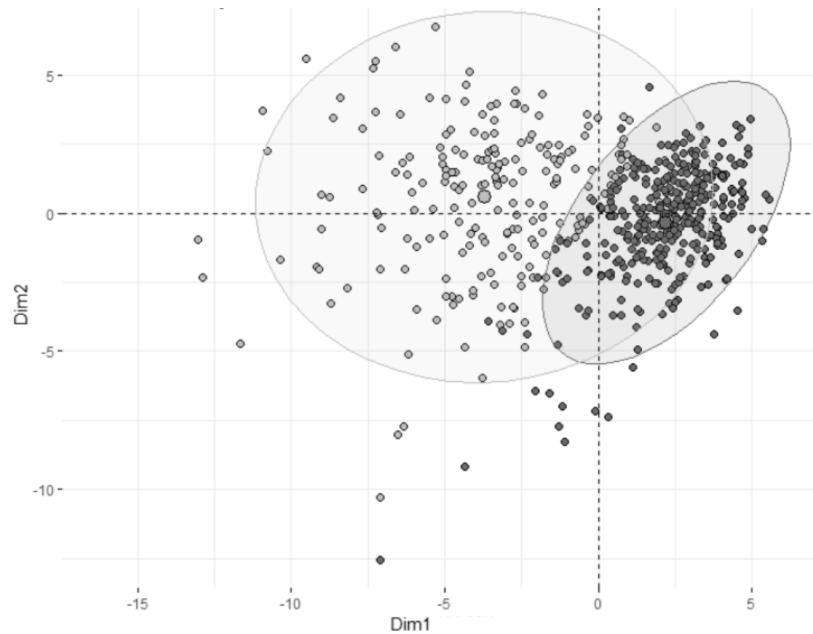


Figure 7. 2D PCA plot from 30 feature dataset

Most common algorithms in unsupervised learning for clustering is k-means and DBSCAN. For anomaly detection is the Local Outlier factor and for the Neural Networks the deep Belief Nets, Hebbian Learning, Generative adversarial networks etc.

NLP

One of the most commonly used artificial intelligence subfields is Natural Language Processing. Speech recognition and natural language understanding and processing are the main fields of the respective NLP algorithms. Every day huge amounts of textual data are produced. The categorization, clean-up and insights process is able to be streamlined with the help of NLP. NLP is the parsing and semantic interpretation of human generated text, allowing machines to learn, analyze and understand the context. By using NLP algorithms , valuable insights regarding the concept , the entities , the sentiment data analysis , in the form of voice or text, can be provided. The input and output of an NLP system can be voice, text, image.

NLP Techniques and Semantics

There are various techniques that are used heavily during the NLP process with the most common to be:

Grammar induction is a task that produces a formal grammar, grammar with no given context, that describes a language's syntax.

Lemmatization is used to identify word's lemma according to the meaning within the context by using techniques like removing inflectional endings, dictionary lookup etc. in order to return the word's base dictionary.

Morphological segmentation task is used to split words into individual elements and recognize it's class. Morpheme is the smallest meaningful unit of a language. It can be classified as free, when it can function independently as word or bound when it appears only as part of words which does not have a meaning on it's own.

Part-of-speech tagging technique that can identify words with similar grammatical properties, part of speech. The same word can be noun or verb or adjective

Parsing technique is used to provide the "Parse Tree" of a provided sentence, which is the trivial representation of the linguistic structure of a string. "Constituency Parsing" and "Dependency Parsing" are the two different types of parsing. "Constituency Parsing" adjusts on building out the "Parse Tree" utilizing a Probabilistic Context-Free Grammar according to Stochastic grammar. Whereas "Dependency parsing" concentrates on the words relation within a sentence.

Sentence boundary disambiguation or Syntactic Analysis is defining the boundaries of a sentence usually based on punctuation marks like comma, full stop etc.

Tokenizer breaks the sentence into smaller pieces like marks, words using "Syntactic Analysis" technique.

Stemming, in contrast with lemmatization that depends on correctly defining the sense of a word in a sentence, stemming is not so strict and is just enough to relate words to the same stem, even if this stem is not in itself a valid root.

Word segmentation process which is used to split sentences into words and counts the existence of each word in the sentence. “Bag of Words” or vectorization is based on the process which tokenises - vectorises words after split from sentences, eliminates punctuation , lower case words , counts how frequently each word appears and generate the matrix.

Terminology extraction is used to subtract from the text the respective terms.

Named-entity recognition is a classification task that identifies name entities in random text and assigned to predefined categories.

Natural language understanding (NLU) is a component of NLP, a subset of the understanding and comprehension part of natural language processing. It elucidates the concept of the input string and transforms the unstructured data to classified data assigning them to the appropriate intents. In order to distinguish the meaning, classify and derive to the correct intent from the provided input, specific techniques are used like sentiment and content analysis.

Algorithms for NLP

LSTM - Long short term memory

The LSTM model is based on “Recurrent Neural Network” (RNN) [21], a subset of neural network, with focus to remember previous input using feedback connections over random time intervals and predict the output . LSTM algorithm fits well on processing

and recognizing sequences of speeches, images, videos, handwriting type of data using “memory blocks” in the recurrent hidden layer. The “**memory blocks**” consist of memory cells, which are in charge of storing the network's current state using self-connections “**gates**” units that supervise the information flow in and out using input gates and output gates . **Input activations flow** is controlled by the input gate before reaching the memory cell. **Output gate** is responsible for controlling the cell's activation output flow towards the rest of the network. Logistic sigmoid[22] is usually the activation function that is used by the LSTM gates. **Forget Gate** plays the role of LSTM's safeguard in terms of memory and computational leaks during the self-recurring training. It prevents the continued processing of unsegmented to smaller chunks, subsequences, input streams, therefore defines the duration that the value remains to the cell.

Sequence 2 Sequence

S2S models are deep learning models that are used extensively in machine translation tasks. Seq2Seq technique converts a sequence of items like letters, words etc. to another. The S2S model consists of encoder / decoder modules. The encoder identifies the input's sequence context as a hidden state vector and submits it to the decoder in order to produce the output sequence. The encoder and the decoder most of the time are using RNNs , LSTMs, GRUs etc. as the task is sequence based. The encoder appends the final embeddings at the end of the sequence after processing the input sequence. The outcome from the encoder is sent to the decoder for prediction, and after every successful

prediction, the previous hidden state is used for the prediction of the next instance of the sequence [24].

Named Entity recognition model

This algorithm is used for the identification and classification of a given stream of text to named entities. NER model is able to identify various entities like places, people in the text or speech input dataset. NER model is trying to split the input into chunks of segments and classify them in defined categories as tokens without formatting.

Word Embedding

Word embedding consists of feature extraction algorithmic methods which are based on the logic that words that express the same meaning have a semantic lookalike relationship and almost the same vector space distance. Each single word is assigned to a real valued multidimensional vector in a defined lower dimensional vector space. Word Embedding is using various techniques and “Embedding Layer” is one of them.

“**Embedding Layer**” is using “Neural Language Modeling” (NLM) for language modeling and word’s classification with supervised Backpropagation algorithms on the front end of the NN applying also weight[25] and co-occurrence [25] matrix. The text that is required for the initialization should be pre-processed so that each single word has passed through the “One-hot”[41] encoding transformation. “One-hot” encoding transforms the word to a vector which has all zero cells except one which is used to spot the specific word (1xN). The dimensions that will be used are defined in the model and dictates the vector’s space size. This method needs plenty of training data and consumes a

lot of time-resources but will provide the learning to the embedding layer. Another method is the “**Word2Vec**” which accelerates the Word Embedding NN learning process by analyzing the contextual words relation taking into consideration the transformed word vector’s offset. It is using two models, the “Continuous Bag Of Words” (CBOW) model and the “Continues Skip-Gram” (CSG) model. The CBOW gains to foretell the word according to the context , thus CSG foretells the words that are around a specific word.

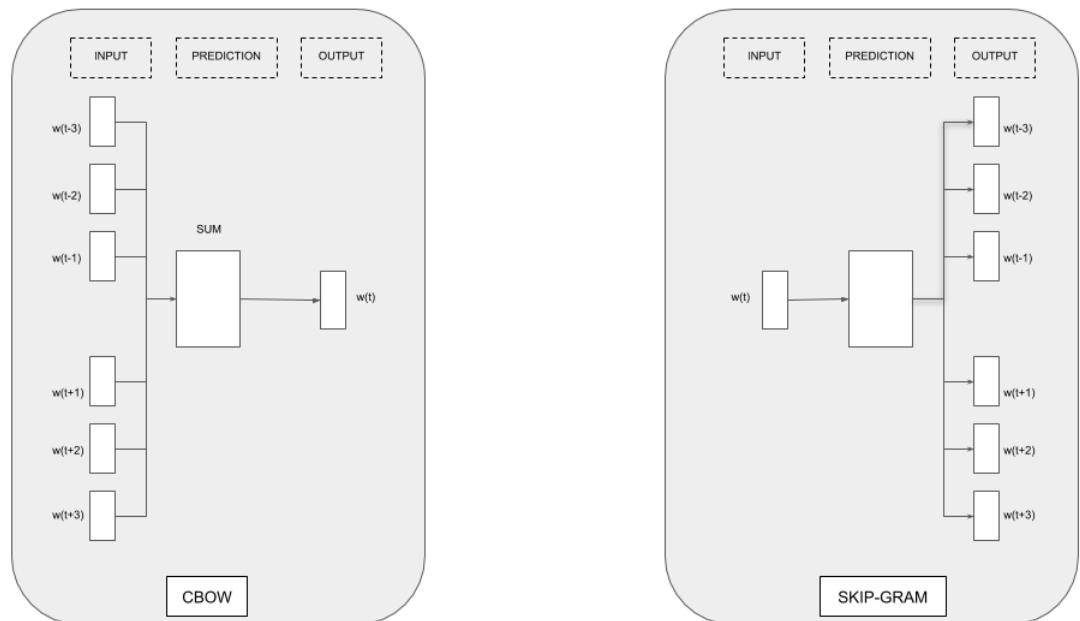


Figure 8a. CBOW vs SKIP-GRAM

Speech Recognition

Speech recognition aims to identify the words within an audio stream by applying language, pronunciation and acoustic algorithms.

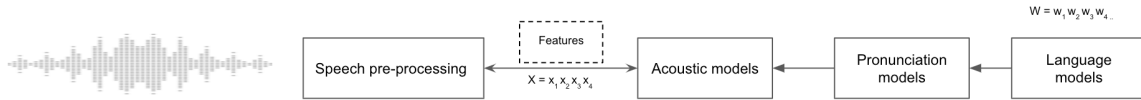


Figure 8b. Speech Recognition process flow

Figure 8b represents the sequence flow that is taking place during the speech recognition, where W represents the text sequences to audio features X . Depending on the component, alternative statistical models are used. Before the neural network evolution, the component that was heavily used for the language component was N-GRAM and for the acoustic, the Gaussian Mixture Model (GMM). Hence, if the analyzed audio features are X , most probable the text sequence is W that generated the audio features.

$$W^* = \arg \max p(X|W) p(W)$$

where $p(X|W)$ represents the acoustic model and $p(W)$ the language model which predicts the possibility of the relation between the words in the sequence. In order to accommodate the need of speech recognition in large vocabularies, the classification of the speech signal into smaller chunks is a need. The pronunciation models are used to extract the audio units front the corresponding words. In case of using NN, instead of GMM, DNN and LSTM are used, instead of N-GRAM, neural language models are used and instead of classical signal pre-processing convolutional models are used.

Connectionist temporal classification

Connectionist temporal classification (CTC) is based on RNN networks which is a must to use in case of data in a time sequence as the previous states and the historical learnings are memorized (“feedback-loop”) in order to predict the next. The problem that CTC came to solve is the need for the location alignment of each single character in the audio input that requires it to be in the precise location. CTC does not depend on the input's output's alignment and works with probabilities. It takes the highest probability of all possible alignments, using scoring and loss functionality, among input and label and summarizes them. These scores are used in the RNN backpropagation to adjust the weights.

Neural machine translation and Google NMT

Neural machine translation (NMT) is a subset of machine translation and is the most powerful algorithm to perform language translation with the power of deep learning and representation learning. It is using RNN to identify the probability of the input's text words and converts from one language to another. All parts of the neural translation model are trained jointly end-to-end to maximize the translation performance [27].

The Google Neural Machine Translation[28] is an improved NMT system which is using “example-based machine translation” (EBMT)[29] using millions of examples in order to improve the actual quality of the transactions. Around one hundred different languages are supported up to now.

Conversational Interfaces, ChatBots

ChatBot's objective is to use any applicable technology in order to mimic the conversation among human beings. The main components that the chatbot consist of are: the web or application interface in order to retrieve the input data, the natural language processing algorithms in order to analyze and segment the sequence of words or speech, the classification of the contextual meanings with entities which conclude to the flow selection (intents) , the response text or audio that will come directly from the chatbot's response implementation logic or programmatically using webhooks.

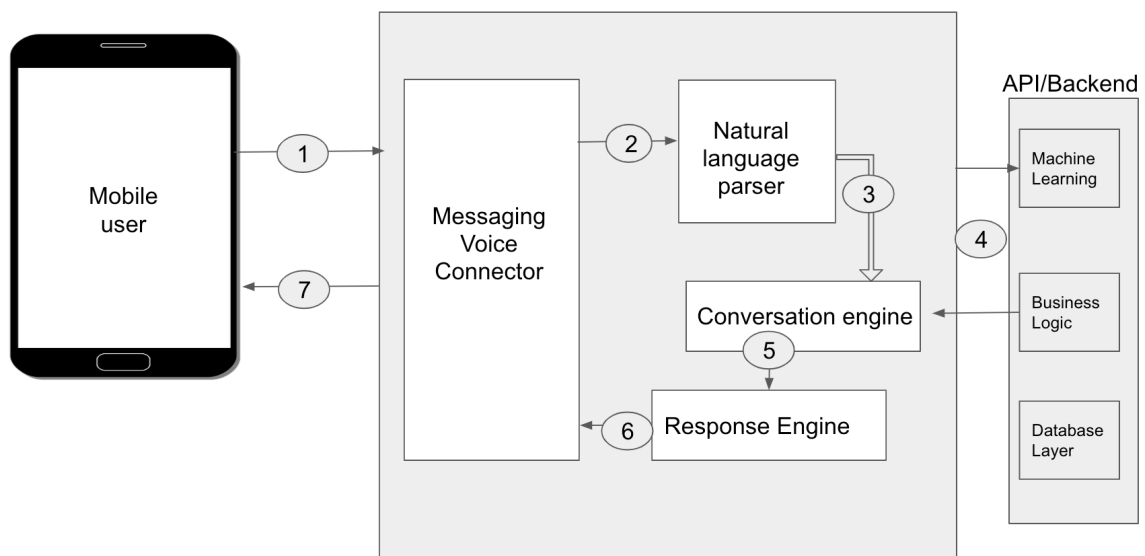


Figure 9. High level design of the chatbot's operating mode

Analysing *Figure 9*, mobile users are sending a voice or text message which is reaching the respective connector. The "Messaging Voice Connector" is converting speech to text

for further processing using the “Natural Language Parser”. The words are breaking down to defined keywords via the “Natural Language Parser” and the “Conversational Engine” is trying to fetch the correct dialog according to the segmented keywords that lead to the correct intent. User’s request is analyzed by the chatbot in order to locate the “Intents” and extract the “Entities”. This process of locating the “Intents” and extracting the “Entities”, is the fundamental prerequisite in the chatbot’s kernel. The “Conversational Engine”, either has a predefined response to serve via the “Response Engine” or it performs external requests to REST APIs to retrieve the response after the functional processing. For the design, training and optimization of the chatbot, human intervention is vital and plays a key role. Chatbot is trained to respond with the right answer, but if the input request is not understood then it may respond with the wrong answer. And at that point the retraining and the human intervention may take place , to identify the unknown words and assign them to the correct “intent”. One of the most powerful platforms for the chatbot creation is Google’s Dialogflow (GDF).

GDF is Google's conversational AI tool empowered by NLP technology. It provides tools to improve application interoperability with users through text and voice chat with AI technology. The platform handles standard protocols and functions that require grammar rules. The main features of DialogFlow are “Small Talk”, “Multilingual Agent Support”, “Cross Platform Support”, ”Fulfillment”, “Training”, “Agent Creation & Management”, “Entities”, “Intents”, “Integrations”, “In-Line Code Editor”, “Analytics”.

DialogFlow agent is the heart of the chatbot. It orchestrated the involved components according to the ML decisions that the agent will take. The main objective is to act as a human being agent, understand the human conversation and respond correctly to the free text/voice conversations. Using accelerated NLP algorithms it can break down the text/voice to classified keywords and define the route that should be followed and the response that should be provided. GDF can undertake the load of the call center agents, provide virtual assistance to all digital services for responding to frequently asked questions or more sophisticated decision driven domains like the health area.

Another interesting functionality that is provided out of the box is the “**Small Talk**” module which provides predefined answers to very simple daily dialogues without requiring any further development. Basic keywords for greetings and welcome intents are handed without any further development. GDF also provides multi language support with some languages to include also local settings, which cover versions of a specific region or country. Unfortunately, the Greek language is not currently supported. Another advantage of GDF is the out of the box integrations with the largest chat platforms out there including but not limited to “Facebook Messenger”, “Skype”, “Twitter”, “Slack”, “Viber”, “Google assistant”, “Amazon Alexa”.

Intents is one of the most critical modules in the GDF ecosystem. One agent can have many “intents” to manage the conversation flow routes. The intent section has the logic of identifying the most optimal route from user’s input keywords to the most relevant intent. These possible keywords and values that can describe an intent are stored under the “**training phrases**”. After the user’s input data segmentation and classification,

the agent will try to match the classified information to the best appropriate intent by matching against the “training phrases”. The agent’s AI NLP algorithms make sure that even the input classified phrases are not identical to the “training phrases”, the correct intent identification will be provided. The core modules of the intent are: “Context”, “Events”, “Training phrases”, “Actions and parameters”, “Responses”, “Fulfilment”.

The **context** provides the conditions that are required for different intents to be triggered. This is happening by passing input and output context variables to each intent where the one’s output variable is a prerequisite for the other’s input variable in order for the intent to be triggered. Therefore, the conversational flow can be rules and guided and the context will be activated so that only the intents that have declared input context can be selected if the previous intent had an output context. Therefore, not only the input text should be matched but also the context expected variable to be present. Using this technique, contextual driven flows can be developed. Contrariwise, by using the “**event**” module an intent can be selected based on predefined input keywords and not on text matching. This is usually used in the initialization process where there are specific conversational routes to be following according to the flow. Prerequisite for the agent to select the correct intent is the “**Training phrases**” module which should consist of phrases as close as possible to the user’s response for more accurate intent matching. The more accurate phrases the better results. Keywords from each “training phrase” are assigned to specific “**Action parameters**”. The “**Actions**” module is triggered when the intent is matched and the classified input text is matched with parts of the “training phrase”. These parts are classified as “**Parameters**” and according to the “**Action**”s

rules, loop back can take place till the parameter gets its value. The loop back is using “**Prompts**”, which are clarification questions to be answered from the user till the user responds with a valid answer for this step and the parameter takes a value. Each parameter is assigned to an “**Entity**” which can be system or custom type. The next step is the response creation. “**Responses**” is the stage of creating static and personalized responses. Fulfillment activation is required for generating rule based contextual responses. The response can be in different formats including text, images, audio and can define the end of the conversation. The responses can be selected from the list of available answers that have been inserted to the system manually, or can fetch the response content programmatically using webhooks and external APIs.

The following diagram shows the basic flow for intent matching and responding to the end-user:

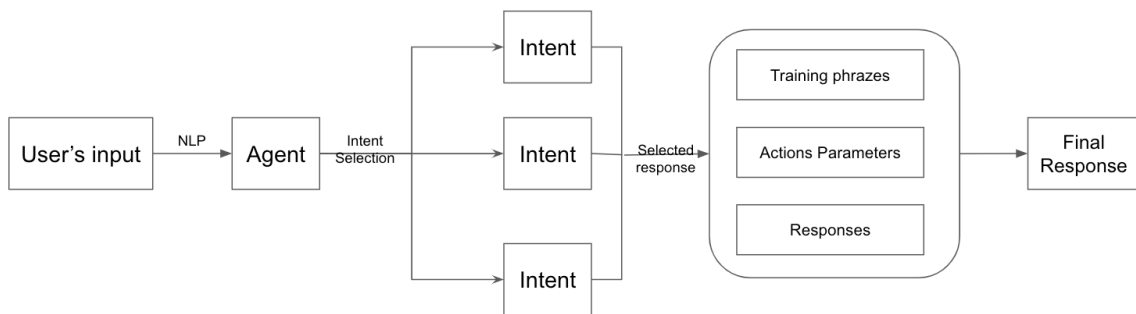


Figure 10. Basic flow for intent matching and responding

The objective of the “Agent“ is to analyze and segment the input data and proceed with the classification analysis of the existence of the keywords to the “training phrases’ of the

respective Intents. The outcome is the accurate Intent selection and the continuation of the dialog flow according to the intent's rules.

Follow-up intents is a subsection of the original intent and inherits the parent's intent context by automatically assigning the input and output context variables to the parent and follow-up intents respectively. The follow-up intent process is triggered only if the parent intent has been selected and can have multi-tree dimensions.

Entities is another important component in the GDF ecosystem and is used for matching and capturing key elements from the user's input data. During the generation of the "training phrases", the important keywords are mapped either to System or Custom entities. System entities are pre-defined data types that are universal like Countries, Cities, Age, email. System types guarantee the validation as well as the collection data process. In case of custom data types, custom Entities have to be used with predefined keywords and sub keywords for increasing the probability to collect the correct value during the keyword matching process.

Fulfillment is part of the "Response" module. If it is enabled, GDF instead of serving the response from the list of responses that has been configured to the system, it triggers the webhook service which invokes an external API call for further processing. Using this technique allows specific functionality to be applied prior to the final response to the user.

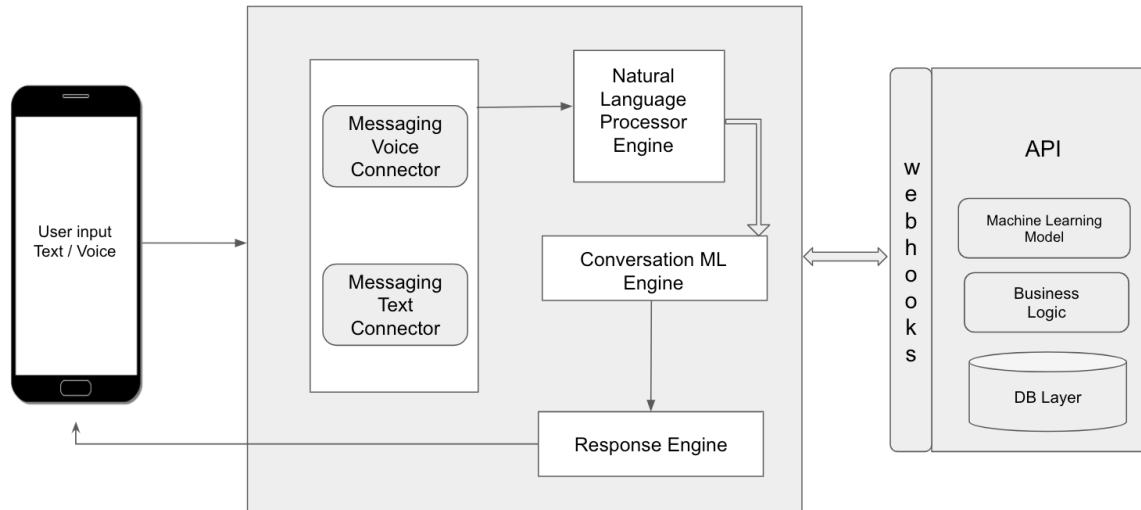


Figure 11. Processing flow with fulfillment backend interactions

Contribution

MV HealthBot is a platform that consists of multiple components in order to facilitate the need for telemedicine health care services. The objective is the patient to have an interface to interact with using human language and retrieve valuable information regarding his health condition, the probability of suffering from specific diseases according to his symptoms and the online interaction with the hospital institutes and the health experts.

The “MV Health Bot” platform is differentiated from the related works chatbot’s implementation as it is more robust , modularized, user friendly and end to end implemented. It is using machine learning algorithms for training the risk analysis models. The trained models are accessible via REST API for the predictions of the patient’s daily data as well as the Covid-19 risk analysis assessment. These predictions

are triggering instant notifications for immediate patient's updates regarding the health condition. Additionally, the MV Health bot may handle free text and speech inputs and ,by using NLP technology, to interpret and classify the input data to known symptoms. All this functionality is missing from the described implementations under the related work section therefore MV Health bot platform will drastically contribute to the healthcare ecosystem.

Chapter III. Design and Implementation

Proposed Architecture

The proposed architecture of the envisaged chatbot system is illustrated in Figure 12.

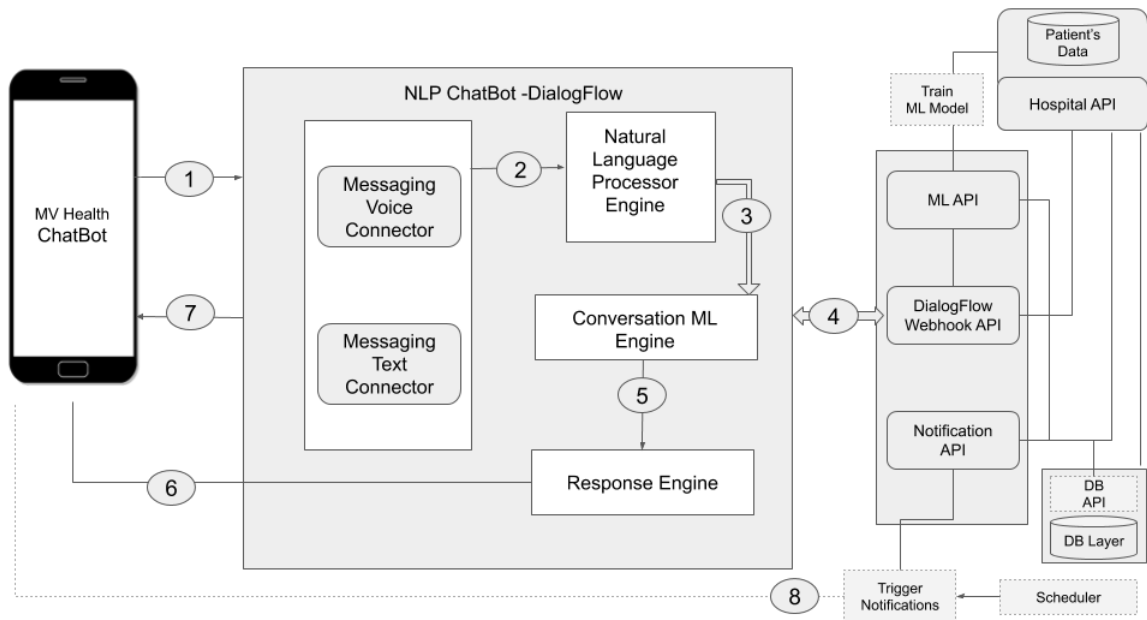


Figure 12. MV Health Bot Architecture

Modularized architecture diagram of all the components in the MV Health bot

The architecture diagram consists of the following components:

Mv Health ChatBot: is the voice and text-based conversational interface which is responsible for enabling the chatbot user to interact with the NLP platform

NLP Chatbot consists of the natural language processing components that orchestrate the identification and conversion of the input orders in text and speech format to intent classification in order to provide the most relevant response to the chatbot user.

Hospital API is the data source interface for retrieving patient's health data.

The "Train ML model" component is an asynchronous process that is used by the platform to generate and train the machine learning model using the hospital's data and the user's generated input data. The trained model is used for health condition predictions and is saved to the Cloud AI so that the ML API can have access to it.

The ML API is the interface that provides access to the trained ML Model via RESTful HTTPs POST.

The DialogFlowWebhook API is used by the NLP platform in order to pass the user's input dialog data and the classified intent to the webhook and retrieve the produced response message after applying the respective business logic that is assigned to the specific intent's handler.

The Scheduler is a crontab like scheduler that has the ability to publish a payload to a specific message queue topic. It is used to initiate the daily notification process.

The Notification API is listening to specific message queue topics. Once the scheduler publishes the message to the specific topic, the Notification API is triggered. It has two processes to follow. The first is to collect the daily health data for all the MV HealthBot subscribed users using the Hospital API. The second is to invoke the ML API and predict the user's health condition by passing to the trained model the user's daily data. If the ML prediction is abnormal the Notification API will send a notification to the respective user.

DB API is consumed from most of the processes in order to retrieve or store data to the database. The database collections that are stored to the database are:

- UserProfile, which holds all the personal user's metadata including his MedicalId
- HealthData, which holds the health care retrieved data from the Hospital API categorized per user per date
- UserDialog, which consists of the interview data that has been collected during the NLP dialog process
- UserTokens, which are specific device identifiers in order to send out the notification messages to the subscribed devices.

Implementation Details

There are three main activities that MV Health bot provides. The daily notifications that are sent to the users in case of health disorder based on the data that are collected via the Hospital API. The offline training of the ML model based on the Hospital's data. The ChatBot engine that provides the interface for the interaction with the end user using NLP , helps in gathering the symptoms as well as other valuable data and sends it to the doctor prior to their scheduled meeting. The scheduled meeting can also be set up using the chatbot.

MV Health Bot ML Model

There are two different models that have been trained for covering different needs. The first one is the Logistic Regression model that is used for the prediction of the Covid-19 disease and the second one is another Logistic regression model which is used for the prediction of heart diseases..

Covid-19 ML Model

The main objective of the model is to accurately predict if the user is suffering from Covid-19 according to the provided symptoms via the chatbot interface. For the training of the respective model the dataset that has been used consists of

- patient's unique identifier ,
- if he is experiencing fever or highFever ,
- if he is coughing
- if he is experiencing shortenOfBreath
- target, not positive in Covid-19 (0), positive with light symptoms (1), positive with severe symptoms that lead close or to death (2)

The data are in csv mode and are ingested to the Google Cloud BigQuery platform (BQ). BQ is acting as the database layer and the BQ ML module is used for ML model creation, training and prediction. The beauty of this environment is that the infrastructure comes out of the box and there are no limitations in terms of storage or processing power.

Model Algorithm

For the implementation and constant training of the Covid-19 model, Google BigQueries ML platform has been used. BQ ML provides the right algorithm per case. In case of forecasting, Linear regression is available. In case of classification, binary and multiclass logistic regression algorithms are available. K-Means is also provided for data clustering on unlabelled datasets and TensorFlow.

In the Covid-19 ML Model , the LOGISTIC_REGRESSION algorithm has been used . In the next section called “Experimentation and Results” more elaboration on the coding of the creation and evaluation of the model is provided.

Framework, storage

Framework: Google Cloud BigQuery ML

Storage: Google Cloud BigQuery

Model Type: Logistic Regression

Loss type: Mean log loss

Accuracy: 99%

Heart Disease ML Model

The main objective of the Heart Disease ML model is to accurately predict if the user is ailing from heart disease. The model has been trained and tested using several contributing risk factors such as abnormal pulse rate, diabetes etc.. The provided data are from real clinical cases.

Model Algorithm

In order to conclude to the best Heart disease ML Model various algorithms have been assessed. SVM, Naive Bayes, Logistic Regression, Decision Tree, Random Forest regression algorithms have been applied and evaluated according to their accuracy. According to the Mean Absolute Error as well as the scoring of each model against the

test set, Logistic regression appeared to be the most performing model and was selected to be used by MV Health Bot with 82% accuracy.

Framework, Storage

The Logistic Regression model has been developed using :

- Python Version: 3.7
- Framework: scikit-learn
- Framework Version 0.20.4

and is stored in Google Cloud Storage using the joblib library.

In order the model hosted in Google Cloud Storage to be used via APIs, Google Cloud AI platform has been used. Cloud AI platform hosts the trained ML models in the google cloud and uses Cloud AI service to infer target values for new data. AI Platform organises the trained models using resources called models and versions. Therefore multiple versions of the models can be uploaded to the AI platform and can be called via HTTPs protocol using REST API.

✔ logisticRegressorModel

Description	logisticRegressorModel_V1
Model	logisticRegressorModel
Model location	<u>gs://</u> <u>/logisticRegression/</u>
Creation time	
Last use time	
Python version	3.7
Framework	scikit-learn
Framework version	0.20.4
Runtime version	1.15
Machine type	Single-core CPU
Manual scaling nodes	1

Figure 18. Logistic Regression Model API metadata

APIs

The MV Health platform provides a set of REST APIs in order to accommodate the requested functionality.

- The ML Api enables the consumption of the trained ML model via HTTP requests, therefore any application can use the API by passing a relevant

HospitalData object as request body and retrieve the respective prediction which indicates if the patient's health condition is in order.

- The Hospital Api receives as HTTP request body the patient's Medical ID . Based on the Medical ID, the Hospital API searches the patients database for the daily health data and returns a json object of type HospitalData.
- The Notification Api consumes both the ML Api and the Hospital Api in order to provide scheduled daily notifications to the MV HealthBot subscribed users regarding their health status and their daily health reports.
- The DialogflowWebhook Api is the HTTP interface that receives from the NLP platform the input user's data classified intent, applies the business logic per intent and responds back to the end user with the response message.

Covid-19 ML API

The Covid-19 ML provides a risk analysis assessment on the probability of the Covid-19 infection. According to the WHO organization, the likelihood of someone to suffer from Covid-19 depends on the following symptoms:

- Fever / High Fever
- Cough
- Shorten of Breath

If these symptoms are present then the user belongs to a high risk Covid-19 group according to the number of parameters that take place to the acquisition.

MV HealthBot API is consuming the Covid-19 Model and is hosted at the Google Cloud as an HTTPS Function. The GCP function interacts with the Covid-19 ML model, which is generated via the BigQuery ML algorithms, using the Big Query SDK query command.

The API parameters are described below:

URL	https://us-central1-mvhealthbot.cloudfunctions.net
PATH	predictCovid19
METHOD	POST
REQUEST BODY	Array{fever, high fever, cough, shortenOfBreath}
RESPONSE BODY	{"predicted_target": 0 or 1 or 2 }

Heart Disease ML API

The Heart Disease ML API provides a risk analysis assessment on patient's heart disease status. The Cleveland dataset [35] from UCI has been used for model training and testing. The Heart Disease API input data consist of the following features:

{Age, Gender, Chest pain type: typical angina, atypical angina, non-anginal pain, asymptomatic, Resting blood pressure, Serum cholesterol, Fasting blood sugar level, Resting electrocardiographic results, Maximum heart rate achieved, Exercise induced angina, ST depression induced by exercise relative to rest, The slope of the peak exercise, ST segment: upsloping, flat, downsloping, Number of major vessels (0-4) colored by

fluoroscopy, Thalassemia: normal, fixed defect, reversible, diagnosis: 0=absence, 1, 2, 3, 4 = level of presency}

The Heart Disease ML API is hosted as a Google Cloud Function. It is a REST HTTP POST API that is consumed via the chatbot questionnaire. It invokes the Heart Disease Logistic Regression model which is hosted in the Cloud AI, feeds the model with the input data and responds back with the model's prediction if the user suffers from HD.

The API parameters are described below:

URL	https://us-central1-mvhealthbot.cloudfunctions.net
PATH	predictHeartDisease
METHOD	POST
REQUEST BODY	Array{'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal'}
RESPONSE BODY	{"predicted_target": 0 or 1 or 2 or 3 or 4 }

Hospital API

The Hospital API simulates the API that each hospital should provide in order external secure Health Applications to retrieve patient's data for process automation. Therefore, the Hospital API has been developed to accommodate the MV Health Bot functional requirements including but not limited to the on demand updates on the patient's heart disease status via the chatbot interface or via daily notifications. The Hospital API requests as an input only the medical patient's ID and is hosted as a Google Cloud Function.

The API parameters are described below:

URL	https://us-central1-mvhealthbot.cloudfunctions.net
PATH	hospitalApi
METHOD	POST
REQUEST BODY	JSON {medicalId}
RESPONSE BODY	JSON{'age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal'}

Notification API

Notification API is responsible for sending to the Health Bot users daily notification alerts regarding their actual daily health data and the respective prediction regarding their health condition. It invokes both the Hospital API to retrieve the health data by medicalId as well as the Heart Disease ML Api to get the prediction of the specific health dataset. The notification API is not accessible via HTTP but instead it is using the message queue logic - Pub/Sub. Therefore it is subscribed to a message queue topic named “gethealthdata” and listens to any published request. The Cloud Pub/Sub queuing architecture is an asynchronous messaging service designed to provide scalability, reliability, security and extensibility. Any application with the appropriate permission can publish an event to the topic “gethealthdata” and automatically the notification API will be triggered and the application that is subscribed to this topic will retrieve the payload.

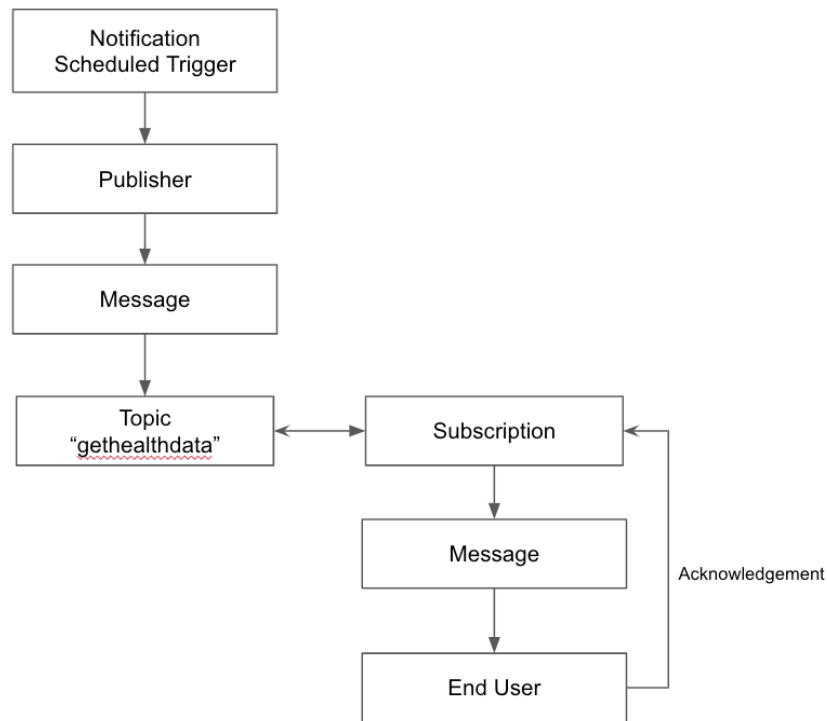


Figure 19. Message Queue Pub/Sub architecture

In MV HealthBot the trigger point which acts as “Publisher” is the Cloud Scheduler which is analysed in the next section The Cloud Scheduler triggers the publish process and sends a message to the topic “gethealthdata”. The Notification API is subscribed to this topic and is executed once it receives the message. The functional part is to call the Hospital API and retrieve the daily data . Then, it uses these data as the request body to the ML API in order to get the prediction regarding the patient's health condition. This information is sended as app notification to the device tray and it opens in the MV Health Chatbot message tray.

DialogflowWebhook Api

DialogflowWebhook Api is the HTTPs interface that can be assigned to the DialogFlow Intent and invoke programmatic functionality in it. It is used for intents that need to have additional functional processing before providing the final answer to the user. MV Health bot is using the Webhook Api in HealthCondition intent where the invoke of the Hospital Api and the Heart Disease ML Api is taking place in order to provide the response according to the model prediction. Additionally, Webhook intercepts the Covid-19 intent for providing the correct response according to the patient's replies and the assessment of the Covid-19 model which is invoked via the webHook handler . Finally, all the conversations are stored in the Firestore database in order for the doctor to revisit and recall the patient's interview.

MV HealthBot NLP

The NLP Chatbot is the core component of the MV Health bot platform. It accommodates the natural language communication with the patient via different interface types , Android - iOS application , web interfaces. It processes the text,voice input data using natural language processing algorithms and classifies the request to the respective intent via the intent classifier which is trained to analyze the phrase and identify the entity keywords within that phrase. Once this process is successfully completed, the response management component is triggered to fetch the appropriate response.

Flow Diagram

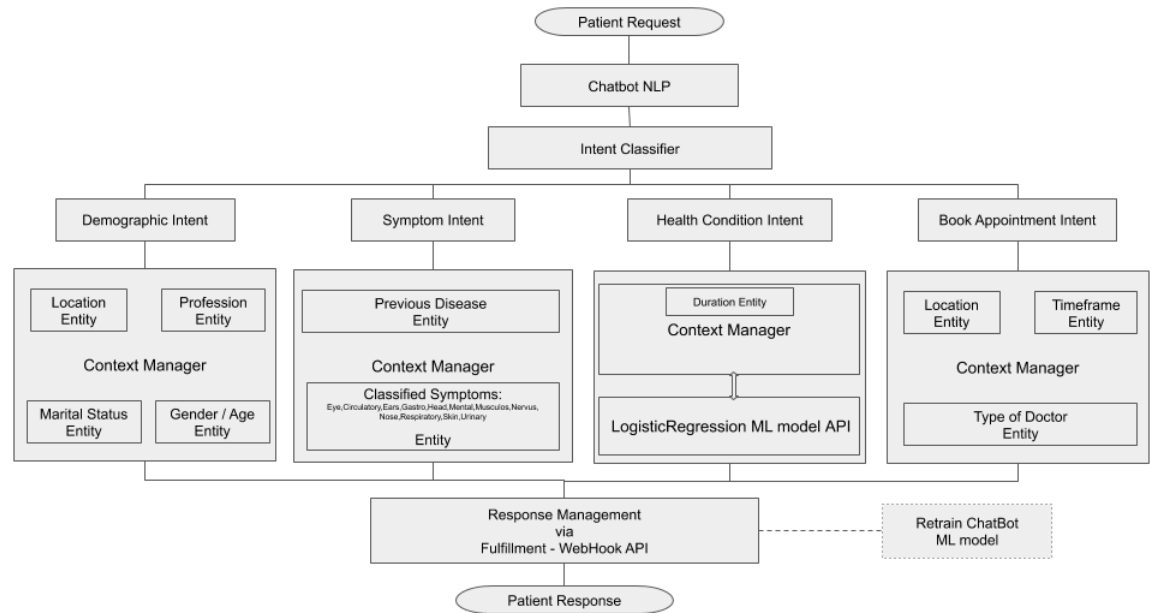


Figure 20. Health Bot Flow diagram

Intent Classifier

The intent classifier consists of four main intent categories. The demographic, symptom, health condition and book appointment intents. The NLP chatbot has been trained based on the training phrase set of each intent in order to match the input data with relevant phrases that can lead to the correct intent selection. Each training phrase may have word reference to keywords that belong to specific entities that can act as custom vocabularies.

Demographic Intent

The demographic intent is used as the first step of the patient's interview. The objective is to collect the information needed regarding the patient's personal data. The variables that are mandatory are listed below:

- Full name, this is important for the identification of the patient as well as to make the whole process more personalized, therefore the patient will feel more comfortable with the process.
- Age, which is important for the patient's age group classification
- Gender, which is important for the patient's gender classification
- Origin, the origin country may reveal correlation with diseases and symptoms
- Marital status, family status classification
- Job
- List of countries that the patient has traveled recently

Each of the questions above will lead to a set of mandatory contextual variables that should be collected before the chatbot proceeds further.

Action and parameters of demographic intent

PARAMETER NAME	ENTITY	VALUE
given-name	@sys.given-name	\$given-name
last-name	@sys.last-name	\$last-name

age	@sys.age	\$age
gender	@gender	\$gender
geo-country	@sys.geo-country	\$geo-country
marital-status	@marital-status	\$marital-status
occupation	@occupation	\$occupation
travel-geo-country	@sys.geo-country	\$travel-geo-country

List of parameters of the demographic intent with entity type and value name assignment

When the demographic intent is matched at runtime, the NLP process provides the extracted values from the input expression as parameters. Each parameter has an entity type which can be a system entity like @sys.geo-country or a custom entity like @occupation. The system entities have a prefilled vocabulary and pattern to accommodate and identify common and generic types like country, age etc. In case of questions related to custom categories there is no system entity populated to be matched against the input data. Therefore, custom entities have been generated :

@gender, which consists of male, female, other

@marital-status, which consists of a list of words and synonyms E.g. married, single, divorced etc.

@occupation, which consists of a list of available professional titles

The rest of the parameters are assigned to system types, therefore DialogFlow provides already the entities with the relevant lists of words and synonyms.

The demographic entity is trained using contextual training phrases that should be expected from patients as responses.

Patient History Intent


The Patient History Intent is responsible for collecting any known diseases that the patient is suffering from. This is a must question during the Health Interview and provides valuable insights for the patient's health status and the interpretation of the current symptoms.

Action and parameters of patient history intent

PARAMETER NAME	ENTITY	VALUE
diseases	@diseases	\$diseases

List of parameters of the patient history disease intent with entity type and value name assignment

The @diseases Entity is a custom entity that consists of a list of known diseases. E.g.

 Diseases	
emphysema pulmonary	emphysema pulmonary
encephalopathy	encephalopathy
endocarditis	endocarditis
epilepsy	epilepsy
exanthema	exanthema
failure heart	failure heart
failure heart congestive	failure heart congestive
failure kidney	failure kidney
fibroid tumor	fibroid tumor
gastritis	gastritis
gastroenteritis	gastroenteritis

The bot is trained to answer questions related to historical known diseases. Any recorded disease will be matched and populated under the \$diseases parameter as an array list.

Symptoms

The Symptom intent is collecting and classifying all the patient's symptoms and provides an assessment regarding his health condition, possible diseases and if he should visit a

doctor. The disease prediction is using the well trained ML models. Symptoms are split into vital health sections. Therefore the following Entities have been created to declare keywords and synonyms per body section.

Entities related to symptoms

ENTITY NAME	VALUE
circulatory	\$circulatory
ears	\$ears
eyes	\$eyes
gastrointestinal	\$gastrointestinal
head	\$head
mentalState	\$mentalState
mouthLaryx	\$mouthLaryx
musculoskeletal	\$musculoskeletal
nasal	\$nasal
urinary	\$urinary

List of entities related to the Symptoms entity.

The objective of the Symptoms Intent is to match the free text phrases and categorize the words into available entities. Therefore if the patient reports specific symptoms related to his gastrointestinal system, the symptoms will be gathered to the \$gastrointestinal variable. The identification and categorization of symptoms will help to have a more constructive report for the respective patient. The bot has to be trained with as many phrases as possible in order to be more accurate to the symptoms of variable matching.

HealthCondition

The healthCondition intent is responsible for providing the patient's health status. The intent is matched with commands related to “ what is my current health condition” or “how am I today?” . Once the HealthCondition indent is triggered, the bot using the webhook API will fetch the health data from the Hospital's API and by using the already trained Regression Model will respond to the health condition question with the respective prediction.

Once the input data is matched to the HealthCondition intent, DialogflowWebhook API is triggered in order to invoke the trained model, retrieve the data from the hospital's API and get the predictions from the model regarding the patient's health condition. Prerequisite for calling the hospital API is the Medical Id parameter. Therefore the bot insists on getting this information before responding with the health condition data and the prediction.

BookDoctor Intent

After the ML model concluded that the patient should see a doctor, the bot initiates the booking process. The desired questions are related to when (Date and Time) as well as the doctor type.

Action and parameters of patient history intent

PARAMETER NAME	ENTITY	VALUE
doctorType	@doctorTypes	\$doctorType
date	@sys.date	\$date
time	@sys.time	\$time

List of parameters of the bookDoctor intent with entity type and value name assignment

The @doctorTypes is a custom Entity that consists of all the available type of doctors.

DoctorTypes	
<input checked="" type="checkbox"/> Define synonyms ?	<input type="checkbox"/> Regexp entity ?
<input checked="" type="checkbox"/> Allow automated expansion	<input type="checkbox"/> Fuzzy matching ?
Podiatrist	Podiatrist
General Practitioner	General Practitioner
Pediatrician	Pediatrician
Endocrinologist	Endocrinologist
Neurologist	Neurologist
Rheumatologist	Rheumatologist
Allergist	Allergist, Immunologist
Psychiatrist	Psychiatrist
Nephrologist	Nephrologist
obstetrician	obstetrician, OB/GYN, gynecologist
Pulmonologist	Pulmonologist
Surgeon	Surgeon
Emergency Physician	Emergency Physician


The date and time are system entities and the default validations for these specific types are applied. If all the responses are in order, this step is declared as the end of the conversation , therefore all the collected contextual data is stored to the database under the specific patientId and also been emailed to the respective doctor.

MV Health bot Notification Alerts

Notification alerts are scheduled messages that are triggered in a specific timeframe and inform the user about his current health status. Notifications are triggered every 10am at the patient's local timezone. The process consists of the following components:

Cloud Scheduler

Cloud Scheduler is responsible for generating the trigger event that will initiate the notification process. It is using the same linux crontab patterns for time scheduling. The current configuration is '0 11 * * *' cron pattern and the timezone is set to Eastern European Summer Time (EEST). This means that every day at 11am EEST the trigger will be fired. Once the trigger is fired it publishes a message payload to the topic "gethealthdata" in order the Notification API to be triggered. The Notification API is subscribed to this topic therefore any message that is published will end to the Notification API listener.

 Cloud Scheduler
← getHealthDataScheduler

Description
Call external API every day at 11am EEST timezone

Frequency *
0 11 * * *

Schedules are specified using unix-cron format. E.g. every minute: "* * * * *", every 3 hours: "0 */3 * * *", every monday at 9:00: "0 9 * * 1". [Learn more](#)

Timezone *
Eastern European Summer Time (EEST) ▼

Target *
Pub/Sub ▼

Topic *
gethealthdata

Payload *
callApi

Cloud Notification

After the scheduler publishes the message to the “gethealthdata” topic , the Notification API is triggered. It calls the Hospital API to retrieve the patient’s health data and invokes the ML API to get the health condition prediction. In order for the notification to go out, Firebase Cloud Messaging (FCM)[37] solution has been used. FCM is the solution that reliably delivers messages cross platform. Each user that has signed up to the MV HealthBot Application has been granted with a unique token. The user has to give his consent that is willing to receive notifications before the token is provisioned. This token is being saved to the Firestore[36] database under the user's profile collection for further processing. The Notification API iterates to the user’s database to retrieve their medicalId and their Unique Token and generates the notification

custom message for each individual user that is subscribed to the MV HealthBot. Once the notification message is ready, the FCM service is invoked to publish the notification messages.

The MV HealthBotApp subscribes to the notification event at the initialization of the app. It initializes a listener that is subscribed to any onMessageReceived event that the Notification API publishes.

Chapter IV. Experimentation and results

MVHealthBot models have been fine tuned using various different algorithms and datasets till the best outcome. The process is splitted to data preprocessing and normalization, model training and testing, model consumption via API, chatBot UI implementation using NLP models for classifying the patient's input, data storage for further assessment.

ML Models

Heart Disease ML Model

Dataset Pre-process & Analysis

Cleveland heart dataset [35] from UCI has been used for model training and testing.

Table 1. Cleveland dataset.

Dataset Characteristics:	Multivariate
Attribute Characteristics:	Categorical, Integer, Real
Associated Tasks:	Classification
Number of Instances:	303
Number of Attributes:	75
Missing Values:	Yes
Creators	<ol style="list-style-type: none">1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.

	4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.
--	-------------------------------------------------------------------------------------------------

As per the UCI recommendation out of the 76 attributes only the 14 are significantly important.

Table 2. Cleveland dataset variable.

“age”	int64	Age of individual in years
“sex”	int64	Gender 1 = male 0 = female
“cp”	int64	Chest pain type 1 = typical angina 2 = atypical angina 3 = non-anginal pain 4 = asymptomatic
“trestbps”	int64	Resting blood pressure in mm Hg
“chol”	int64	Serum cholesterol in mg/dl
“fbs”	int64	Fasting blood sugar level > 120 mg/dl 1 = true 0 = false
“restecg”	int64	Resting electrocardiographic results 0 = normal 1 = having ST-T 2 = hypertrophy

“thalach”	int64	Maximum heart rate achieved
“exang”	int64	Exercise induced angina 1 = yes 0 = no
“oldpeak”	float64	ST depression induced by exercise relative to rest
“slope”	int64	The slope of the peak exercise ST segment 1 = upsloping 2 = flat 3 = downsloping
“ca”	int64	Number of major vessels (0-4) colored by fluoroscopy
“thal”	int64	Thalassemia is an inherited blood disorder that affects the body’s ability to produce hemoglobin and red blood cells. 3 = normal 6 = fixed defect 7 = reversible defect
“target”	int64	the predicted attribute, diagnosis of heart disease 0 = absence 1, 2, 3, 4 = present

The dataset has been checked for null values. Four and two null values have been detected in the ca and thal features respectively (df.isnull().sum()) and updated with the mean value.

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	1	145	233	1	2	150	0	2.3	3	0.0	6.0	0
1	67	1	4	160	286	0	2	108	1	1.5	2	3.0	3.0	2
2	67	1	4	120	229	0	2	129	1	2.6	2	2.0	7.0	1
3	37	1	3	130	250	0	0	187	0	3.5	3	0.0	3.0	0
4	41	0	2	130	204	0	2	172	0	1.4	1	0.0	3.0	0

Figure 13. First five observations of the dataset

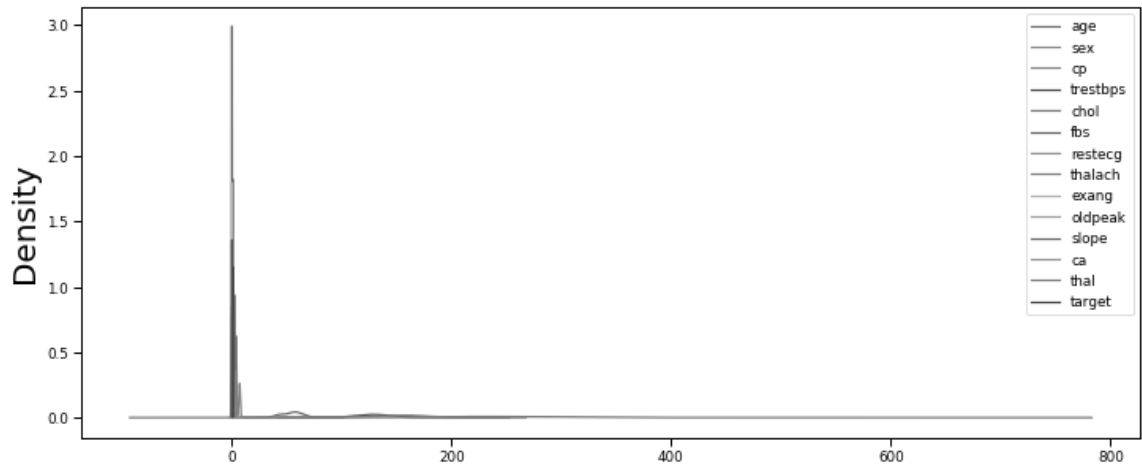
	count	mean	std	min	25%	50%	75%	max
age	303.0	54.438944	9.038662	29.0	48.0	56.0	61.0	77.0
sex	303.0	0.679868	0.467299	0.0	0.0	1.0	1.0	1.0
cp	303.0	3.158416	0.960126	1.0	3.0	3.0	4.0	4.0
trestbps	303.0	131.689769	17.599748	94.0	120.0	130.0	140.0	200.0
chol	303.0	246.693069	51.776918	126.0	211.0	241.0	275.0	564.0
fbs	303.0	0.148515	0.356198	0.0	0.0	0.0	0.0	1.0
restecg	303.0	0.990099	0.994971	0.0	0.0	1.0	2.0	2.0
thalach	303.0	149.607261	22.875003	71.0	133.5	153.0	166.0	202.0
exang	303.0	0.326733	0.469794	0.0	0.0	0.0	1.0	1.0
oldpeak	303.0	1.039604	1.161075	0.0	0.0	0.8	1.6	6.2
slope	303.0	1.600660	0.616226	1.0	1.0	2.0	2.0	3.0
ca	303.0	0.672241	0.931209	0.0	0.0	0.0	1.0	3.0
thal	303.0	4.734219	1.933272	3.0	3.0	3.0	7.0	7.0
target	303.0	0.458746	0.499120	0.0	0.0	0.0	1.0	1.0

Figure 14. Dataset's statistical data

The target column has been normalized so that any value from 1 and above is equal to `df['target'] = df.target.map({0: 0, 1: 1, 2: 1, 3: 1, 4: 1})`.

MV Health Platform has created the `HospitalData` object that the features of the CSV file can be cast to.

```
interface HospitalData {  
    medicalId : number;  
    age : number;  
    ca : number;  
    chol : number;  
    cp : number;  
    exang : number;  
    fbs : number;  
    oldpeak : number;  
    restecg : number;  
    sex : number;  
    slope : number;  
    thal : number;  
    thalach : number;  
    trestbps : number;  
}
```



*Figure 15. Density diagram
Gaussian normal distribution is observed according to the mean value close to zero*

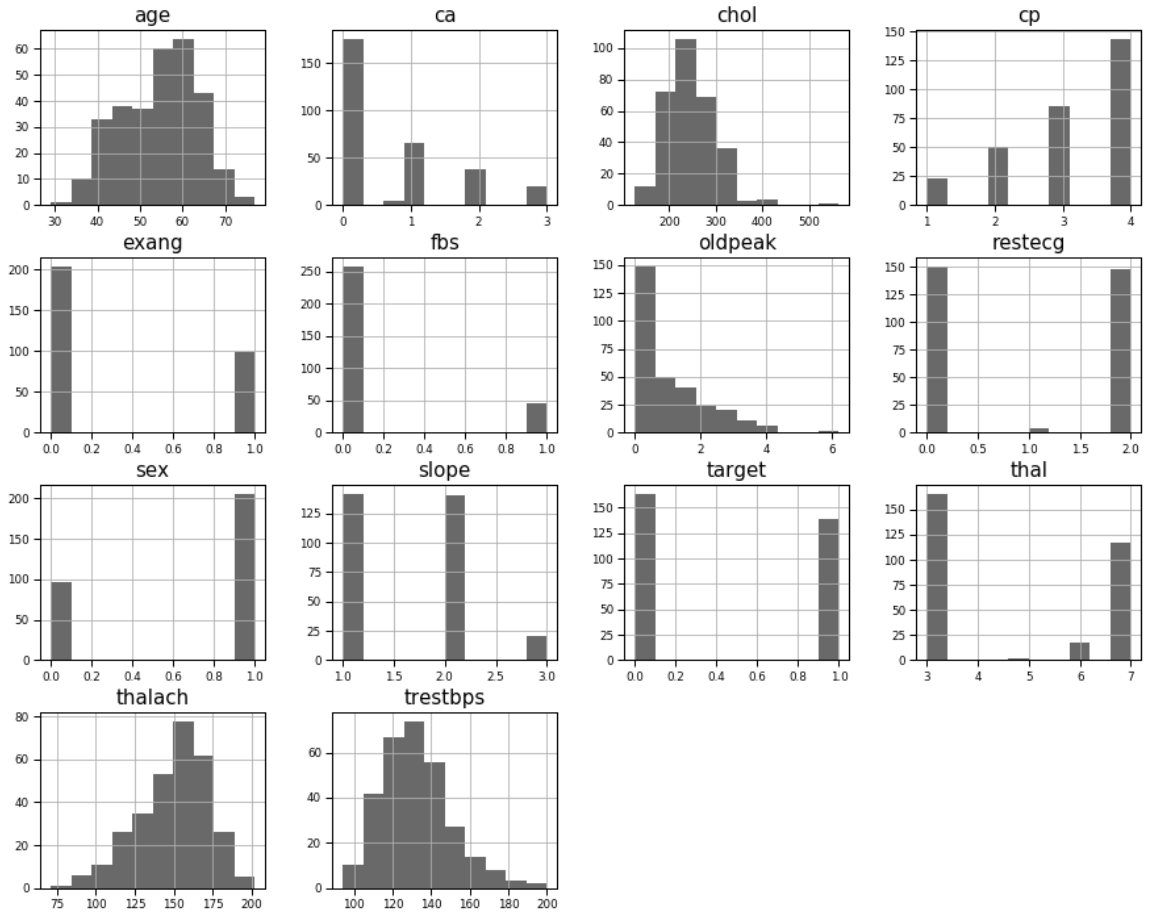


Figure 16. Histogram

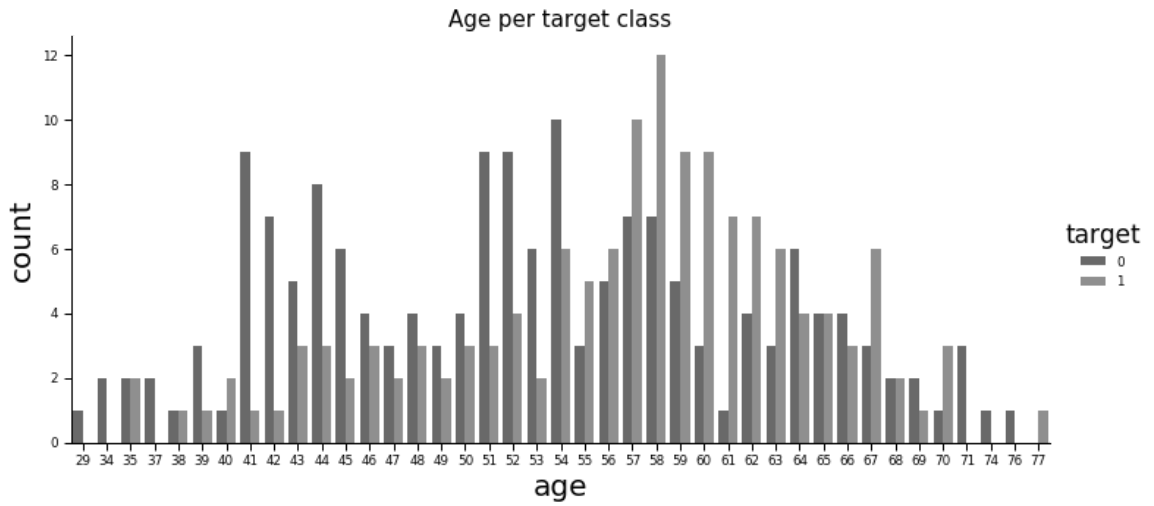


Figure 16a. Age per Target class plot

Figure 13 shows clearly that 57 , 58 and 67 years old have a burst in suffering from heart diseases.

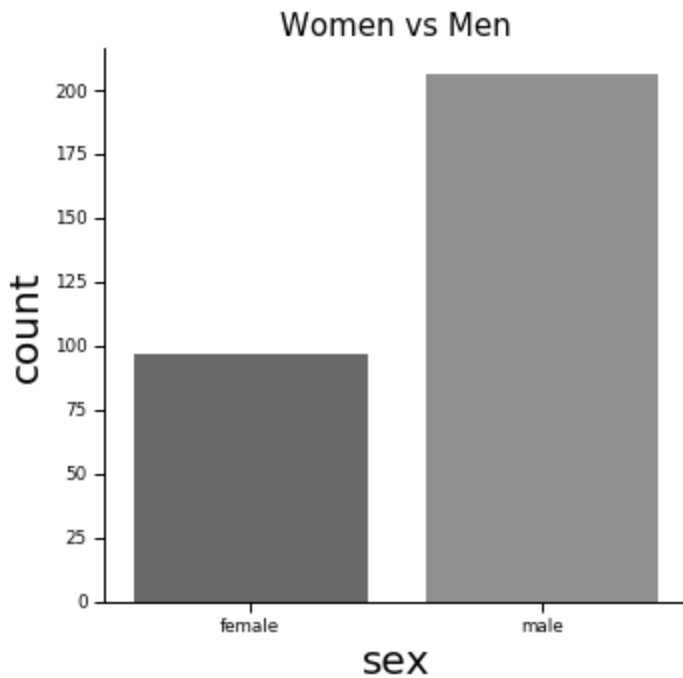


Figure 16b. Distribution of sex observations

Percentage of Female Patients: 32.01%
Percentage of Male Patients: 67.99%

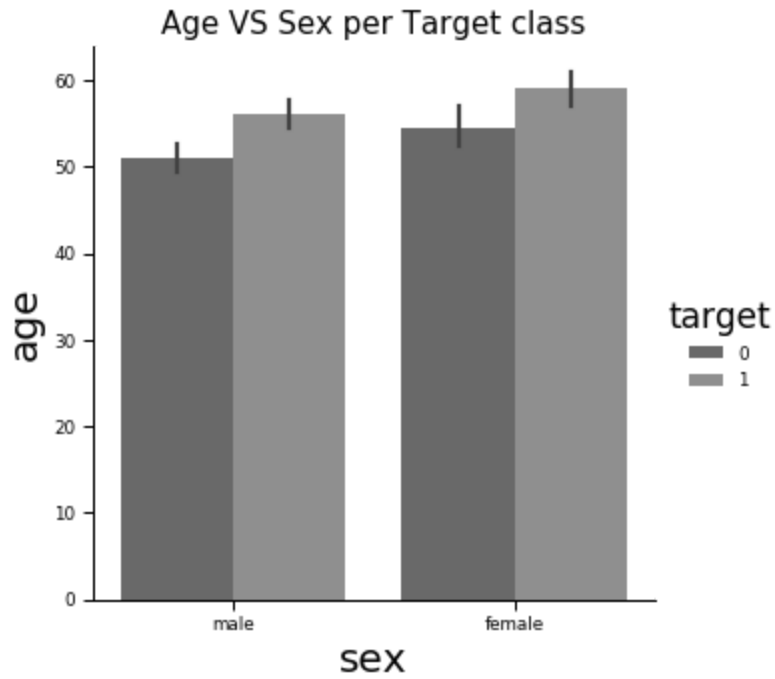


Figure 16c. Age versus Sex per Target class plot

Figure 15 shows that women with heart diseases are in higher age than men,

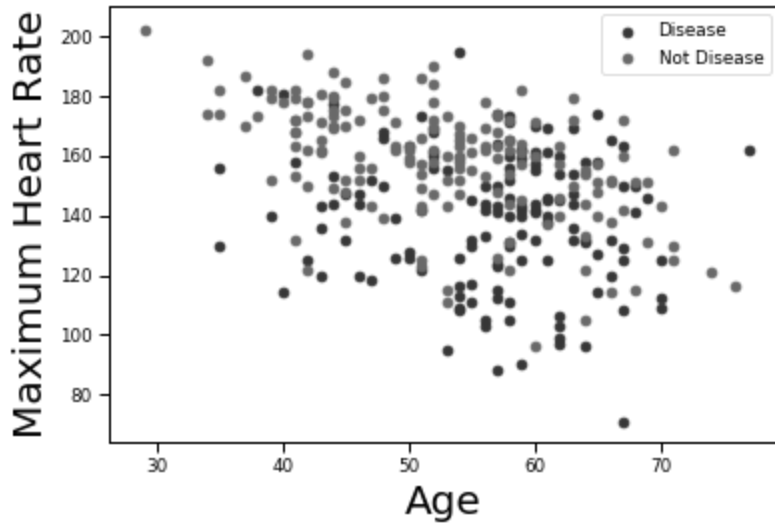
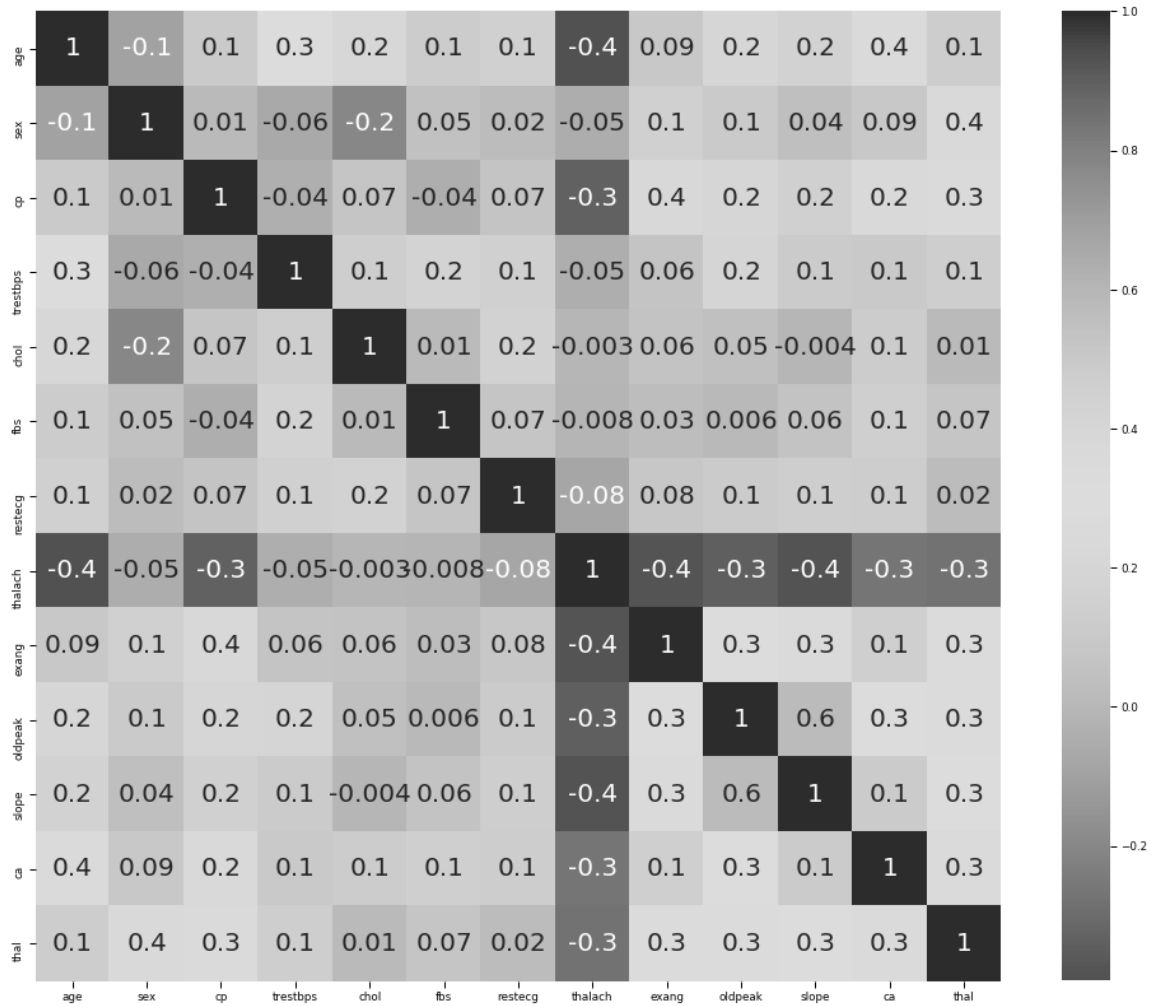


Figure 16d. Scatter plot for disease status per age



*Figure 17. Correlation matrix
Correlations coefficients between the possible pairs of variables*

The Correlation matrix demonstrates the correlation coefficients between the variables. In order to train and evaluate a model's accuracy the dataframe should be split into 33% train set and the rest test set. Multiple algorithms have been tested before concluding to the most accurate model.

Model Creation and Comparison

For the implementation and training of the Heart Disease ML model, five different supervised algorithms have been used. The objective is to predict the “target” variable which is binary, whether the patient has heart disease or not.

Models comparison

The table below contrasts the models that have been trained and according to the Mean Absolute Error as well as the scoring of each model against the test set, Logistic regression is the most performing model and will be used by MV Health Bot.

Model	MAE	Accuracy
Logistic Regression	0.18	82%
SVC	0.44	56%
Naive Bayes	0.2	80%
Decision Tree Classifier	0.22	78
Random Forest	0.19	81

Comparison of the trained model accuracy

Covid-19 ML Model

Dataset Pre-process & Analysis

The Covid-19 dataset has the following structure.

Table 3. Covid-19 dataset.

Dataset Characteristics:	Multivariate
Attribute Characteristics:	Integer, Real
Associated Tasks:	Classification
Number of Instances:	2100
Number of Attributes:	5
Missing Values:	No
Creators	Dataset simulation according to WHO organization symptoms risk analysis

Table 4. Covid-19 dataset variable.

medicalId	Unique Identifier	The unique medical ID of the patient
fever	bit	If the patient has fever 1 = true 0 = false
high fever	bit	If the patient suffers from high fever 1 = true 0 = false
cough	bit	If the patient has symptom of coughing 1 = true 0 = false
shorten of breath	bit	If the patient experienced shortness of breath
target	int64	the predicted attribute, diagnosis of Covid-19 0 = absence 1 = warning 2 = carantine

The data have been ingested to Google Big Query in order to be normalized and been used for the Logistic regression model. The logisticRegression Model has been produced directly from the Google Cloud ML using the embedded autoML algorithms.

Row	uid	fever	highFever	cough	shortenOfBreath	target
2094	F0DA1E7C-1CDD-46EC-B89D-6D6896917F07	0	1	1	1	2
2095	17FC0878-93B9-4A69-89E5-585AFC0B2439	0	1	1	0	2
2096	C91ED5FB-A54D-4609-9FBA-9685B5FDBEAD	0	1	0	1	2
2097	6A74038A-13DA-48C2-BCE3-BA0E1ACAB673	0	1	1	0	2
2098	E63C8401-70BB-4AF1-B2CF-754FB065C16C	1	1	0	1	2
2099	BE2FA04E-208C-4E36-85E4-943B6995AE08	1	1	0	1	2
2100	400471B2-BF79-4B45-A822-F2B3F1268775	0	1	0	1	2

Figure 21. First five observations of the dataset

Model Creation, Evaluation and Prediction

The Big Query ML is executing a query on the provided dataset with declared target attribute and exports the model.

```
CREATE OR REPLACE MODEL `covid.ml_model`
OPTIONS(model_type='logistic_reg', input_label_cols=['target']) AS
SELECT fever,highFever,cough,shortenOfBreath,target FROM `covid.covid19`
```

This query generated the logistic regression model called covid.ml_model.

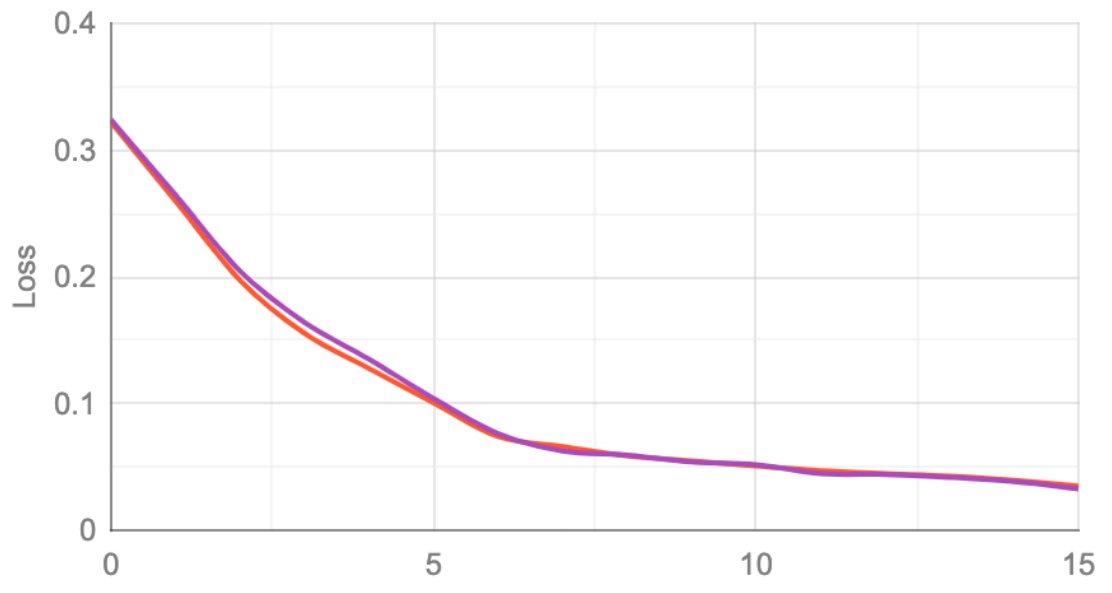
The confusion matrix below shows the percentage of actual labels that were classified correctly and incorrectly. The last column shows the percentage of total samples for the corresponding actual label.

Actual labels	Predicted labels			% samples
	0	1	2	
0	100%	-	-	48.56%
1	-	100%	-	26.32%
2	-	-	100%	25.12%

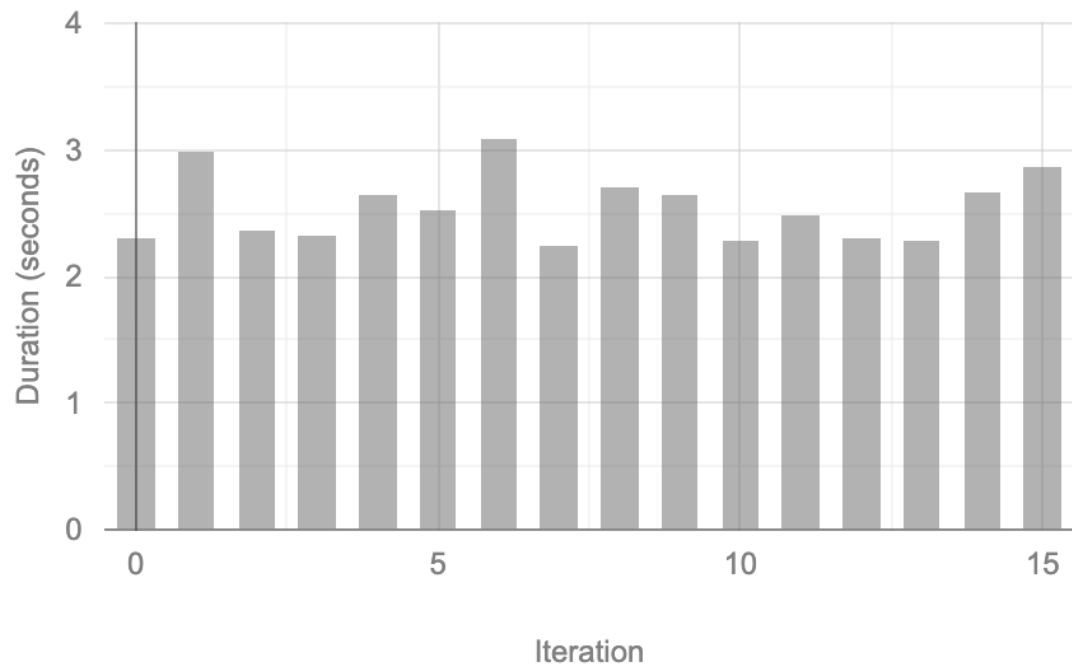
Training Process

Training Options	
Max allowed iterations	20
Actual Iterations	16
L1 regularisation	0.00
L2 regularisation	0.00
Early stop	true
Min relative progress	0.01
Learn rate strategy	Line search
Line search initial learn rate	0.10

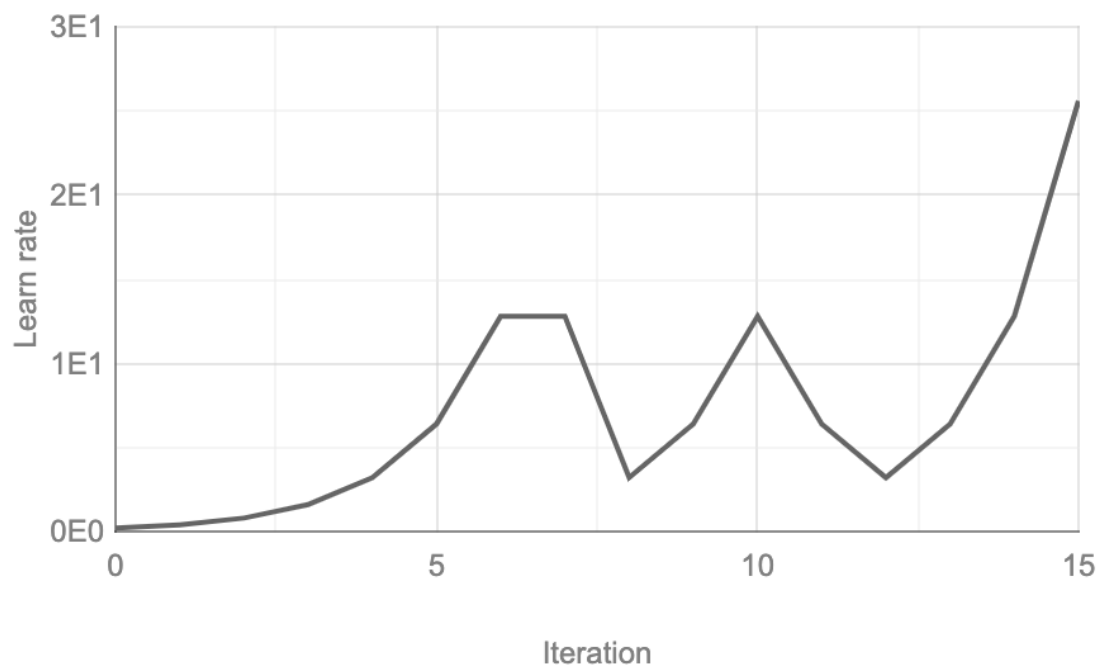
Loss



Duration (seconds)



Learn rate



Evaluation

Aggregated Metrics	
Threshold	0.0000
Precision	1.0000
Recall	1.0000
Accuracy	1.000
F1 score	1.000
Log Loss	0.1198
ROC AUC	1.000

Prediction

In order to validate the model, Big Query provides out of the box the PREDICT function in which you can select your own row values and get the probability of the results.

```
SELECT * FROM ML.PREDICT(MODEL `covid.ml_model`,(SELECT 0 as fever,1 as highFever,1 as cough,1 as shortenOfBreath)) limit 1;
```

The selected predicted target value is equal to “2” with 0.995 probability, which means that the patient is at high risk and should be in “quarantine” . The rest of the options have very low probability figures.

predicted_target	predicted_target_probs.label	predicted_target_probs.prob	fever	highFever	cough	shortenOfBreath
2	2	0.9949533021525833	0	1	1	1
	1	0.005046696468325623				
	0	1.3790911796499367E-9				

MV Health Bot ChatBot

The chatbot provides the interfaces for the patient to input his data and receive the appropriate response. The chatbot analyzes the input using natural language processing and provides the respective answer. The input is matched to the relevant intent and according to the context, the chatbot either provides a predefined answer or invokes the webHook API to fetch the programmatic response. The webhook API invokes the correct intent handler and applies the appropriate business logic which may also consist of the ml model predictions.

Demographic Intent Dialog

The demographic Intent is responsible to collect the personal data that are needed to identify the patient. It works with input text as well as speech recognition.

hi

Welcome, May I know your name please?

My name is Maria Vasiliou

Hi Maria, could you please let me know your age

I am 43 years old

May I know your gender? Male or Female?

Female

Where are you from? What is your country?

Greece

What is your marital status? Single, married, widowed, divorced, separated ?

single

What is your job? If not applicable you can respond with other

CTO

To which countries you have traveled recently?

italy, uk

Great Maria , I have recorded till now that you are 43 year years old, marital status: single, your origin is from Greece and you have traveled to : Italy and United Kingdom. Now send me a comma delimited list of historical diseases if any, otherwise respond with none

Ask something...



The intent has been trained to the respective answers and the different patterns. The process has been executed correctly as the response shows that all the parameters have been assigned to the correct values:

<< Great **\$given-name.original** , I have recorded till now that you are \$age years old, marital status: **\$MaritalStatus**, your origin is from **\$geo-country** and you have traveled to : **\$travel-geo-country**. Now send me a comma delimited list of historical diseases if any, otherwise respond with none >>

The information regarding the \$given-name of the user will be needed in all other intents in order to prompt the patient in a more friendly way by using his name. Therefore, there is a context variable defined called, cntidentity, which is used as output context variable.

Contexts 



Add input context

5 cntidentity  Add output context



Patient History Intent

The final response from the Demographic intent prompts the user to continue by sending a list of known historical diseases that the patient has been suffering from.

I have been diagnosed with
Gastroesophageal Reflux and Celiac disease

Thanks Maria !! I have recorded:
gastroesophageal reflux and Celiac Disease
. Please respond with the list of symptoms
that you are experiencing.

Ask something...



The matched disease entity values will be passed through using the context parameter, cntdisease, to the next entity.

Contexts ? ^

cntidentity ⊗ Add input context

5 cntidentity ⊗ 5 cntdisease ⊗ Add output context ×

Symptoms

The patient is prompt to input the symptoms that he is experiencing. The Symptoms Entity has a list of available symptoms that will be matched against the input parameters.

I have red eyes and I feel a terrible headache

Do you want to report more symptoms?
Please send me more symptoms. I have recorded: red eyes, headache. If not, thank you very much for your cooperation and the clear answers. If at any point in time you want to check your current health just ask for it. You can also book a doctor's appointment if you want to.

Ask something...



Health Condition

The health condition intent is used form the patient in order to get the results from his latest medical exams. It invokes the WebHook API consuming the healthConditionHandler. The handler expects as an input the medicalID. It makes a call to the hospital API to fetch the heart disease data for the specific medicalID and invokes the heart disease model to get the prediction for the specific dataset.

how is my health condition today?

What is your medical id?

my medical Id is 5424324

Your Resting blood pressure is:120, your cholesterol: 231 and your Fasting blood sugar level < 120 mg/dl.Results from ML model shows that with this health data you should see a doctor. Should I book an appointment for you now? .

Book Doctor's appointment

The doctor's appointment is an additional intent where the patient can define the type of doctor, the date and the time that the appointment would like to take place. The NLP is clever enough to identify the parameters either to the same sentence or if not, to ask various questions till it gets all 3 desired parameters.

can you book an appointment with a cardiologist for the day after tomorrow at 4pm

Done, I've booked you the best Cardiologist for 2020-04-01T12:00:00+02:00 at 2020-03-31T16:00:00+02:00

Covid-19 Application

Covid-19 application has been designed with a more UI friendly approach, using predefined options for selection. It requires the patient to be authenticated or at least to get a unique session id as a guest.

Authenticate

1:02 ↗



Email

Your email address

Password

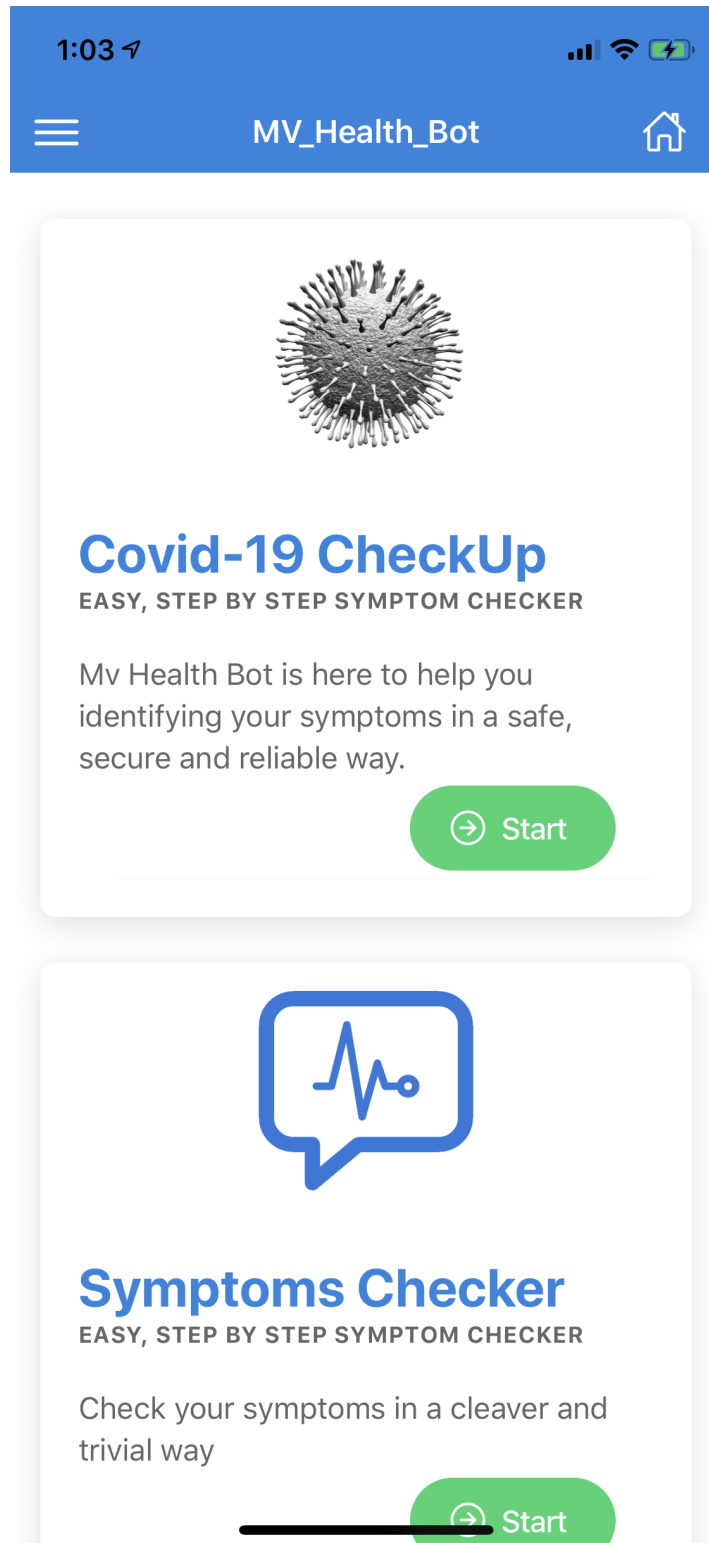
Your password

Login

 Register

 Proceed as Guest

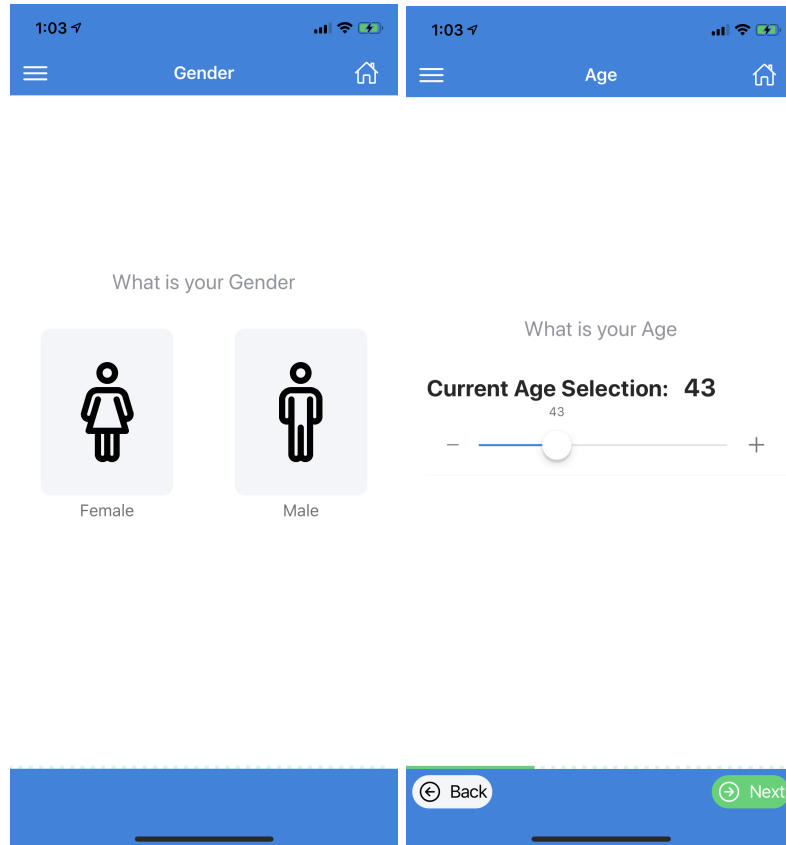
Menu Selection

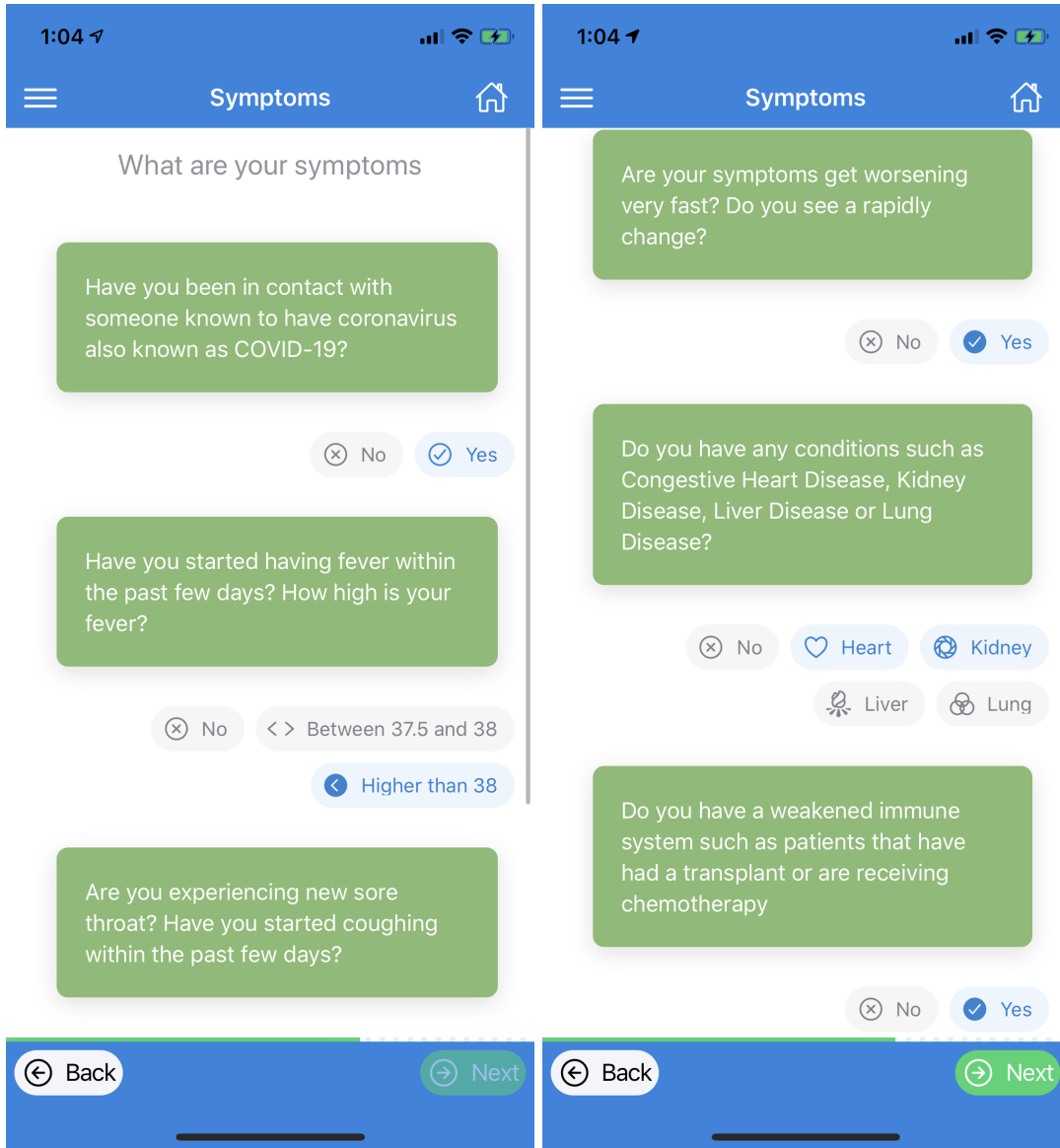


Covid-19 CheckUp

The application is requesting the patient to provide his gender and age value.

Then the dialog with the Covid-19 appropriate dialog according to the WHO organization starts.





The responses are collected and are used to invoke the Covid-19 Model for retrieving the prediction according to the input variables. In the specific case the selected symptoms contribute to the highest probability of the patient being positive to Covid-19. Therefore the patient is getting a red flag to urgently visit a hospital and the next steps that should be followed immediately.



CheckUp Results

Based on the symptoms you should urgently visit a hospital.

surroundings

high fever

cough

shorten of breath

rapid change

Heart Disease

Kidney Disease

weakened immune system



What Should you do immediately

Seperate yourself from people and pets in your house

Call the emergency number

Wear a surgical mask



Back



Home

Conclusion

MV Health Bot has been implemented to provide a different window in the current way that healthcare interviews, symptom collection and diagnosis take place. It is vital to understand that by digitizing the health sector and providing the patient's data for AI purposes, more powerful predictive ML models will be provided and more diseases will be identified and prevented on time by analyzing the patient's symptoms. This is the objective of MV Health Bot, by using an intuitive web and app interface, by speaking to the patient in his natural language, by matching and diagnosis according to the symptoms potential diseases using the well trained ML models , to provide patients with an end to end healthcare tool.

[Bibliography/References/Works Cited.]

- [1] Turing, Alan (October 1950), "Computing Machinery and Intelligence", *Mind*
- [2] Hutchins John, the Georgetown-IBM system, 7th January 1954
- [3] Reifler, Erwin (February 2–5, 1960). "The solution of MT linguistic problems through lexicography". *Proceedings of the National Symposium on Machine Translation*.
- [4] Dostert, Leon E. (1955). "The Georgetown – I.B.M. experiment, 124–135". *Locke and Booth*.
- [5] Gupta, N.; Nau, D. (1992). "On the Complexity of Blocks-World Planning"
- [6] Winograd, Terry (1971-01-01). "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language"
- [7] Harnad, Stevan (1992), "The Turing Test Is Not A Trick: Turing Indistinguishability Is A Scientific Criterion"
- [8] SHRDLU - <http://hci.stanford.edu/winograd/shrdlu/>
- [9] Y. Bengio; A. Courville; P. Vincent (2013). "Representation Learning: A Review and New Perspectives".
- [10] Stuart J. Russell, Peter Norvig (2010) *Artificial Intelligence: A Modern Approach*, Third Edition, Prentice Hall ISBN 9780136042594.
- [11] Richard Ernest Bellman; Rand Corporation (1957). *Dynamic programming*. Princeton University Press. p. ix. ISBN 978-0-691-07951-6.
- [12] Kohavi, Ron (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann. 2 (12): 1137–1143. CiteSeerX 10.1.1.48.529
- [13] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [14] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S.

- Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, "A Closer Look at M
- [15] G. James (2003) Variance and Bias for General Loss Functions, *Machine Learning* 51, 115-135.
- [16] Jolliffe I.T. *Principal Component Analysis*, Series: Springer Series in Statistics, 2nd ed., Springer, NY, 2002, XXIX, 487 p. 28 illus. ISBN 978-0-387-95442-4
- [17] Everitt, Brian (2011). *Cluster analysis*. Chichester, West Sussex, U.K: Wiley
- [18] Behrens-Principles and Procedures of Exploratory Data Analysis-American Psychological Association-1997
- [19] Glantz, Stanton A.; Slinker, Bryan K.; Neilands, Torsten B. (2016), *Primer of Applied Regression & Analysis of Variance (Third ed.)*, McGraw Hill
- [20] Hayde, A. F.; Twede, D. R. (2002). Shen, Sylvia S. (ed.). "Observations on relationship between eigenvalues, instrument noise and detection performance". *Imaging Spectrometry VIII. Proceedings of SPIE*. 4816: 355. Bibcode:2002SPIE.4816..355H. doi:10.1117/12.453777.
- [21] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8), Nov. 1997.
- [22] Raul Rojas. *Neural Networks - A Systematic Introduction (PDF)*. Retrieved 15 October 2016.
- [23] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, pp. 115–143, Mar. 2003.
- [24] Jan Chorowski, Navdeep Jaitly "Towards better decoding and language model integration in sequence to sequence models".
- [25] Lebre, Rémi; Collobert, Ronan (2013). "Word Emdeddings through Hellinger PCA". *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. 2014.
- [26] T. Sainath et al., "Convolutional neural networks for LVCSR," *ICASSP*, 2013.
- [27] Kyunghyun Cho; Bart van Merriënboer; Dzmitry Bahdanau; Yoshua Bengio (3 September 2014). "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches"

- [28] Wu, Yonghui; Schuster, Mike; Chen, Zhifeng; Le, Quoc V.; Norouzi, Mohammad (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation".
- [29] Carl, Michael; Way, Andy (2003). Recent Advances in Example-Based Machine Translation. Netherlands: Springer.
- [30] Ahmed Fadhil (2018). Beyond Patient Monitoring: Conversational Agents Role in Telemedicine & Healthcare Support For Home-Living Elderly Individuals
- [31] Divya S et al., (2018) "Self-Diagnosis Medical Chatbot Using Artificial Intelligence"
- [32] Krishnendu Rarhi (ORCID: 0000-0002-5794-215X), Abhishek Bhattacharya, Abhishek Mishra, Krishnasis Mandal (2018) "Automated Medical Chatbot"
- [33] Maria das Graças Bruno Marietto, et al.,(2013) , "Artificial intelligence markup language: a brief tutorial"
- [34] Benilda Eleonor V. Comendador, , et al., (2015), "Pharmabot: A Pediatric Generic Medicine Consultant Chatbot "
- [35] Dua, D., Karra Taniskidou, E., 2017. UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA <http://archive.ics.uci.edu/ml>
- [36] Cloud Firestore database, <https://firebase.google.com/docs/firestore>
- [37] Firebase Cloud Messaging, <https://firebase.google.com/docs/cloud-messaging>
- [38] V.Manoj Kumar et al., (2016) "Sanative Chatbot For Health Seekers"
- [39] Juanan Pereira, Óscar Díaz, (2019) "Using Health Chatbots for Behavior Change: A Mapping Study"
- [39] Flora Amato, et al., (2019) "Chatbots meet eHealth: automatizing healthcare"
- [40] Kavitha B. R., et al. , (2019) "Chatbots meet eHealth: automatizing healthcare"
- [41] "One-Hot Encoding". www.sciencedirect.com. Retrieved 2020-01-20