babel

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION AFFILIATED TO VTU, BELGAUM, APPROVED BY AICTE GOVT. OF KARNATAKA)

YELAHANKA, BANGALORE-560064

A Project Report on

DISEASE DIAGNOSIS SYSTEM BY EXPLORING MACHINE LEARNING ALGORITHMS

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

ALLEN DANIEL SUNNY	1NT14CS015
SAJAL KULSHRESHTHA	1NT14CS136
SATYAM SINGH	1NT14CS141
SRINABH	1NT14CS160

Under the Guidance of

Mr. MOHAN B. A.

ASSOSIATE PROFESSOR

COMPUTER SCIENCE AND ENGINEERING

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(Accredited by NBA Tier-1)

2017-18

Contents

List of Figures

DECLARATION

We hereby declare that

1. The project work is our original work.
2. This Project work has not been submitted for the award of any degree or examination at any other university/College/Institute.
3. This Project Work does not contain other persons data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This Project Work does not contain other persons writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
   (a) their words have been re-written but the general information attributed to them has been referenced;
   (b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.
5. This Project Work does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

Signature

ALLEN DANIEL SUNNY	1NT14CS015	
SAJAL KULSHRESHTHA	1NT14CS136	
SATYAM SINGH	1NT14CS141	
SRINABH	1NT14CS160	

Date :

ACKNOWLEGDEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. I express my sincere gratitude to our Principal Dr. H. C. Nagaraj, Nitte Meenakshi Institute of Technology for providing facilities.

We wish to thank our HoD, Dr. Thippeswamy M.N for the excellent environment created to further educational growth in our college. We also thank him for the invaluable guidance provided which has helped in the creation of a better project.

I hereby like to thank our guide Mr. Mohan B A, Associate Professor, Department of Computer Science Engineering on his periodic inspection, time to time evaluation of the project and help to bring the project to the present form.

Thanks to our Departmental Project coordinators. We also thank all our friends, teaching and non-teaching staff at NMIT, Bangalore, for all the direct and indirect help provided in the completion of the project.

Signature

ALLEN DANIEL SUNNY	1NT14CS015	
SAJAL KULSHRESHTHA	1NT14CS136	
SATYAM SINGH	1NT14CS141	
SRINABH	1NT14CS160	

Date :

ABSTRACT

In this project, aspects of the design of a system for diagnosis of Common disease that can be detected by the Doctor and patient on entering the symptoms into the symptom. Before developing the system, an analysis is done by comparing the machine learning algorithms on Disease-Symptom Knowledge Data-set prepared by New York Presbyterian Hospital of patients admitted during 2004. Based on the analysis the most accurate algorithm is used in the system to achieve the reliability. Also, in this report we will be discussing the hurdles we faced to achieve the required results, as in our project the machine learning algorithms are implemented on the pure text based data set. This project can possibly help doctors and patients as well, as early detection is beneficial for right treatment and early recovery.

Chapter 1

INTRODUCTION

1.1 Background

Machine learning is one of the key methods used in modern day analysis. With the explosion of the tech industry, and the rise of big data; it became necessary to analyze and predict trends. Slowly over the years machine learning has branched out into almost every major industry, and performs functions that were almost unheard, compared to a mere few years ago.

Machine learning uses various algorithms in order to process the vast amount of data that is generated. These algorithms are usually developed from a statistics perspective, and have only recently been co-opted by machine learning. Machine learning algorithms are broadly classified in four types. Regression algorithms and classification algorithms.

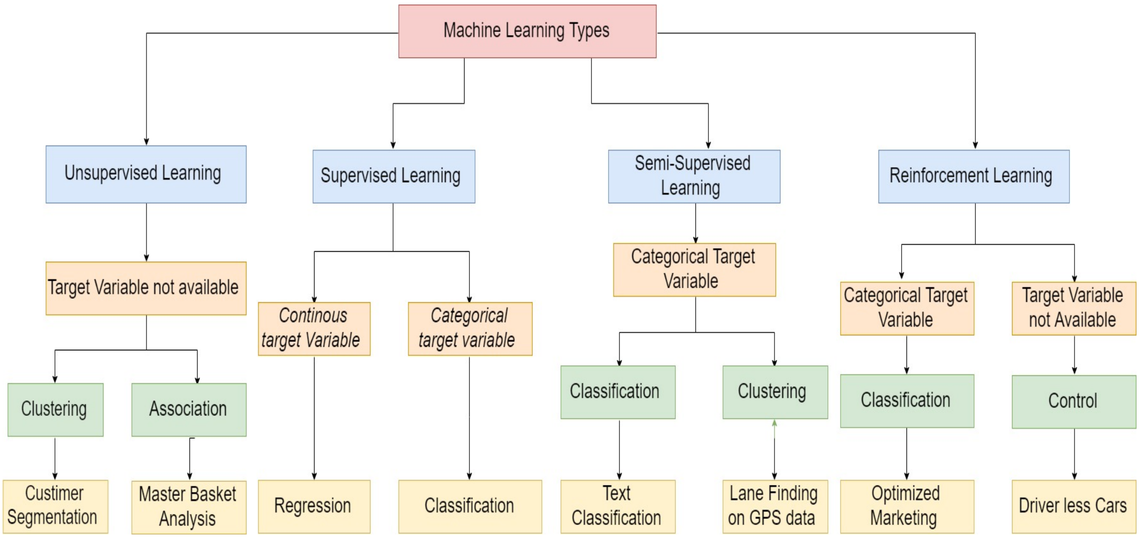
- Supervised learning- Supervised learning uses the data that has been previously inputted by the user, and based on the said input, generates a required output. The output can be guessed by the algorithm based upon the previous patterns of input and output. For example, Nave Bayes and Decision trees.

- Unsupervised Learning- In this case the computer is trained with unlabeled data , and is particularly useful when the user does not have a clear idea on how to process the given data or what to look for in the said data. The algorithms try to figure out certain rules and might group the data into various classes. For example, Clustering algorithms and Association rule learning algorithms (K-Means clustering)

- Semi-supervised Learning-A mix between the Supervised and Unsupervised algorithms. It is used only if there are a few labels on the data, but there are not enough labels to warrant a supervised algorithm.

- Reinforcement Learning- It is a branch of artificial intelligence. It is used in order to let the algorithm learn the basics of the data through multiple iterations through the given data set.

Figure 1.1: Machine Learning Types



In this project we have concentrated exclusively upon the supervised algorithms. Supervised algorithms can be broadly broken down into two sub divisions-

- Regression algorithms- Regression algorithms are used to predict continuous values. For example , Nave Bayes and KNN algorithms.

- Classification algorithms- Classification algorithms are used in order to predict discrete values. For example, linear regression and K Means algorithms.

For this project we have extensively used scikit Learn which is a module that is built on top of sci py library in python version 2 onwards. Scikit Learn is a python library that exclusively focuses on data science and the various classification, regression and cluster-ing algorithms including support vector machines, k-NN, random forests, Logistic Regression, gradient boosting, Nave Bayes, k-means, and Decision Tree, and is designed to operate within the python script for the Python numerical and scientific libraries NumPy and SciPy.

Libraries used

- NumPy which is Based on n-dimensional array package

- SciPy which is Fundamental library for scientific computing

- IPython is used for the Enhanced interactive console
- Sympy gives us the advantage of using Symbolic mathematics

Users: If this is a small indication of companies that have presented on their use, then there are very likely tens to hundreds of larger organizations using the library. It is good when we tend to do lots of test and it has managed releases and is suitable for prototype and production of the similar projects.

1.2 Brief history

In recent years we have observed a growing and worrying trend across the developed world. Medical costs are being driven sharply upwards in countries, who are recognized across the world as leading pioneers in science and industry. Countries include the United States and western European nations. The rising cost of medical treatment can be attributed to a number of factors including very high pricing for pharmaceutical drugs and the use of more expensive medical techniques , to name a few. Increasing medical costs seem to be thus intertwined with the overall development of a country.

As India slowly joins the ranks of world powers, it is not inconceivable to think that we might eventually face the same problem that the other developed nations grapple with currently. In order to preempt such a situation, it is vital to introduce more economic and viable alternatives within the medical industry.

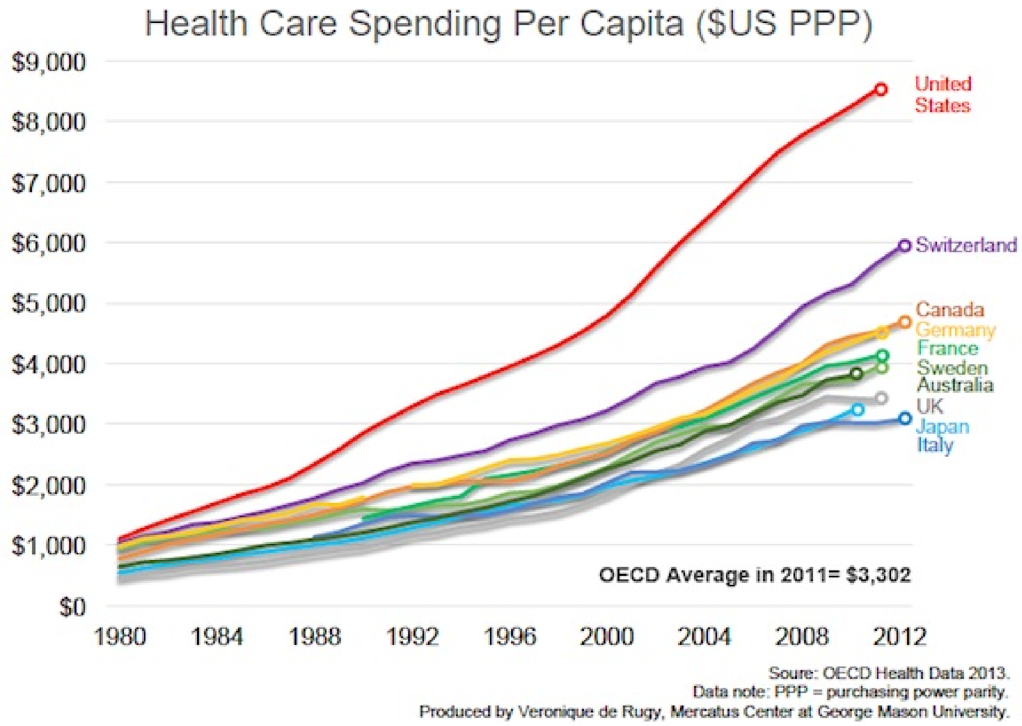
With the growth of machine learning, we can introduce disease diagnostic tools to the general public, lowering the need for them to depend on expensive medical treatment.

1.3 Architecture and characteristics

From the above diagram we can see the general representation of the project through its various phases. Steps

1. We have the given data set and the test data. Test data is used to train the given data set, by taking a small portion of the given data, and finding the underlying trends in said data. The logic is then applied to the entire data set, and a result is derived.
2. Data pre-processing must be carried out on both the test data and the data set. Pre-processing removes the inherent errors of the dataset, as well as converting the data set to a form that can be read by the computer.

Figure 1.2: Health Care Spending per Capita [6]



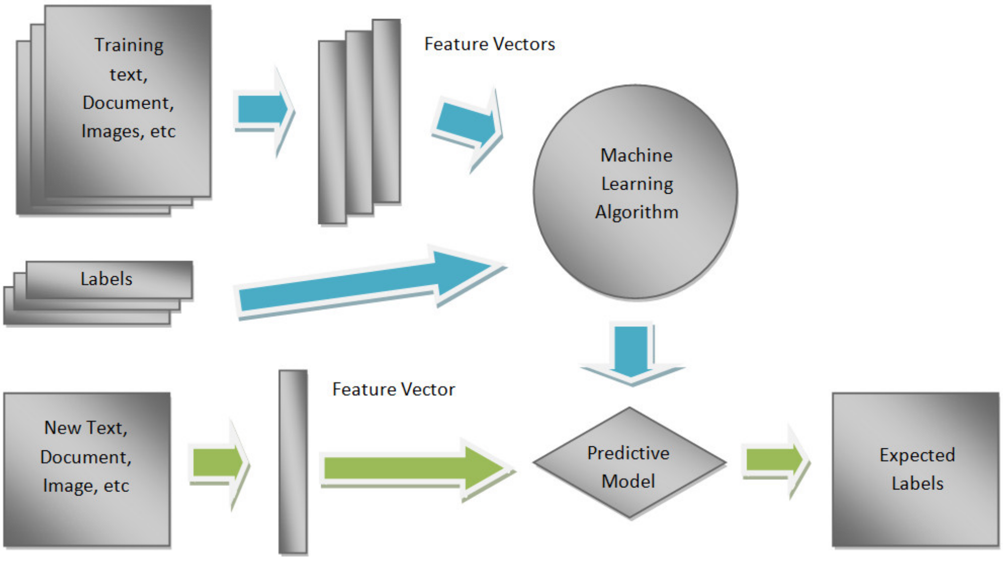
3. The given data set uses vectors in order to map values and deliver a coherent picture. For the test data, a simple text data vector is taken and results are mapped from that. Due to the size of the original data set, there we use 2 vectors, mainly the symptoms, and the disease. Thereby matching each symptom to the required disease. The probability that the disease is the one specified is then generated, based upon the algorithm used.

4. The vectors are then fed into a machine leaning algorithm. For this project, we have used algorithms such as Nave Bayes and Apriori. The machine learning algorithm then outputs a predictive model, based upon the algorithms parameters.

5. Predictive model, the predictive model is used to generate an accurate prediction og the given data set. The predicted model is then compared with the actual model, and the accuracy of the said system is measured. Measured accuracy is then determined to be an accurate value of the said system.

6. The final output is of course the predicted disease. It is then needed to see if the outputted disease is the same one that was should be generated from

Figure 1.3: Supervised Learning Algorithm


Training text, Document, Images, etc
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc
Feature Vector
Predictive Model
Expected Labels

the inputted symptoms.

1.4 Research motivation and Problem statement

1.4.1 Research Motivation

The objective of our project is to analyze the various machine learning regression algorithms, and compare the accuracy and performance of each on our disease symptom data set. Using the most optimum algorithm, we will then ask users to input various symptoms they might be having, and then output a list of likely diseases, with the predicted accuracy of such. Novel method have been used in this project, to improve the accuracy as well as the correctly identifying the required disease.

1.4.2 Statement of the Problem

This project will also attempt to answer questions based on the use of various algorithms including,

- Which is the best suited algorithm for the text based data ?
- What is the most optimum method of coding a given algorithm ?

- What is the most optimum way of converting text to tokens and then running an algorithm with it ?
- Various ways in order to pre-process the data ?

1.5 Research objectives

1.5.1 Primary objectives

The achievement of our primary objective could only be fufilled in modules, with the fulfillment of each part of the task, a step towards our final goal. Although the primary objective of the project has been succinctly defined in the project title itself i.e. Disease Diagnosis based on various machine learning algorithms, it has been a step by step process.

Our module objectives have been to hard code the various machine learning algorithms , including Naive Bayes , K means algorithm and KNN algorithm to name a few. Hard coding the algorithm, while also keeping in mind the given data set as well as the restrictions imposed upon it through the use of the given software.

As each algorithm calculated the given disease based upon the input parameters, we also measured the given accuracy of each algorithm and compared all the. The algorithm with the highest measured accuracy was taken as the primary algorithm to be used in the project.

Hence, through the above defined module based system we achieved the given primary objective of disease prediction, based on inputted systems.

1.6 Organization of the report

Chapter 2 deals with related work that has been undertaken in this field. It specifies the previous research that has been done in this domain. Chapter 3 relates to the system requirements that are needed to run the given program. Chapter 4 deals with the design of the project, both the architectural and the logical components of the given system. Chapter 5 deals with the actual code, containing snippets of screenshots of the code, along with the pseudo code. Chapter 6 contains the various test cases that we needed to run in order to test out the given program. Chapters 7 and 8 deal with the results as well as our aims for the future respectively.

Chapter 2

RELATED WORK

2.1 Inroduction

There has not been a System developed which is entirely based on detecting the Disease on giving Symptom as an input but there have been attempts to develop it. It is a very complex thing to do as the one symptom can be associated with many other diseases, which is why the reliability of the system is always a major concern.

2.2 Previous work

In [1], the authors propose an ontology, called Generic Human Disease Ontology (GHDO), which deals with the knowledge about human diseases with their types, causes, symptoms and treatments.

In [2],the database deals with number of patient cases as proto-type and stored in a separate database. The Disease of the Patients can be predicted with the help of patient datbase. The system has made use of production rules and a neural network.

In [3], the authors have tried to develop the system using artificial intelligence to identify the cause of the headache based on the data input by the patient.

In [4], the approach to develop the disease diagnosis system based on symptom is carried out by making the use of Bayesian network. To make machine understand, it creates the knowledge graph by the extracted information.

2.3 Algorithms

2.3.1 K-Nearest Neighbor

The KNN algorithm is one of the backbone of data mining and predictive analysis. The KNN algorithm stores all previous cases of such data and based on such, generates new cases and patterns. The KNN algorithm was one of the earliest cases of a predictive modelling algorithm. It uses statistical formula in order to measure the relative distance between each of the data nodes. The formulas used can be of various types including,

- Euclidean Distance
$$\sqrt{\sum_{i=1}^{k}}(x_i - y_i)^2 \tag{2.1}$$
- Manhattan Distance
$$\sum_{i=1}^{k} |x_i - y_i| \tag{2.2}$$
- Minkowski Distance
$$(\sum_{i=1}^{k} |x_i - y_i|^p)^{\frac{1}{p}} \tag{2.3}$$

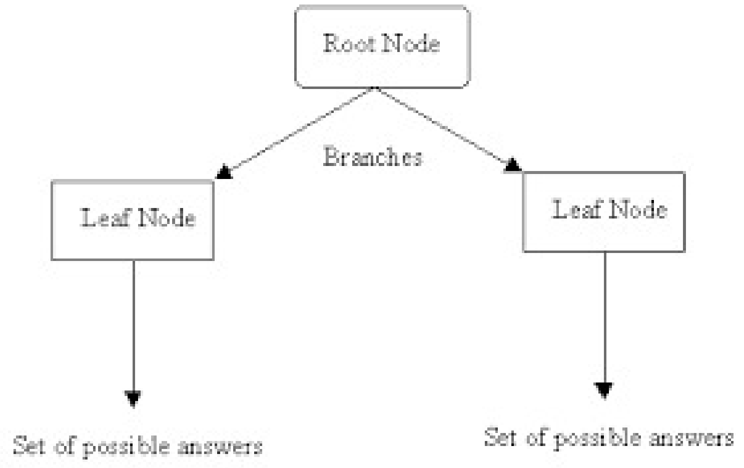
Data can only be analyzed if they are continuous variables. For non- continuous variables, we need to first standardize the data, Hamming distance and other standardizations are to be used in this case.

2.3.2 Decision Tree

Decision trees are used to approximate a discrete valued function, in which the final learned function is the decision tree itself. Decision trees are more useful than regular algorithms, as they can have improves readability. The tree does this by reducing all values to individual nodes, and then generating a simple if else case for each node pair. The Decision tree has been very useful in the various fields, from medical diagnosis (How we first approached the algorithm) and other fields such as stock market analysis.

Unfortunately some problems with decdion trees include , the problem that datafitted needsot be continous ,choosing the required attribute selecetion , seeing how deep the tree needs to grow and as with any algorithm , the computational speed at which it processes data .

Figure 2.1: Decision Tree



2.3.3 Naive Bayes

Nave Bayes is broadly based on the Bayes algorithm with one key difference. The Nave Bayes algorithm assumes that relationship between different parameters of the system do not exist. It is explicitly useful for large data sets, and despite its simplicity, it exceptionally performs other more sophisticated algorithms regularly. To calculate the conditional probability the formula is given below.

$$P(H|E) = \frac{P(E|H) * P(H)}{P(E)} \tag{2.4}$$

where

- P(H) is the probability of hypothesis H being true. This is known as the prior probability.
- P(E) is the probability of the evidence(regardless of the hypothesis).
- P(E—H) is the probability of the evidence given that hypothesis is true.
- P(H—E) is the probability of the hypothesis given that the evidence is present.

  As with other algorithms, it is required for the data to be preprocessed before it can be inputted into the algorithm. Here, the various numerical predictors is required to be converted into their categorical counterparts, and then inputted into various frequency tables. The main method by which we can achieve this, is by normal distribution.

– Mean

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.5}$$

– Standard Deviation

$$\sigma = [\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)]^{\frac{1}{2}} \tag{2.6}$$

– Normal Deviation

$$(\sum_{i=1}^{k} |x_i - y_i|^p)^{\frac{1}{p}} \tag{2.7}$$

Observe the above equations, as the root of the Nave Bayes equation. These equations are the final solution of the nave bayes equation. The root values of the data set are plugged into the equation and the data is then normalized. Like all data mining equations, the data must first be preprocessed and then set into its function.

2.3.4 Apriori

The Apriori Algorithm is an significant algorithm for mining frequent itemsets for boolean association rules. Apriori uses a ”bottom up” approach, where frequent subsets compute one item at a time, this step is known as candidate generation, and groups of candidates are tested beside the data. Apriori is designed to operate on database containing transactions.

The support of an itemset is the share of transaction in the database in which the item X appears. It signifies the popularity of an itemset.

First it creates a frequency table of all the items that take place in all the transactions. Then only those elements are significant for which the support is greater than or equal to the threshold support. The next step is to make all the possible pairs of the significant items keeping in mind that the order doesnt matter. After which we count the occurrences of each pair in all the transactions. only those itemsets are considered which cross the support threshold.

2.4 Summary

Based on the previous works done on Disease Diagnosis System, we can say that there is a scope of improvement in order to produce the probable results. We have implemented four algorithms to find the suitable one for the text based many-to-many dataset. These four algorithms are KNN, Decision tree, Apriori and Naive Bayes algorithm. By studying how the previous systems are implemented we can design our model for this particular Disease Diagnosis System which works on feeding the symptoms as an input.

Chapter 3

SYSTEM REQUIREMENTS SPECIFICATIONS

3.1 General Description

The requirements depends upon the various factors which are taken into consideration, for this project the requirements are based on algorithm used, programming language used and the data-set on which the project is based on. The main beneficiary of the application will be the doctors and the people who want know about the probable disease which they have, based on the output of system patients can choose the doctor of that particular specialty; also the doctor can check for other probable disease in order to have better diagnosis. The application needs to be compatible with the most of the laptops and household computer systems.

3.1.1 Product Perspective

The final application is coded on Python 3.0 and Python 2.7, so until the newer version of this application is developed which will be independent of the required environment, the product needs Python3.0 and Python 2.7 installed on the system. The user can use any IDE such as Spider and IDLE or can use the freemium open source distribution such as Anaconda, the application will run smoothly. The hardware requirements for these IDEs are the normal systems which are used for the normal day-to-day work so, no special characteristics are needed but in order to increase the performance and time efficiency one can have the system with better configuration such as higher RAM capacity, latest and high performance processors, having SSDs instead of HDDs, etc.

3.2 User characteristics

The Users of this system will be mostly doctors and the people who may encounter with some symptom of the disease which they are unaware off. It will be helpful for the doctors as the doctor can be reminded and cross-check with the possible diseases (to overcome human errors like diligence, versatile, tiredness). This can also be helpful for the patients to find out the diseases, when no other help is possible and also can go to specialist of that particular disease instead of going to the general doctor and getting referred which results in time and money wastage and causes human effort. The User need to input the list of Symptoms which they are experiencing. The output of the input given will be the probable disease with the more-likely probability.

3.3 System requirements

3.3.1 Hardware requirements

The Disease Diagnosis System is implemented on the Python, and to run Python on to systems, the recommended system requirements are :-

- Processors:
  - Intel Core i5 processor which runs at 2.60 GHz, and has 8 GB of RAM
  - Intel Xeon Phi processor 7210 which runs at 1.30 GHz, with 32 GB of RAM.
- Hard-Disk space: 2 GB to 3 GB
- Operating systems: Windows 10, macOS, and Linux

However, the minimum requirements are

- Processors: Intel Core i3 processor
- Hard-Disk space: 1 GB
- Operating systems: Windows 7 or later, macOS, and Linux

3.3.2 Software requirements

- Python versions: Pyhton-2.7.X, Python-3.6.X

- Included development tools: anaconda, anaconda-env, Jupyter Notebook (IPython)
- Compatible tools: Microsoft Visual Studio, PyCharm

The application is dealing with the Machine learning algorithms and the data-set which is completely text based so, python needs the following libraries and tools.

- PIP and NumPy: Installed with PIP, NumPy 1.13.1, scikit-learn 0.18
- Windows: Python 3.6.2, PIP and NumPy 1.13.1, scikit-learn 0.18.2

Functional Requirements

For the proper functioning of the machine learning algorithms, the foremost requirement is conversion of the text data into the numerical data, which is nothing but the data manipulation and the data pre-processing. This is done because the computer understands only the language of 0s and 1s; so, it requires numbers to work on, as the system do not understand words.

The data set is present is in the Disease and Symptom relationship i.e. response and feature is in textual form which needs to be converted into Response-Matrix and Feature-Matrix, this can be achieved by using Pandas for the importing the data-set, reading the data-set and selection of the required entities, with this we can manipulate the data to the desired form. For processing the data-set we can use the Feature-Extractors available in Sckit-learn Library. It makes easier for the system to learn the vocabulary and convert the vocabulary into Document-term-matrix.

Performance Requirements

The Disease Diagnosis system tells the probable diseases based on giving Symptom as input. This is related to the lives of human being, so the system is expected to be reliable. If the system is not reliable the consequences can be enormously dangerous as by wrong treatment and medication, the life of the being may come to critical condition.

For the system to be reliable it has to be accurate, the accuracy is checked by comparing the predicating values with the true values. If the system gives the expected predictions with its probability of the occurrence of the disease, it means that the accuracy is good and the system is reliable.

3.4 Summary

For this project, if the hardware of the system is better than the recommended hardware requirements it will result in the faster processing which in turn will give faster prediction. Also the systems need to have required software and tools installed in order to run the application, without which system will fail to process the request.

Having good accuracy results is must for our system as a human life is major concern, for that best algorithm or the combination of algorithm based on accuracy is chosen for the accurate results, which can avoid the confusion of the doctors decision and can save time.

Chapter 4

DESIGN

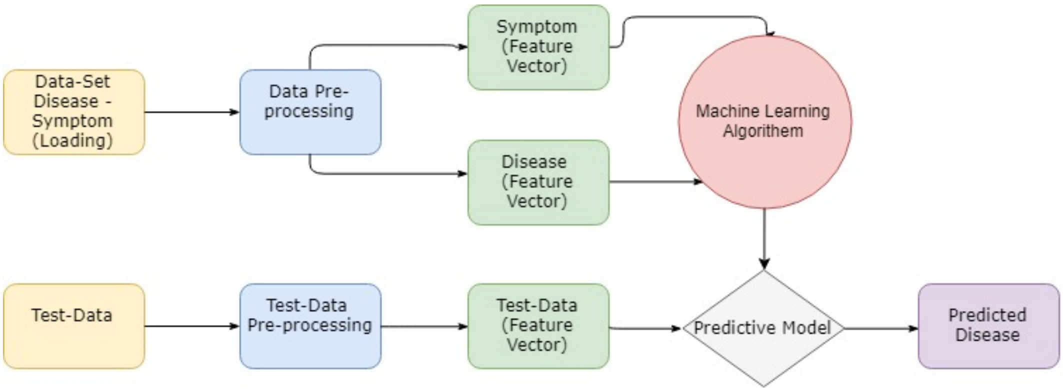
4.1 Architectural Design

System architecture refers to the underlying conceptual model of the system. This can refer to ta systems views, the structure of the system, as well as how the system behaves under certain conditions. In other words, a strong grasp of the system architecture can help the user understand the scope as well as the limitations of the said system.

The system architecture, like that of most other large scale structures, comprises of smaller components that we will attempt to individually explain. In addition to this, we will also explain how the various system components come together.

Some of the various parts of our system are-

Figure 4.1: Flow Chart Disease Diagnosis System



- Data Set: One of the key principles of data mining is the ability to extrapolate the underlying trends inherent in the data, by sampling a relatively small size of the data. The conclusions drawn from the small sample are then mapped onto the entire data set. The accuracy of the conclusions reached is then measured.

  The sample data that is taken is known as the training data. The training data is randomly taken form the overall data set. Testing data is a larger sample of data that is also taken from the overall data set. The conclusions reached from the training data set are applied from the testing data set, and as previously explained, the accuracy in measured. Models require the use of similar data for the training and testing data. That helps in the reduction of data discrepancies, as well as better understanding the underlying model of the system.

- Data preprocessing: Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real world data requires cleaning and preprocessing before it can be handled by the algorithm. This is due to the obvious reason that real world data often contains errors that the algorithms that are used, cannot process such errors. Data Preprocessing thus takes this raw data and further processes it, removing errors , and saves it for further analysis.

  Data goes through a sequence of steps during preprocessing:

  – Data Cleaning: Data is cleansed through various methods. These methods include the addition of data, for example filling in data slots that are left blank. Or data reduction, for example the removal of commas or other unknown characters.
  – Data Integration: In this step, the data is integrated from various sources. The data is then checked for any integration errors and they are summarily handled.
  – Data Transformation: Data in this step is normalized based upon the given algorithm. Data normalization can be done through a variety of methods. This step is required in most data mining algorithms, as the data inputted needs to be as clean as it can. Data is then combined and built up.

  - Data Reduction: This step in the process aims to reduce the data to more manageable levels.
  - Data Discretization: Involves the reduction of a number of values of a continuous attribute by dividing the range of attribute intervals. Data set and test data separating data into training and testing sets is an important part of evaluating data mining models.

- Data set and test data The data set is separated into the training and testing data sets. The data is usually split into two parts. The testing set is usually a smaller part of the overall data set. The training data is used to calculate the underlying patterns of the same. Testing then sees if the patterns hold. Similar data is needed for training and testing, usually derived from the same data set. This helps us understand the underlying characteristics of the data.

  After the model has been preprocessed by the use if the training data set, the second step is to test the accuracy of the system. We can do this by plugging in various examples that are run against the testing data set. For most algorithms, the testing part of the model is done automatically, that is data examples are taken from the underlying data set without outside input required. Hence the entire testing process is done fairly quickly and an accuracy measurement is generated. For a third verification, the user can give values that are not in the data set, to cross verify. This is however not usually needed as the data taken up automatically by the process is enough to generate a fairly exact accuracy measure.

- Predictive model: Predictive modeling as the name suggest involves the prediction of events that have yet to happen. Although primarily concerned with the prediction of future events, the principles of predictive modeling can be applied to any instance, where there is an unknown value to be found.

  Predictive modeling techniques are used to calculate the probability of a given occurrence happening, based on a set of input factors. This makes it highly useful in disease prediction, as it can be used to predict what disease the individual is suffering from, based upon a persons input symptoms and previous diagnoses. Another example could be the probability of traffic on a given day, based on the date and other environmental factors.

  Modeling of this kind is broadly based upon machine learning, and the two fields overlap considerably. Predictive modeling has branched into several different fields including military applications , and has attracted considerable interest in commercial fields; where is it commonly called as predictive analysis.

- Predicted Disease based upon the various algorithms in the end we have a final output of a disease that has been predicted based upon certain input symptoms.

Chapter 5

IMPLEMENTATION

5.1 Methodology

For the implementation of this project, we have used various methods, for the collection and the analysis if the data. In addition to this, each algorithm that has been used has its own pros and cons that have been adequately explained in the earlier sections. The usefulness of each algorithm will be proved by their accuracy that will be specified in chapters 6 and 7.

Data collection-
For the collection of data, we have used a standard data set that was generated by the United States department of health and human services. All algorithms that we used, use this standard data set. For our project, we required a data set that contained a collection of diseases, along with at least six to seven symptoms describing each disease. The data set obtained from the aforementioned governmental study fulfills all the qualities that we needed from a said data set. However, we have made some minor adjustments to the set regardless. Cleaning the set from typing errors and other small mistakes that would create uncertainty in our final result.

Language and environment used-
The entire project was carried out in the python language, this includes the actual algorithms, as well as the website and back end connections, from which users could input the various symptims.

Python is a dynamic language that was built relativity recently compared to other programming languages such as c and java. Hence python tends to fix a lot of the problems that were encountered by the users of said languages Python has an in built garbage collector as well as support for

Unicode. Python emphasized white space, enhancing code readability as well as an increased likelihood of large scale projects to be built due to the same. Python also contains various libraries that are specifically designed to help in the generation of machine learning projects. Libraries such as sci-kit learn, which we have used extensively in this project. In addition to this, there are python interpreters such as Anaconda, which support other interpreters such as sypder and the Jupyter notebook. This is in addition to the normal python idle that helps tremendously in the support of machine learning projects. It is due to all of these reasons, from the general python language, to the various libraries, along with the various interpreters, that caused us to choose this language over several others.

Algorithms used- The algorithms used, were standard machine learning algorithms, whose use and general execution has been explained in depth in the chapters preceding. We could not however use all the machine learning algorithms, as we had expected to do at the start of the project. Machine learning algorithms, as have been previously defined in the preceding chapters are divided into two types, regression and classification algorithms. As classification algorithms are not useful in those type of analysis, we were forced to shelve plans involving the same.

Of the algorithms left, we chose those that were regression based. In addition to those, we chose algorithms that were more useful I text based analysis. These were the Aprioi and the Naive Bayes algorithm .However in addition to these two, we also explored decision trees and the KNN algorithm, both of which generated the required output, but to a degree that was far less accurate than the previous two algorithms.

The required accuracy and the final disease types are however faithfully executed by all these algorithm s, and hence their inclusion in the same.

5.2 Process

Although there are many different variations of the algorithms that we have chosen, it takes only a certain method in order to process the data set that we have selected. Obvious pre requites being that it need to be a classification algorithm, as well as an algorithm that works well with various text based data sets. Nave Bayes algorithm and the Apriori algorithm seem to handle the change from a text based data set, to the generation of meaningful analysis. Other algorithms such as KNN or the simpler linear regression, do not seem to hold the same amount of accuracy as the others.

Although each algorithm works on a set of different principles, the general process used in order to generate a meaningful analysis is broadly the same. Described below, is the set of steps we undertook in the analysis of each algorithm.

(a) Obtaining the data set: Although a crucial obvious first step in all project. Some analysis run on the inclusion of multiple data sets, some of which are created, solely for the purpose of the project. In our project however, we have obtained an already existing data set that was tailor made for what we had envisioned as the project goal. This disease data set, is a compilation of more than two hundred and fifty diseases along with a list of their corresponding symptoms. The selected data set, is used as the standard pre-processed data that each algorithm uses. The use of a single data set greatly simplifies the accuracy comparison of the sated algorithms. This Dataset is a knowledge database of disease-symptom associations generated by an automated method based on information in textual discharge summaries of patients at New York Presbyterian Hospital admitted during 2004 [5].

(b) Pre-processing the data: Data comes in many different forms, from the standard excel sheets, to a spreadsheet of radio waves. The raw data that is obtained, might contain errors stemming for the collection methods, or simple human errors. Data obtained from the world contains numerous problems, including incomplete data (attribute values are missing or it lacks certain attributes that are useful), noisy data (Contains various errors), inconsistent (data that contains discrepancies). A crucial part of the data mining cycle is the cleaning of this data, and the conversion of the said data into a more understandable form, this is usually a.csv file, or any other simple file format, that the algorithm can read easier. Data pre-processing is further divided into a few parts.

- Data Cleaning: Data cleaning is broadly concerned with the removal of old, incomplete or deleted data. Various data cleaning methods may use parsing methods in order to clean out data that is redundant , along with various code fragments, and other undesirable bit of information that distract from the readability of the data.
- Data Integration: As the name suggests, data integration involves the integration of data from various sources into a single database or storage device. Data integration is the primary method of building large scale enterprise databases. In this project, due to the

practical reasons of not using multiple data, data integration is not wholly supported in this project, and yet forms a very important part of data pre-processing.

- Data Transformation: This crucial step involves the transformation of the data, into a form that is more acceptable to the given algorithm. The data needs to be transformed, without the underlying meaning behind the data being lost. Data transformation can be of various types, depending on the type of data that is being processed as well as what the final transformation of the data is intended to do. Transformations such as smoothing (removing noise from all data), Normalization (Scaled to fall within a certain range), Attribute/ feature construction, Aggregation and Generalization are all various types of data transformation. Normalization can be further divided into types such as min-max normalization, and z-score normalization.

  min-max normalization :

  $$v' = \frac{v - min_A}{max_A - min_A}(newmax_A - newmin_A) + newmin_A \tag{5.1}$$

  z-score normalization :

  $$v' = \frac{v - mean_A}{standdev_A} \tag{5.2}$$

  In reference to our project, we have transformed the given text based code into tokens, by the means of an inbuilt tokenize. The tokens are then assigned a given weight and plugged into the required algorithm. Tokenization is a crucial step, and the transformation of the data from a text based data set, into that which is a numeric format, is one of the cornerstones of our analysis. The tokenized values are then to be reconverted to the numeric format at the end of the algorithmic process.

- Data reduction: Data reduction, as the name suggests like data integration, involves the reduction of very large scale data sets into smaller more manageable chunks. The challenge here is to have a data set that has been sufficiently reduced and yet maintains the underlying integrity of the dataset. Various strategies of data reduction are, Dimensionality reduction (Removal of unimportant attributes), Aggregation and clustering (Removal of redundant or closely related data sample sets) and sampling.
- Dimensionality Reduction: It deals with changing the very attributes of the system. Here, the attributes are scanned for unimportant attributes that can skew the analysis of the system. The

said attributes are then identified and then deleted. There are multiple ways to go about this, some of these methods are -
Direct Methods:

- Feature selection (i.e., attribute subset selection)
- Select a minimum set of attributes (features) that is sufficient for the Data mining task.

Indirect Methods:

- Principal component analysis (PCA)
- Singular value decomposition (SVD)
- Independent component analysis (ICA)
- Various spectral and/or manifold embedding (active topics)

Heuristic methods (due to exponential number of choices):

- Step-wise forward selection
- Step-wise backward elimination
- Combining forward selection and backward elimination
- Combinatorial search exponential computation cost

Another popular method of data mining is buckets, of which we have used extensively in our project, especially with the inclusion of the Apriori algorithm. This is the Discretization Method which is used to assign bucket(Symptom) to each value(Disease) of the collection.

(c) Splitting of the data set: After the data is processed, and the satisfactory processed data is generated, we come to the actual data analysis. At its most basic, data analysis involves, finding an underlying relationship between a small sample sizes of data, and then applying it to a larger fraction of the data. To accomplish the same, we divide the data set into two parts, mainly the training and testing parts. The algorithm is run on the training set of the data, and the underlying logic is disseminated. It is then applied to the larger testing data set, with the final values being tested for accuracy within the system. The training model is fitted upon the dataset using a supervised learning method.

(d) Observing the output: With the final generation of the output through the various algorithms, we have, we required a suitable mechanism for viewing the final generated disease. By the use of a simple python based website, the disease symptoms are inputted into a search bar, and the required diseases are outputted along with the percentage accuracy of the same. The system uses python and HTML to generate a front end website. Python is also used to fetch the data form the algorithm as well, on the back end. In the produced output, the probable percentage will be shown along with the probable disease.

5.3 Pseudocode

5.3.1 K-Nearest Neighbor Algorithm

Algorithm 1 *K-Nearest Neighbor*

```
Classify(X,Y,x)//X:training data (Symptom), Y:class labels of X(Disease),
x: unknown sample
for i=1 to m do
Compute distance d($X_i$, x)
end for
Compute set I containing indexes for k smallest distances $d(X_i, x)$
return majority label for $\{Y_i$ where $i \in$ I$\}$
```

5.3.2 Naive Bayes

Algorithm 2 *Naive Bayes*

```
predict(testfeatures):
for x in testfeatures:
for y in featuresft[x]:
count=count+y
end for
count=count+1 (extra 1 added to count to compensate for 0 probability(count
stores frequency of each feature))
featurespriorprob=featurespriorprob*(count/(datasetdf['Symptoms'].count()));
end for
diseaseinfoindexed = diseaseinfo.setindex('Disease')
for x in diseaseinfo['Disease']:
for y in testfeatures:
for z in featuresft[(diseaseinfoindexed.at[x,'beginindex']):(diseaseinfoindexed.at[x,'endindex'])]
count=count+z
end for
count=count+1extra 1 added to count to compensate for 0 probability(count
stores frequency of each feature)
likelihood=likelihood*(count/(diseaseinfoindexed.at[x,'classcount']))
```

```
end for
classprob=(likelihood*(diseaseinfoindexed.at[x,'classpriorprob']))/featurespriorprob
if(classprob ¿ predictedclassprob):
    predictedclassprob=classprob
    predictedclass=x
    likelihood=1
close if
print("Disease:",predictedclass)
```

5.3.3 Apriori Algorithm

Algorithm 3 *Decision Tree*

```
getdisease(Inputsymptom,buckets):
for bucket in buckets:
    score = set(Inputsymptom) AND set(bucket)
    score = float(len(score))/float(len(Inputsymptom))*100
    if score greater than 50:
        print (score)
        disease = getdiseasegivenbucket(bucket)
        print (disease)
        diseasescore[disease] = score
    close if
    print (diseasescore)
end for
```

here

- Inputsymptom : List of symptoms entered by the user(Input)
- buckets : Mapping of Disease and Symptoms
- buckets : Mapping of Disease and Symptoms based on Inputsymtom
- score : probability of the probable disease
- getdiseasegivenbucket() : function to obtain the disease name from the bucket list

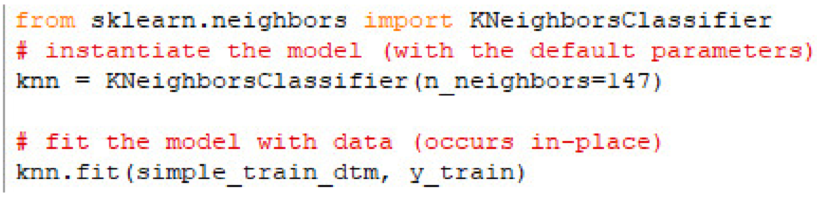
Chapter 6

TESTCASES

6.1 Case 1

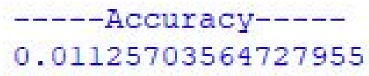
We implemented the K-nearest neighbor by using the function provided by Scikit-learn.

Figure 6.1: KNN - implementation



```
from sklearn.neighbors import KNeighborsClassifier
# instantiate the model (with the default parameters)
knn = KNeighborsClassifier(n_neighbors=147)

# fit the model with data (occurs in-place)
knn.fit(simple_train_dtm, y_train)
```

Figure 6.2: KNN - Accuracy



```
-----Accuracy-----
0.01125703564727955
```

But the accuracy which we got was very less to be precise it was 0.01125703564727955.

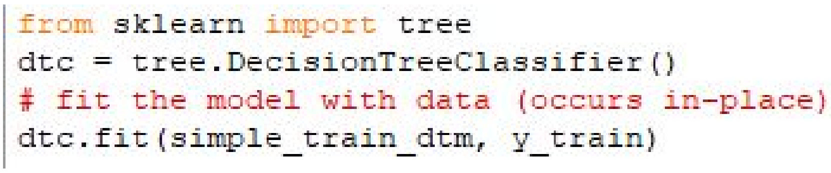
This happened because the dataset is Many-to-many structured and also is text-based where as the in-built functions best works on one-to-one and one-to-many numerical dataset. But still we tried to implement it using another algorithm Decision Tree.

6.2 Case 2

We implemented the Decision Tree by using the function provided by Scikit-learn.

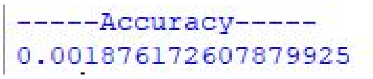
But again the accuracy very less to be precise it was 0.001876172607879925.

Figure 6.3: Decision Tree - implementation



```
from sklearn import tree
dtc = tree.DecisionTreeClassifier()
# fit the model with data (occurs in-place)
dtc.fit(simple_train_dtm, y_train)
```

Figure 6.4: Decision Tree - Accuracy



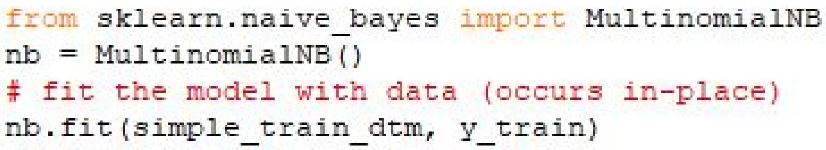
```
-----Accuracy-----
0.001876172607879925
```

This happened because the dataset is Many-to-many structured and also is text-based where as the in-built functions best works on one-to-one and one-to-many numerical dataset. We gave another shot by implementing the Naive Bayes Algorithm.

6.3 Case 3

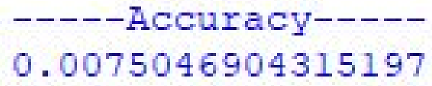
We implemented the Naive Bayes by using the function provided by Scikit-learn. The results were expected to be good as it is probabilistic model but unfortunately the results were totally wrong.
But again the accuracy very less to be precise it was 0.007506904315197.

Figure 6.5: Naive Bayes - implementation



```
from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB()
# fit the model with data (occurs in-place)
nb.fit(simple_train_dtm, y_train)
```

Figure 6.6: Naive Bayes - Accuracy



```
-----Accuracy-----
0.0075046904315197
```

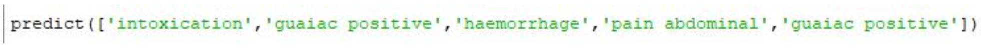
This happened because the dataset is Many-to-many structured and also is text-based where as the in-built functions best works on one-to-one and one-to-many numerical dataset. Now we concluded that bulit-in functions cannot work on text-base many-to-many dataset. Now we moved on to implementing by using Naive Bayes by altering it according to the dataset and required output.

6.4 Case 4

We implemented the Naive Bayes by coding it manually. For doing so we needed to convert the text-based dataset into numerical form, we did it by using the built in functions, hence tokenization was achieved. The results were good as it is probabilistic model but unfortunately with some restrictions.

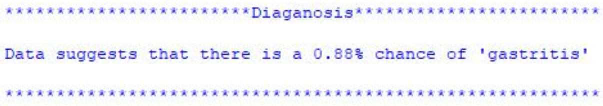
6.4.1 Symptoms belonging to same Disease

Figure 6.7: Naive Bayes - input-1



```
predict(['intoxication','guaiac positive','haemorrhage','pain abdominal','guaiac positive'])
```

Figure 6.8: Naive Bayes - Output-1

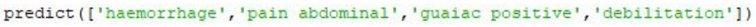


```
************************Diaganosis***********************

Data suggests that there is a 0.88% chance of 'gastritis'

*********************************************************
```

Above input and output pictures represents the input symptoms belonging to the same disease.

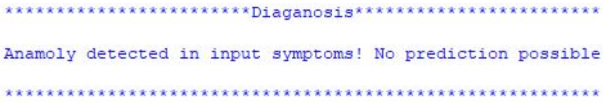
6.4.2 Symptoms belonging to different Disease

Figure 6.9: Naive Bayes - input-2



```
predict(['haemorrhage','pain abdominal','guaiac positive','debilitation'])
```

Figure 6.10: Naive Bayes - Output-2



```
************************Diaganosis************************

Anamoly detected in input symptoms! No prediction possible

**********************************************************
```

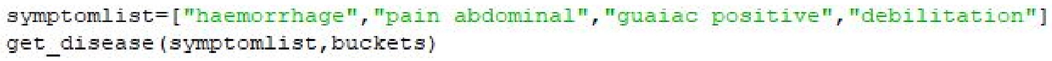
Above input and output pictures represents the input symptoms belonging to the same disease.

We always got the correct answer but with some restrictions. the restrictions were we needed to input the symptoms present in the dataset. Also the system cannot tell the probable disease if the input symptoms belong to different Diseases. After which we tried to implement

6.5 Case 5

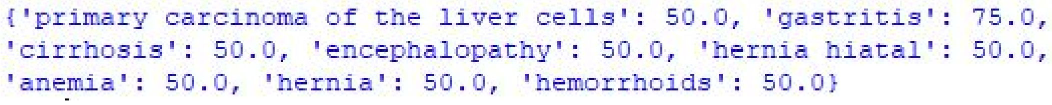
We implemented the Apriori Algorithm by Coding it manually.In order to do so we were required to make buckets witch is basically mapping up all the symptoms to its diseases.

Figure 6.11: Apriori - Input



```
symptomlist=["haemorrhage","pain abdominal","guaiac positive","debilitation"]
get_disease(symptomlist,buckets)
```

Figure 6.12: Apriori - Output



```
{'primary carcinoma of the liver cells': 50.0, 'gastritis': 75.0,
'cirrhosis': 50.0, 'encephalopathy': 50.0, 'hernia hiatal': 50.0,
'anemia': 50.0, 'hernia': 50.0, 'hemorrhoids': 50.0}
```

By implementing apriori algorithm, this system we were able to detect the probable Disease even if the input symptoms belong to different Disease also it gives the probability of that particular Disease, but probability is based on the frequent dataset produced by the available dataset.

Chapter 7

RESULTS

Our team has used supervised machine learning algorithms to develop a system that allows the doctor to predict probable diseases based on patient symptoms as observed which results in better diagnosis and further treatment. This system was designed and implemented by analyzing the dataset obtained from New York Presbyterian Hospital through the data collected by patient diagnosis in the year 2004 by exploring various supervised learning algorithms.

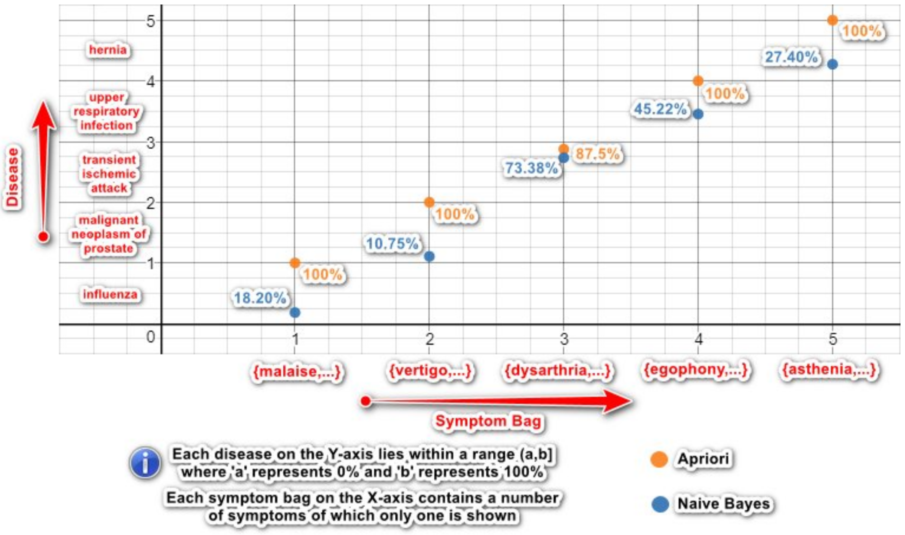
Based on our exploration of various supervised machine learning algorithms on the dataset, Apriori and Naive Bayes algorithms showed better results compared to k-nearest neighbors and Decision Tree algorithms which showed relatively poor performance. Hence final system was designed through the implementation of Naive Bayes and Apriori algorithms that were finally used for disease diagnosis.

To compare the two working algorithms of our project, The Naive Bayes and The Apriori Algorithm we have plotted the scattered graph between the symptoms and the predicted disease and the probability of the occurrence of the predicted disease.
Below is the table of the set of symptoms, Diseases and the the probability of the occurrence of the predicted disease.

DISEASE	SYMPTOMPS	PROB. NAIVE BAYES	PROB. APRIORI
INFLUENZA	uncoordination, pleuritic pain, snuffle, throat sore, malaise, debilitation, dysuria	18.20	100.0
MALIGNANT NEOPLASM OF PROSTATE	passed stones,qt interval prolonged, dysuria, vertigo, paresis, hemianopsia homonymous, tumor cell invasion, hemodynamically stable, orthostasis	10.75	100
TRANSIENT ISCHEMIC ATTACK	syncope, room spinning, headache, Stahliś line, extrapyramidal sign, dysarthria, speech slurred, vertigo	73.38	87.5
UPPER RESPIRATORY INFECTION	rapid shallow breathing, egophony, indifferent mood, labored breathing, cystic lesion	45.22	100
HERNIA	gag, hyperventilation, excruciating pain, nausea, posturing, pain, pain abdominal, asthenia	27.40	100

Based on the above table the Scattered Graph is plotted having symptoms on the x-axis and diseases on the y axis.

Figure 7.1: Naive Bayes and Apriori Analysis



Based on the analysis we can say that both algorithm are working fine but apriori algorithm is giving better results as the this algorithm is based on frequent item dataset. Moreover it gives us probability of symptom corresponding to all the disease. In that way a person will be able to know if he/she is having more than one disease.

Since machine learning algorithms are dependent on datasets based upon previous knowledge obtained from the physical diagnosis, the volume of dataset must be comprehensive with minimum number of outliers. As the data set grows more and more, diseases are added, the scope of diagnosis increases with better predictions in the upcoming future.

Chapter 8

CONCLUSION AND FUTURE WORK

Based on the various algorithm implementations, we have concluded that algorithms such as Nave Bayes and Apriori are highly useful in the implementation of the given data set. Based on the primary objective, we can conclude that all primary goals have been met, with the disease being predicted based on the input symptoms, using multiple algorithms. The accuracy of said algorithms must then be taken and a comparison done, to see what the most accurate algorithm, based on text is based input system.

Although the above mentioned algorithms are working perfectly, to get a large enough set of algorithms , we must hard code algorithms such as SVM. We shall then compare these models to those we have already implemented in order to check the given accuracy. In addition to this , more work needs to be done in refining and expanding the given data set, as the current data set only contains 148 diseases from an estimated ten thousand that currently afflict humans in one way or the other. The expansion and the refinement of the given data set will give us a more accurate set of values, and a more error free output.

Although the use of the algorithms is needed, for an accurate diagnosis, a multitude of various factors are taken into account, which is one reason why human professionals are still required in the medical industry. Some factors might not even be inputted by the user, causing a completely wrong diagnosis.

Bibliography

[1] Maja Hadzic and Elizabeth Chang. *Ontology based Multi agent Systemn support Human Disease Study and Control.* Proceedings of the conference on Self Organization and Autonomic Informatics 2005, pp.129-141.

[2] Dipanwita Biswas, Sagar Bairagi , Neelam Panse and Nirmala Shinde. *Disease Diagnosis System* International Journal of Computer Science Informatics, Volume-I, Issue-II, 2011

[3] Anthony Farrugia, Dhiya Al-Jumeily, Mohammed Al-Jumaily and David Lamb *Medical Diagnosis: are Artificial Intelligence systems able to diagnose the underlying causes of specific headaches?* Sixth International Conference on Developments in eSystems Engineering 2013.

[4] Prerna Agarwal, Richa Verma, Anupama Mallik. *Ontology Based Disease Diagnosis System with Probabilistic Inference* 2016 1st India International Conference on Information Processing (IICIP).

[5] New York Presbyterian Hospital 2004. *Disease-Symptom Knowledge Database* AMIA Annu Symp Proc. 2008. p. 783-7. PMCID: PMC2656103. http://people.dbmi.columbia.edu / friedma /Projects /DiseaseSymptomKB /index.html

[6] Veronique de Rugy, Mercalus Center at George Mason University. *Health Care spending per capita ($US PPP)* OECD Health Data 2003.

Appendix A

PAPER

Appendix B

PLAGIARISM REPORT